Threat Assessment in the Cloud Environment – A Quantitative Approach for Security Pattern Selection

Priya Anand, Jungwoo Ryoo
Department of Information Sciences and Technology
Pennsylvania State University
University Park, PA, USA - 16802
{axg36,jryoo}@ist.psu.edu

Hyoungshick Kim, Eunhyun Kim
Department of Computer Science and Engineering
Sungkyunkwan University
Suwon, Gyeonggi-Do, Republic of Korea
{hyoung,eunhyun}@skku.edu

ABSTRACT

Cloud computing has emerged as a fast-growing technology in the past few years. It provides a great flexibility for storing, sharing and delivering data over the Internet without investing on new technology or resources. In spite of the development and wide array of cloud usage, security perspective of cloud computing still remains its infancy. Security challenges faced by cloud environment becomes more complicated when we include various stakeholders' perspectives. In a cloud environment, security perspectives and requirements are usually designed by software engineers or security experts. Sometimes clients' requirements are either ignored or given a very high importance. In order to implement cloud security by providing equal importance to client organizations, software engineers and security experts, we propose a new methodology in this paper. We use Microsoft's STRIDE-DREAD model to assess threats existing in the cloud environment and also to measure its consequences. Our aim is to rank the threats based on the nature of its severity, and also giving a significant importance for clients' requirements on security perspective. Our methodology would act as a guiding tool for security experts and software engineers to proceed with securing process especially for a private or a hybrid cloud. Once threats are ranked, we provide a link to a well-known security pattern classification. Although we have some security pattern classification schemes in the literature, we need a methodology to select a particular category of patterns. In this paper, we provide a novel methodology to select a set of security patterns for securing a cloud software. This methodology could aid a security expert or a software professional to assess the current vulnerability condition and prioritize by also including client's security requirements in a cloud environment.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

IMCOM '16, January 04-06, 2016, Danang, Viet Nam
 ISBN 978-1-4503-4142-4/16/01$15.00.
DOI: http://dx.doi.org/10.1145/2857546.2857552

General Terms

Security, Management

Keywords

Cloud Computing, Threat Assessment, STRIDE-DREAD Model, Security Patterns, Risk Analysis

1. INTRODUCTION

In recent years, cloud storage has provided ubiquitous and easily-accessible remote data repositories to clients. While many applications of cloud computing require security assurances like confidentiality, integrity, and availability from cloud providers, current research and development in cloud computing environments demonstrates that there are still many shortcomings in these areas. Cloud software is primarily developed and implemented by software/IT professionals. They review the organizational needs and requirements during the requirement-engineering stage. Regardless, we cannot assume software developers to be security experts who can estimate the vulnerabilities that can be introduced by poor software-coding practices. Neither software professionals nor security experts are the sole responsible party for the existence of any software vulnerabilities in a system. Software security [16], a relatively new field, aims to offer assurance that software will function correctly even during malicious attacks. McGraw states that *"software security best practices leverage good software engineering practice and involve thinking about security early in the software lifecycle, knowing and understanding common threats (including language-based flaws and pitfalls), designing for security, and subjecting all software artifacts to thorough objective risk analyses and testing"* [16]. Hence, we need software security professionals to be part of any organization to bridge the gap that may exist between software professionals and security experts. Together via the field of software security, they are able to provide a more comprehensive analysis of a system and identify potential vulnerabilities more accurately. A vulnerability can be defined as *"a flaw or weakness in an information asset, security procedure, design, or control that could be exploited accidentally or on purpose to breach security"* [20]. Any flaw or weakness can be a threat to a system, which eventually turns into a security loophole for attacks, and it may be exploited by the threat agents/hackers. As a scientific fact, we cannot develop a vulnerable-free system, only discovered vulnerabilities can answer the question *"What can happen?"* In other words if a user is completely unaware of the system's vulnerabilities, he/she cannot pos-

sibly imagine what could go wrong. Therefore, estimating risk factors becomes essential for an accurate vulnerability assessment of the system. When we deeply analyze the effects of a vulnerability in a cloud (or any information system), the controlling organization would undoubtedly face a serious challenge in overcoming the threat, but ultimately the victims of this scenario would be the end-users or clients. Security awareness has become crucial and inevitable while dealing with any kinds of cloud systems. When a vulnerability is exploited by a threat agent, all parties involved are affected by the damage and must face the consequences. Security is a shared responsibility and every party involved in a cloud environment must take a certain level of responsibility or role in the whole process. While we cannot expect the software professionals to be experts in security policies, users or clients cannot be expected to be aware of software vulnerabilities. Therefore, cloud developers should take a lead role in fixing software vulnerabilities to gain their clients' trust on their service.

In early 1960s, information systems were developed to meet the needs of marketing and organizational database management as well as to enhance information technology. By that time, security was not a large part of the software development process. Only when software systems were hacked by attackers did the need for secure coding become accepted among software architects. Apart from natural disasters like fires, floods, or other acts of God, all attacks are initiated by human agents. Inherently, software vulnerabilities are not intentionally introduced by the developer, but they are successfully identified by the attacker. Software may contain many vulnerable codes in it and believed to be a secure system, until vulnerabilities are identified. If these vulnerabilities are identified by a hacker or a malicious insider first, they can become the launching point for an attack. If it is identified by a security professional or software architect, it helps in the security enhancement process. In other words, it depends on who identifies the vulnerabilities first. Obviously, there is a huge expectation on cloud developers to identify threats, calculate risk and fix the vulnerabilities in a cloud. However, it is a challenging task for software professionals to assess vulnerabilities and fix security loop holes with essential software security protocols.

In this paper, we propose a new methodology for threat assessment in cloud environments, which would act as a guiding tool to select the required security patterns from a pattern classification scheme. We identify some cloud-based threat factors based on the Microsoft STRIDE-DREAD [10] model. Our proposed methodology would begin its evaluation from software architects of client organizations by assessing their security requirements and priority for threat tolerance. After assessing client-side specifications, using the DREAD model, we evaluate the risk associated with each threat category using a 0, 5, or 10 degree scale. Using the calculated risk factor and user threat tolerance level, we form a threat assessment matrix and security index for each threat category in the STRIDE model. This resulting security index is ranked in a descending order to identify the seriousness of the threat faced by the clouds. We relate our threat ranking to an existing security pattern classification proposed by Hafiz et al. [2] to identify potential security patterns to fix those threats. We propose the pattern classification by Hafiz et al [2] as a potential solution to select a security pattern mainly due to the fact that classification is based on the STRIDE threat model. Our proposed methodology would act as a guiding lens to assess the threats in the order of their intensity and eventually choose a security pattern based on the threat assessment results. Security patterns [2], provides architectural solutions to recurring system problems and define a way to express requirements and solutions concisely as well as to provide a communication vocabulary for designers. To show the predictability of our proposed approach, we demonstrate an example using XEN Hypervisor vulnerability lists and assess the serious threats facing a XEN cloud environment. Our proposed approach includes dynamic perspectives from all stakeholders involved in the system. Although it is primarily meant to aid cloud developers to improve security features by auditing the underlying software, it also includes clients' security and privacy requirements in the initial stage of analysis, and it bridges software security features as the ending domain by linking a known security pattern classification scheme. This approach leads to a ground where cloud developers, clients, and software security experts could find a common spot to share their requirements and find a solution that is acceptable to all the parties. Our proposed methodology would be very beneficial to clients when they decide to adopt a private or a hybrid cloud for their organization.

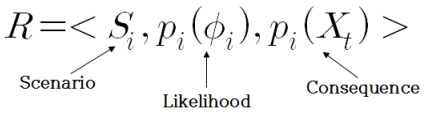

Figure 1: Kaplan's Definition of Risk. Source: Kaplan, Stanley, and B. John Garrick. "On the quantitative definition of risk." ***Risk analysis***

1.1 Role of risk evaluation in threat assessment - Theoretical perspective

Kaplan and Garrick argued that when we claim, *"What is the risk?"* we raise three different questions: *"What can happen? (i.e., what can go wrong?), How likely is it that will happen? If it does happen, what are the consequences?"* [13]. Authors formulate a triplet (S_i, L_i, X_i) that provides an answer to these questions, where S_i represents risk scenarios, L_i denotes the frequency that scenario *'i'* can occur and X_i represents the damage index, which is a measurement of consequences. Hence, risk is defined as the complete set of these triplets. Kaplan's definition of risk, R, is represented in Figure 1. We can imagine likelihood, frequency and probability to represent the same meaning, but the meaning differs in its own context [13]. 'Frequency' refers to the outcome of repeated design, an experiment that is "objective" in nature. Probability represents the degree of confidence on an uncertainty, based on an evidence-based mathematical procedure. Bayesian Theorem, as used in Kaplan's risk definition, states that those kinds of numbers do not exist in the real world but in our brains [12]. Kaplan uses the Bayesian definition on "subjective" probability to represent the likelihood of a particular scenario to define risk.

We can retrieve the data from the history of previous at-

tack to prove a hypothesis, which may become a statistical fact rather than a probability. We cannot predict the future, so likelihood calculation should depend on the weakness/flaw/bug existing in the system. Every moment, there can be an innumerable number of attempts from hackers to break a system, but most of those attempts go unnoticed by the security professional until the actual breach takes place. In other words, only the successful hacking attempts come to light, and success of an attack has a close relation with the vulnerability level of the system. When we claim that the likelihood of an attack is close to zero, it sends a strong message about the vulnerabilities mitigated by the current controls. Vulnerability level not necessarily increases the relative risk factor of the system, but the level of vulnerable sources plays a significant role in calculating risk. Damage can be of any form in a cloud environment: for example, data loss, denial-of-service, data breach or network security issues. When an existing vulnerability is successfully exploited by the threat agent, the damage caused is learned in the form of consequences. In other words, when nothing goes wrong and the likelihood of a successful attack is very low, its consequence has a zero impact on the system. Hence, we have to prioritize and address the threat which has very high risk in the first place and the one with least risk to the system may be focused at a later point of time. In our proposed metric, we insist this logic by following a unique method using risk assessment to rank the threats and provide a way to choose a security pattern from a pattern classification.

2. RELATED RESEARCH

2.1 Literature review on relation between threat and risk

Software vulnerability is specifically tied to the software component of the system, DaCosta et al. [6] defines it as, *"a fault in the specification, implantation, or configuration of a software system whose execution can violate an explicit or implicit security policy"*. In a Software Development Life Cycle (SDLC) [8], a vulnerability may emerge at any stage of development. A software professional never develops a program with an intention to include vulnerabilities in the coding or implementation phase. In other words, software vulnerabilities are not intentionally incorporated in the system, it was never meant to accommodate a malicious input, but it was designed to function normally when some expected inputs are received. A secure-coding process always includes the input validation process to handle unexpected inputs. A naive software development and the implementation process concentrates on the proper functioning of the system, whereas a secure software development and implementation process includes all possible input and output scenarios. Software systems should be developed with security being an integral part of SDLC, but it is a complex process. Many researchers have challenged Kaplan and Garrick's definition of risk and emphasized the need to include some other relevant factors. Aven [3] claims defining risk based on subjective probabilities is a narrowed approach and important uncertainty factors can be truncated or overlooked. The author comprehends the definition of risk based on other researches [15, 11, 22] as Risk = (A,C,P), where *'A'* represents the events, *'P'* stands for probabilities or likelihood, and *'C'* represents the consequences. Aven proposes the definition as, Risk = $(A,\ C,\ Pf^*,\ U,\ K)$, where Pf^* represents the relative probability, *'U'* - the uncertainty factor and *'K'* indicates the background knowledge about the estimate and description. We use Kaplan's definition of risk while calculating the risk in cloud environment by considering the attack event, likelihood of exploitation and its consequences. In our proposed metric for ranking threats, we calculated risk factor using DREAD [18] scale, that covers Aven's definition of risk by various parameters.

Zhang describes the role of vulnerability to open the link between events and consequences while defining risk [21]. Zhang states "a system's vulnerability represents the extent or the capacity of the system to respond to or cope with a risk event" [21]. From that standpoint, Kaplan's definition of risk is merely based on a subjective probability and the consequences and system's vulnerability is not included.. Zhang explains how a vulnerability source mediates between events or scenarios and its consequences. A vulnerability source decides whether we have to think about the likelihood or consequences of an attack. Hence, in our proposed model, we included risk as a major decisive factor in the securing process. Farahmand et al. [7] argues that estimating subjective probabilities for human attacks or human threat agents can be complex and sometimes those kinds of attacks are almost purely random. As a result, authors identify that risk should be described based on vulnerability of assets and motivation behind the attacks. Moreover, the motivation for every attack may differ and that is not an important factor to define risk. Regardless of the motives behind the attack, if a system can be successfully attacked, is facing the risk of security compromise. In our approach, we focus on current vulnerabilities existing in the system. Bannerman [4] points out that estimating probability of attacks in software systems can be difficult. The author claims that ignoring the influence of organization-specific vulnerabilities while defining risk as a significant flaw in its definition. In our approach to rank the serious threats of the system under review, we estimate the risk by including organization-specific vulnerabilities. Bannerman emphasizes the importance of vulnerability in any risk definition by stating, *"Vulnerability may increase or decrease an organization's exposure to risk event"* [4]. Aven proposed a framework for defining risk by replacing subjective probabilities as a measure of uncertainty and highlights the need to fix vulnerabilities in the system [3].

2.2 Literature review on security pattern selection based on threat assessment

Devanbu et al. [5] explains the importance to address security requirements during the requirement-engineering stage, but most of the security requirements become evident only after the functional requirements of the software have been completed. Authors argue that system requirements and design are done first and security protocols are added at the end. In most cases, we do not know the exact security requirements until the entire software development process is finished. Therefore, building security right from the development stage is not possible in all cases. So, when a security breach takes place, we have to choose "local fixes" as the next viable solution. Our approach provides a way to rank the security requirements based on the threat intensity, and helps the software developers to identify potential security

patterns to fix the vulnerabilities. Fernandez [9] claims that patches are not the best solution to fix the security vulnerabilities in software because patches on their own may open possibilities for new attacks. The author proposes two different solutions to improve the security of software design, the first method is to examine the final production code and search for possible problems. The second method is to plan for security from the beginning of the software development life cycle. Fernandez [9] chose to support that the security of a system can be best addressed by implementing security principles from the beginning of the whole life cycle, but that approach is not possible in all cases since some of the security requirements may come to light after developing the software. Patches are wide used to fix a vulnerability or implement a security protocol in a developed software. Basically, local fixes or patches are not the architectural solution to the system. When a patch is not implemented in an architectural mode it may introduce misconfigured or redundant patterns in the system.

3. THREAT RANKING COMPUTATION

All vulnerabilities in a software system or cloud may not be necessarily pose a threat or a serious risk to the system. For example, a patient's health record stored in cloud is a sensitive information, where an unauthorized access or an exposure of record is a serious security breach. In banking sectors, service availability may be considered as the top most requirement and other kind of security breaches are still important, but may not be a top priority. In other words, not all threats or vulnerabilities could occupy the top priority list in a system. Moreover, some threat in a software system may require a great amount of resource, where the cost to fix it would be much higher than the loss incurred by the exploit of that vulnerability. Similar to the Heart Bleed bug case [14], fixing the vulnerability costs more than the loss incurred by the exploitation of vulnerability. Many organizations would prefer to invest less money into to security strategies where possible. The risk assessment team makes prioritization of vulnerabilities from the organization's economic viewpoint. Basically, while listing the vulnerabilities all the following factors should be considered: 1) if a particular vulnerability is exploited how much cost is required to replace the asset, 2) how much cost is required to retrieve the data back and maintain, 3) how much cost should be invested to protect the asset, 4) would exploiting that vulnerability would affect the company's profit or revenue, and 5) how much should be offered by clients or users to prevent the exploit of a vulnerability. Clients rely on vendors for the security aspects of the software and vendors or service providers should be responsible to improve the security aspects of the system.

3.1 Impact of client perspective in risk assessment

While relative risk is assessed in the risk identification process, vendors should be aware that a particular asset may be valuable to the company but it may be invaluable to clients. Any successful attack on the software system would directly affect the customers, but it would also expose the company to liability and public embarrassment. Eventually the company lose its credibility in the market. A weak authorization policy designed by the software professionals becomes highly vulnerable to password-guessing attacks. Hence, there is a huge responsibility placed upon the user to set a strong password. Users or clients should be aware of legal rights to claim compensation upon security breaches before signing up for a new service. In our proposed model, we start our evaluation by assessing clients' requirements.

3.2 Threat modeling based on STRIDE-DREAD Theory

Information Security (InfoSec), as defined by the standards published by the "Committee on National Security Systems (CNSS), formerly the National Security Telecommunications and Information Systems Security Committee (NSTISSC) [19], is the protection of information and its critical elements, including the systems and hardware that use, store, and transmit that information". The NSTISSC model of information security evolved from a concept developed by computer security known as the CIA triangle [20] that represents three basic key aspects of information: Confidentiality, Integrity and Availability. While confidentiality, integrity and availability being the basic aspects of information protection, it is people, process and technology that decide how this protection actually takes place [2]. An attack is an exploitation of a vulnerability to realize a threat. Therefore, threat modeling for cloud environments may help identify potential threat events, exploitation of cloud assets from each threat perspective. Our proposed STRIDE-DREAD model usage for cloud environments would also help us analyze the associated risks and rank the threats based on their intensity. As stated by Howard and Le Blanc [18], *"You cannot build a secure system until you understand your threats."* Similarly, for a cloud security, a developer or CSP has to identify possible threats and understand how attacks are performed on a cloud environment.

In our proposed metric we use Microsoft's STRIDE model, which is a standard threat modelling approach. The STRIDE acronym is formed from the first letter of each of the following threat, they are Spoofing, Tampering, Repudiation, Information Disclosure, Denial-Of-Service, and Elevation of Privilege. We define each threat category in Table 1, and directly relate our definitions with cloud attack scenarios. While STRIDE is a well-tested framework for traditional software systems, we use this model to gauge a cloud user's preference or expectations on cloud security. In our proposed metric for quantifying cloud security needs, the STRIDE threat model is gauged by client organization, by inquiring their security requirements from the cloud vendor/developer. Not all client organization may have same priority or preference on security, it differs for clients based on their domain, type of service availed from clouds, or on the nature of data/information stored. So, software architects from client organization assign a value for each threat using the metric proposed in Table 2 based on their requirement on private clouds or hybrid cloud.

3.3 Proposed Threat Assessment Ranking Matrix

For each given threat factor in STRIDE model, clients assign a value, and it can be either one of the following Critical, High, Medium, Low or None. These values are assigned within the range of 0 to 1, and all five categories of

Table 1: Correlation between STRIDE model and cloud environment

Threat Categories	Cloud Security Perspectives
Spoofing: an attacker poses as an authorized user using an identity	An attacker using another user's authentication information (username, password) and access the data in cloud.
Tampering: modifying data with malicious intention	Without user's knowledge or permission, an attacker modifying the data on cloud or over a complete network.
Repudiation: ability to filter some malicious action when enough proof is missing	In a multitenant environment like cloud, when an authorized user performs some illegal operation and if the system lacks the ability to trace that prohibited action, then it becomes an action without other parties having any way to prove otherwise.
Information Disclosure: exposure of information to any individual who are not supposed to have access to it	A cloud user (or a malicious insider) accessing co-tenant's workflow without any authorization to do it.
Denial of Service: denying service to valid users	A cloud user (or an attacker) gains control of a co-tenant's virtual machine and make their web server unavailable or unusable.
Elevation of Privilege: unprivileged user gains privileged access thereby compromise or destroy the entire system	A cloud user (or an attacker) gains access to all system defenses to projects as a trusted system itself.

Table 2: Proposed weightage for threat priorities - client perspective

Nature of Threat	Weightage	Definition
Critical	1.0	Highest - degree of concern and given first priority
High	0.75	High - degree of concern
Medium	0.5	Medium degree of concern
Low	0.25	Lowest degree of concern
None	0.0	No concern for the given threat

threats are equally spaced in this interval. This value indicates user's priority for any particular threat and also depending upon their requirements. Generally, clients should be someone from the organization with a minimum knowledge about various software attacks, and software security requirements. Client may give same level of priority for one or more threats in the system. In the matrix, $U_{S,M}$, represents client's degree of concern / priority for Spoofing attack, where *'S'* denotes spoofing, and *'M'* denotes degree of concern from user's perspective. Similarly, five more user perspective values like $U_{T,M}$, $U_{R,M}$, $U_{I,M}$, $U_{D,M}$, $U_{E,M}$, will be calculated and it represents tampering, repudiation, information disclosure, denial-of-service, elevation of privilege threats respectively. For an easy understanding, we use the first alphabet of threat category in STRIDE model to represent client preference / priority for that particular threat category. Once client requirements are recorded, we calculate the risk for an attack to take place by exploiting a particular threat. We use the DREAD model to perform this evaluation (see Table 3).

Using the DREAD model, risk level of software can be calculated for the following five different perspective, Damage potential, Reproducibility, Exploitability, Affected Users, and Discoverability. We assign three different risk levels in our methodology: 0, 5, and 10. For example, damage potential caused by exploiting a spoofing threat in the system is represented as $R_{D,W}$, where *'D'* represents damage potential and *'W'* represents the probability of that risk to take place based on the current state of the software. $R_{D,W}$, can be assigned any one of the values (0, 5, or 10) based on the risk factor, consequences, and vulnerability. We apply the theoretical perspectives explained in Section I of this paper as a guiding lens to validate our threat assessment. Following our theoretical perspectives, we include the source of threats, risk potential and also user perspectives. For example, $R_{DI,W}$ represents discoverability of any existence of a threat in the system, and that threat can be any one of the following: Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, or Elevation of privilege. If a particular vulnerability is hard or impossible to discover by anyone then probability of exploiting that threat would be very low, hence it is considered a low or zero risk. Therefore, this would be given a value 0. If the threat can be identified by guessing or tracking the networks then it is considered a medium level risk, and gets a risk factor of 5. If the existence of a threat can be easily or obviously identified then it is under high risk and gets a risk factor of 10. After assigning risk factors, the product of threat prioritization and its underlying risk level is calculated. For example, User preference on spoofing threat $U_{S,M}$ is multiplied with $R_{DI,W}$ which is the risk of discovering the existence of spoofing threat in the system, to get a combined score, $P_{S,DI}$. (see Table 4).

This process is repeated for each threat in STRIDE model with five different perspectives from DREAD model. This combined product plays a significant role to make the final ranking of serious threat to be addressed. For example, if

Table 3: Relative risk assessment scaling factors [10]

Risk Factor	0	5	10
Damage Potential (If a threat occurs, how much damage is caused?)	Nothing	Individual user data in cloud is affected	Complete system or all users get affected
Reproducibility (How easy is it to reproduce the data if threat is exploited?)	Very hard or impossible, even for the administrators or cloud vendors	One or two steps from an authenticated user	Very hard or impossible, even for the administrators or cloud vendors
Exploitability (What is needed to exploit this threat?)	Advanced programming and cloud networking knowledge or an elite hacker	Malware existing in Internet or using attack tools	Simple attack tools or even a web browser
Affected Users (How many users will be affected?)	None	Some, but not all	All users
Discoverability (How easy to discover that this threat existing in the cloud?)	Very hard to impossible	Can identify it by guessing or monitoring network traces	Obviously visible or very easy to identify

Table 4: Proposed Threat Assessment Ranking Matrix

Risk Factor	$\mathbf{R}_{D,W}$	$\mathbf{R}_{R,W}$	$\mathbf{R}_{E,W}$	$\mathbf{R}_{A,W}$	$\mathbf{R}_{DI,W}$	Cumulative Index	Final Rank
$U_{S,M}$	$P_{S,D}$	$P_{S,R}$	$P_{S,E}$	$P_{S,A}$	$P_{S,DI}$	$\sum P_{S,D} + P_{S,R} + ... + P_{S,DI}$	
$U_{T,M}$	$P_{T,D}$	$P_{T,R}$	$P_{T,E}$	$P_{T,A}$	$P_{T,DI}$	$\sum P_{T,D} + P_{T,R} + ... + P_{T,DI}$	
$U_{R,M}$	$P_{R,D}$	$P_{R,R}$	$P_{R,E}$	$P_{R,A}$	$P_{R,DI}$	$\sum P_{R,D} + P_{R,R} + ... + P_{R,DI}$	
$U_{I,M}$	$P_{I,D}$	$P_{I,R}$	$P_{I,E}$	$P_{I,A}$	$P_{I,DI}$	$\sum P_{I,D} + P_{I,R} + ... + P_{I,DI}$	
$U_{D,M}$	$P_{D,D}$	$P_{D,R}$	$P_{D,E}$	$P_{D,A}$	$P_{D,DI}$	$\sum P_{D,D} + P_{D,R} + ... + P_{D,DI}$	
$U_{E,M}$	$P_{E,D}$	$P_{E,R}$	$P_{E,E}$	$P_{E,A}$	$P_{E,DI}$	$\sum P_{E,D} + P_{E,R} + ... + P_{E,DI}$	

clients consider one particular threat to be a serious or a critical threat, and during risk analyzation process, if the discoverability of existence of that threat is assigned as zero then it is not considered a serious threat at all. This calculation would provide some room for a serious threat to take the first place in the ranking process. To get the perspective and severity of one particular threat based on user requirement and current risk level of exploiting the vulnerability, we get the cumulative index of a threat. Cumulative index for spoofing is calculated as $\sum P_{S,D} + P_{S,R} + P_{S,E} + P_{S,A} + P_{S,DI}$. This is repeated for all other threats and we finally make a ranking on threats with serious threats holding a high cumulative index and severity of other threats will be represented in descending order in the final rank column. As a guiding tool to implement the required security features and security patterns, we provide the security pattern classification proposed by Hafiz et. al. [17] in Figure 2.

4. CASE STUDY

We performed a case study of our proposed methodology using XEN hypervisor cloud environment. First, we start from client/user's perspective about cloud security requirements based on own requirements. Each threat in STRIDE model will be assigned an impact level by software architects or a security expert from client organization. For an explanatory purpose, we assign following impact levels for XEN cloud environment. For this case study, we designed our own client requirements by placing a high emphasize for Spoofing attacks, and zero tolerance for gaining privileges. We place a high concern on authentication protocols and access control, as we expect this cloud environment to provide very high resistance on authentication and other human interaction attacks. Apart from Spoofing and Elevation of privilege threats, all other threats are important but not at a high priority. Our metrics are assigned as follows for XEN cloud environment: Spoofing - 1.0 (Critical), Tampering - 0.75 (High), Repudiation - 0.25 (low), Information disclosure - 0.5 (Medium), Denial-Of-Service - 0.5 (medium), and Elevation of privilege - 1.0 (Critical). These factors depicts the security requirements from our end, and software developers or security professionals are not expected to proceed with that order, but assign related risk factor for each threat by analyzing the underlying software and implemented security protocols. So, software developers or security experts will be assigning relative risk factors for each threat in the STRIDE model based on five different categories of the DREAD model. For example, in this case, spoofing was considered as a critical threat. Now, security experts analyze the likelihood of successful spoofing attacks and its consequences using parameters from Table 6. Even though, spoofing attack is considered as a critical issue by clients, it may not be a critical one to be addressed and it will be decided by security experts by looking into the software and already implemented security protocols. For this case study, we enumerated 62 different vulnerabilities identified in the XEN cloud environment [1]. CVE (Common Vulnerabilities and Exposure) details also provide CVSS (Common Vulnerability Scoring System) score on vulnerability sever-

Table 5: Resultant Matrix - Threat Assessment on XEN Cloud Vulnerabilities

Risk Factor	**Client Rank**	**R** $_{D,W}$	**R** $_{R,W}$	**R** $_{E,W}$	**R** $_{A,W}$	**R** $_{DI,W}$	**Cumulative Index**	**Final Rank**
$U_{S,M}(0.75)$ (Spoofing)	3	P $_{S,D}(5) = 3.75$	P $_{S,R}(5) = 3.75$	P $_{S,E}(10) = 7.5$	P $_{S,A}(10) = 7.5$	P $_{S,DI}(0) = 0$	$\sum$ P $_{S,D}$+ ... + $_{S,DI} = 22.5$	2
$U_{T,M}(1.0)$ (Tampering)	1	P $_{T,D}(10) = 10$	P $_{T,R}(0) = 0$	P $_{T,E}(0) = 0$	P $_{T,A}(5) = 5$	P $_{T,DI}(0) = 0$	$\sum$ P $_{T,D}$+ ... + $_{T,DI} = 15$	4
$U_{R,M}(0.25)$ (Repudiation)	6	P $_{R,D}(10) = 2.5$	P $_{R,R}(0) = 0$	P $_{R,E}(5) = 1.25$	P $_{R,A}(10) = 2.5$	P $_{R,DI}(5) = 1.25$	$\sum$ P $_{R,D}$+ ... + $_{R,DI} = 7.5$	6
$U_{I,M}(0.5)$ (Information Disclosure)	4	P $_{I,D}(5) = 2.5$	P $_{I,R}(5) = 2.5$	P $_{I,E}(0) = 0$	P $_{I,A}(0) = 0$	P $_{I,DI}(10) = 5$	$\sum$ P $_{I,D}$+ ... + $_{I,DI} = 10$	5
$U_{D,M}(0.5)$ (DoS)	5	P $_{D,D}(10) = 5$	P $_{D,R}(10) = 5$	P $_{D,E}(10) = 5$	P $_{D,A}(10) = 5$	P $_{D,DI}(10) = 5$	$\sum$ P $_{D,D}$+ ... + $_{D,DI} = 25$	1
$U_{E,M}(1.0)$ (Elevation of Privilege)	1	P $_{E,D}(5) = 5$	P $_{E,R}(5) = 5$	P $_{E,E}(0) = 0$	P $_{E,A}(0) = 0$	P $_{E,DI}(10) = 10$	$\sum$ P $_{E,D}$+ ... + $_{E,DI} = 20$	3

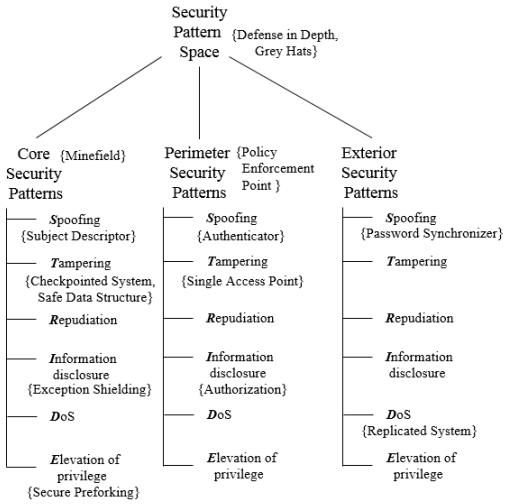

Figure 2: STRIDE based Security Pattern Classification [17]

ity. For example, if the discoverability of DoS vulnerability in the software is close to impossible, then it is considered as a very low risk, hence it may not get a high priority and fixed immediately. Security experts cannot come to that conclusion considering one particular perspective, but inspecting the system from different perspectives like damage potential, reproducibility, exploitability, number of users being affected and discoverability of that particular threat in the system. When user considers a threat to be critical, and if security experts believe that it possess very low risk, then the threat in the next severity level will be focused.

In this case study, first column based out of STRIDE model is assigned by clients, and remaining 5 columns based out of DREAD model are evaluated using CVE list [1] and CVSS [1] score for XEN cloud environment. To identify the ranking of threats faced by the system we multiply relative risk factors with user assigned metrics, and the result is displayed in each cell of the first row. The sum of all products (critical factor * relative risk) is displayed as cumulative index. In this case, Tampering was assigned as a highest threat by client, its relative risk was analyzed from five different perspectives by cloud developers, and its final added value is calculated as 15. We repeat this process for remaining threats using vulnerability listings from CVE [1] and calculate the final resultant for each threat. We represented our updated matrix in Table 5. By applying our proposed threat priority analyzing matrix, multiple parties like clients, cloud developers, and security professionals get involved while making a final decision about ranking on serious threats to be addressed. In this methodology, the client's preference is not blindly followed, but their input is included in the quantification process and given a significant weighting. Similarly, inputs from CVE vulnerability listing on XEN cloud environment is calculated following the DREAD model, and risk factor is cross multiplied with user input to get a cumulative index. From the example explained above, though client ranked the threats in one particular order, final analysis shows the threats to be addressed in a different order.

Once user's ranking is verified with XEN cloud environment, security experts and software developers can get an idea whether the system already satisfies user's requirements to the expected level or lagging in certain aspects. Once we identify the ranking of threats to be addressed, we related it to the STRIDE-based security pattern classification proposed by Hafiz et. al [17], and it is represented in Figure 2. Though we are aware of existing security pattern classifications, there is a need for a guiding tool that could lead us to select patterns from a particular category. Through our proposed metric, we have provided a way to quantify the security requirements of a cloud environment, and our

quantification would provide the solution by linking it to an existing pattern classification scheme.

5. FUTURE DIRECTIONS

In this paper, we presented a security metric to quantify the ranking of threats currently existing in the cloud system based on their severity. Our primary motive is to provide a catalog of security patterns that could help cloud developers to identify the security pattern to be applied in the system. We linked an already existing classification scheme as a potential solution, and we have reserved our idea on classification of security patterns for cloud environment based on threat model as our future work.

6. CONCLUSIONS

In this paper, we propose a quantitative methodology to rank the threats in a cloud environment. Cloud environment generally comprises many stakeholders like clients, cloud developers, vendors, and security experts. Through this ranking system, we insisted that before implementing any security protocol, inputs from all stakeholders should be considered. Security requirements of every client or organization may differ, and it becomes a challenge to cloud developers or security experts to include clients while making a decision. In this paper, we proposed a metric system that starts from the client's input and eventually becomes the base model for the threat-ranking process. Our metric also shows that if threat ranking was merely done by software developers or security experts, it would have resulted in a different ranking scale, and that may not be a required service from client's perspective. On the other hand, we didn't give any over-weighting for client's requirement, because those requirements would have been an implemented security protocol in the system. Cloud developers makes the decision from the technical and security viewpoint, and through this metric we find the actual requirement of the system. This metric would guide the cloud developers to identify the top priority threat that has to be addressed first and also links to an existing security pattern catalog.

7. ACKNOWLEDGEMENT

The Authors would like to thank William Aiken for his reviews and helpful comments. This material is based upon work supported by the National Science Foundation under Grant No. (1514568). This work was also supported in part by the National Research Foundation of Korea (No. 2014R1A1A1003707), the ITRC (IITP-2015-H8501-15-1008), and the NIPA (NIPA-2014-H0301-14-1010). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the aforementioned agencies.

8. REFERENCES

[1] Cve details. http://www.cvedetails.com/vulnerability-list/vendor_id-6276/XEN.html.

[2] B. Arief and D. Besnard. *Technical and human issues in computer-based systems security.* Technical Report Series-University of Newcastle upon Tyne Computing Science, 2003.

[3] T. Aven. A unified framework for risk and vulnerability analysis covering both safety and security. *Reliability Engineering & System Safety*, 2007.

[4] P. L. Bannerman. Risk and risk management in software projects: A reassessment. *Journal of Systems and Software*, 2008.

[5] P. T. Devanbu and S. Stubblebine. Software engineering for security: a roadmap. In *Proceedings of the Conference on the Future of Software Engineering. ACM*, 2000.

[6] D. D. et al. Characterizing the 'security vulnerability likelihood' of software functions. In *Proceedings International Conference on Software Maintenance.* IEEE, 2003.

[7] F. F. et al. Managing vulnerabilities of information systems to security incidents. In *Proceeding of the 5th international conference on Electronic commerce, ACM*, 2003.

[8] P. C. et al. *Documenting Software Architectures: Views and Beyond.* Pearson Education, 2002.

[9] E. B. Fernandez. A methodology for secure software design. In *Proceedings of the International Conference on Software Engineering Research and Practice*, 2004.

[10] M. Howard and D. LeBlanc. *Writing Secure Code.* Microsoft Press, 2002.

[11] S. Kaplan. *Risk assessment and risk management-basic concepts and terminology.* Risk Management: Expanding Horizons In Nuclear Power and Other Indurstries, 1991.

[12] S. Kaplan. *The words of risk analysis.* Risk analysis, 1997.

[13] S. Kaplan and B. J. Garrick. *On the quantitative definition of risk.* Risk analysis, 1981.

[14] I. Lioupras and E. Manthou. *Don't let my Heart bleed!: An event study methodology in Heartbleed vulnerability case.* Informatik Student Paper Master (INFSPM), 2014.

[15] W. W. Lowrance. *Of Acceptable Risk: Science and the Determination of Safety.* William Kaufmann, 1976.

[16] G. McGraw. *Software Security.* IEEE Security & Privacy, 2004.

[17] P. A. Munawar Hafiz and R. E. Johnson. Organizing security patterns. *Software, IEEE*, 2007.

[18] M. D. Ryan. Cloud computing security: The scientific challenge, and a survey of solutions. In *Journal of Systems and Software*, 2013.

[19] N. S. Telecommunications and I. S. Security. National training standars for information systems security (infosec) professionals. www.cnss.gov/Assets/pdf/nstissi_4011.pdf, June 1994.

[20] M. Whitman and H. Mattord. *Principles of information security.* Cengage Learning, 2011.

[21] H. Zhang. A redefinition of the project risk process: Using vulnerability to open up the event-consequence link. *International Journal of Project Management*, 2007.

[22] E. Zio. *An introduction to the basics of reliability and risk analysis.* World scientific, 2007.