NETWORK RISK EVALUATION FROM VULNERABILITY DETECTION TOOLS

Aniwat Hemanidhi[1,*], Sanon Chimmanee[2], Parinya Sanguansat[3]

[1, 2] Faculty of Information Technology, Rangsit University, Pathumthani, Thailand
[3] Faculty of Engineering and Technology, Panyapiwat Institute of Management, Nonthaburi, Thailand
*e-mail: ahemanidhi@hotmail.com

Abstract

Security is always main concern in every organization networks. Security audit becomes very important procedure to ensure that the organization network is secured. Unfortunately, it requires human expertise, from both auditors and analysts, with extreme time consumption. Therefore, some form of automatic detection and analysis is preferred. Ordinarily, there are two types of vulnerability detection tools: hardware appliances and software applications. Network risks can basically be divided into 4 levels. However, they are tightly to be classified by different standard. This paper presents a simple criterion for evaluating the network risk from various vulnerability detection tools called as “Network Risk Metric”. In this experiment, NetClarity (hardware-based) and Nessus (software-based) are implemented on two network zones of Rangsit University (RSU): Internal Zone and De-Militarized Zone (DMZ). Then, the results from both test tools on each target network are applied with the proposed metric. As a result, the network risk has been fairly evaluated from our independent platform not bias by any government organizations or relative vendors.

Keywords: risk evaluation, risk assessment, vulnerability detection, security metrics, network security

Introduction

Network threat is an important aspect related to computer and information security. There are many types of threats that force security professionals to find the best solution for protecting networks. These threats could be made directly by physical attack, logically performed via data communication network, dialogues attack and social engineering. To diminish possibility of attacks, vendors will basically offer patches to fix their product vulnerabilities. The organization insiders, who use IT resources in the wrong manner, or IT staffs, who lack to audit communication logs, might also lead organization network into dangerous condition as well. Misconfiguration of IT equipment, for example, firewalls and IDS/IPS, could raise IT security into critical troubles, as many vulnerabilities are available for hackers to attack. The faster network vulnerability detection and analysis, the better system protection and attack countermeasure. The plan-protect-response (PPR) is a highly time-consuming procedure. Thus, IT professionals prefer automated security test tools to detect vulnerabilities, analyse the results and provide them reasonable advisories. This paper will focus on the main topics related to evaluate risk from network vulnerability detection. Two network zones, internal and DMZ, of Rangsit University are investigated using two vulnerability detection tools: NetClarity and Nessus. The first one is a well-known hardware appliance and the latter is famous open-source software. The list of information security vulnerabilities and exposures called “Common Vulnerabilities and Exposures (CVE)” is mentioned as a link to a standard vulnerability databases named “The U.S. National

Vulnerability Database (NVD)". An open framework that addresses vulnerabilities across many disparate hardware and software platforms called "Common Vulnerability Scoring System (CVSS)" is used to compare as a vulnerability scoring systems. A security metric is referred as security levels, security performance, security indicators or security strength (Tito Waluyo et al. 2011). A novel Security Metric Objective Segments (SMOS) model that is a high-level security metric objective taxonomization novel for software-intensive system has been introduced (Reijo M. Savola et al. 2009). This model is the most common classification, which is visualized in a nested circle presentation. The security metrics can be categorized various ways. One simple classification is to consider metrics that denote the maturity level of processes believed to contribute to the security of a system, versus those that denote the extent to which some security characteristic is present in a system. Security measurement and metrics efforts that are conceived at a high level of abstraction and formalism are often difficult to interpret and apply in practice. There are many open problems in the security metrics area including application security, network security, software security, etc., (Wayne Jansen 2009). A risk analysis process that intertwines security requirements engineering with a vulnerability-centric and qualitative risk analysis method are introduced. CVSS becomes an open framework developed by National Institute of Standards and Technology for assessing vulnerabilities criticality across many disparate hardware and software platforms. (Golnaz elahi et al. 2011). This method adapts the CVSS for evaluating risks of vulnerabilities within *i* models of system requirements and stakeholders' goals. However, the proposed metric evaluations may be inaccurate and unreliable, and metrics aggregating vulnerability scores into one value. To deal with multiple standards of vulnerability detection scoring model, we propose simple criterion called "Network Risk Metric" with general equation to evaluate network risk from different vulnerability test tools in percentage. In information security for web-based applications, the detection and diagnosis of software vulnerabilities are important tasks for the user (Peter Kok Keong Loh et al. 2010). However, there are often significant differences in the content, organization, and format of different vulnerability reports (Peter Kok Keong Loh et al. 2010; DHS/NIST]. In the detection and diagnoses of different web application vulnerabilities, it should be noted that some vulnerabilities are more dangerous than others in term of potential damage/risks (D. Subramanian et al. 2009). Thus, there is needed for appropriate metrics to grade the various vulnerabilities as well as the different scanners. A fuzzy classification metrics that are used to grade web application scanners and vulnerability has been proposed (Peter Kok Keong et al. 2010).

The rest of this paper is organized as follows. Details of related elements and proposed metrics are explained in the next methodology section. In experimental result section, risk evaluation from proposed metrics will be explained with some example. In the last section, conclusion as seen from the experimental result will be discussed. Future work will also describe here as well.

Methodology

Network Metrology

There are many types of data communication networks in RSU. However, it can be classified into three main zones including internet zone, intranet zone and DMZ. Each zone will be separately protected and filtered by firewall (Figure 1). In this paper, only intranet zone and DMZ are chosen to be tested by two vulnerability detection tools: NetClarity Auditor (Version 8.1.3) and open-source software named "Nessus" HomeFeed (Version 5.0.1). To guarantee the reliability of experimental result, both test tools are connected to the same network simultaneously. NetClarity is the hardware appliances so it is comfortably interfaced via RJ-45 Ethernet socket of the core switch. For Nessus, it is pre-installed on a computer laptop (Intel® Core™ 2 DUO CPU P8700 2.53GHz, Windows 7 64 bits) connecting to the same core switch as well. Some specific details for deep investigation are available to be adjusted such as ports, protocols and services. However, to make it fare, default configurations of both test tools are selected. IP Addresses of both test tools must be set within the same subnet of the target network. In summary, overall procedures are: 1) search all active hosts, 2) scan and detect vulnerabilities and 3) report the result (including analysis and recommendation).

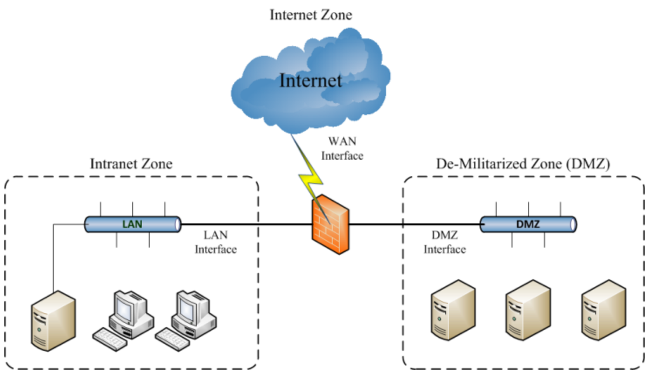

Figure 1 Network Diagram of Rangsit University (RSU)

Risk Level

NetClarity classifies its risk level from 2 lists of information security vulnerabilities and exposures “CVE” and a full disclosure moderated mailing list for the detailed discussion and announcement of computer security vulnerabilities called “Bugtraq ID (BID)”. The CVE list is mentioned as a link to NVD which is maintained by the MITRE Corporation and publicly available to anyone interested in correlating data between different vulnerability or security tools, repositories and services. NetClarity classifies risks into 4 levels (Table 1).

Table 1 Risk level of each vulnerability types classifies by NetClarity

Risk Level	**Vulnerability Type**
Low	Less important vulnerability - harder to exploit and usually causes little or no damage to the network assets.
Medium	Slightly more important than a Low-level vulnerability but usually hard to exploit. Medium level vulnerabilities might allow an attacker to gain access to the network.
High	Very important vulnerability that may be easy to exploit and allow an attacker to cause serious damage to the network.
Serious/Critical	Extremely important vulnerability that may be easy to exploit and allow an attacker to cause critical damage to the network.

Nessus also classify vulnerability into 4 levels: Low, Medium, High and Critical. It uses many references to determine risk level such as CVE, CVSS, BID, CERT Advisory ID, Open Source Vulnerability Database (OSVDB), Exploit Database ID (EBD-ID) and IAVA. However, the most important severity ratings are derived from an industry open standard designed to convey vulnerability severity and help determine urgency and priority of response called “The Common Vulnerability Scoring System (CVSS)”, where 0 is “Info”, less than 4 is “Low”, less than 7 is “Medium”, less than 10 is “High” and a CVSS score of 10 will be flagged “Critical” or “Serious”. This standard has jointed effort involving many groups including: CERT/CC, Cisco and Microsoft. Deep details of these CVSS metrics could be found at http://www.first.org

Host searching and time consumption

When configuration is done, both test tools start to scan the set network. After a period of time, each of them will report number of found active hosts. From experimental results, it shows that NetClarity has slightly better performance in searching active host. However, it is much slower than Nessus. Both factors impact on performance of vulnerability test tools. Nevertheless, we desired to take an effect form number of found active host merely. This paper will concentrate to evaluate risk form number of found vulnerabilities on active hosts rather than performance of time response to detected attacks. The basis assumption is that detected vulnerabilities are not attacked yet. Time consumption from host searching will play its role in the next future work.

Vulnerability Detection

The most important factor affected to this research is the ability of vulnerability detection. Although some detected vulnerabilities are in the same condition, e.g. same port or same protocol, NetClarity and Nessus might classifies them into different risk level depended on what standard they desired to apply for that subject. This deviation has significantly influenced on the accomplishment of vulnerability detection.

Proposed Risk Evaluation

To cope with various standard of vulnerability classification, this paper proposed simple criterion for evaluating the network risk from several vulnerability detection called as “Network Risk Metric”. Two major ideas, “Weighted Cut-off Severity Normalization” and “Probability of Trust”, will be introduced in each phase of the proposed risk evaluation metric. The overall procedure is separated into two phases: 1) Differentiate server and client risk and 2) Risk evaluation.

Phase1: Differentiate server and client risk

There are 5 steps in this phase. The first step starts with searching active hosts and scanning detected vulnerability of each active host. Both NetClarity and Nessus have ability to specify MAC Address and operating system of the target but it is system administrator’s responsibility to separate which one is server or client. Let H be number of active hosts. Subscript s refers to “server” and c refers to “client”. Thus, H_s represents number of detected active server and H_c represents number of detected active client. The next step is to find the “Cut-off” for server and client group. From our experiment, the number of detected vulnerabilities on server is basically much higher than thus detected on client. Too distinct value of data may cause our result inaccurate since the distribution range of detected vulnerability is too wide. Therefore, we need the cut-off value f to limit diffusion of data. The server cut-off can derive from the following equation:

$$f_s = \frac{\sum_{i \in s} \sum_{l=1}^{L} n_{i,l}}{LH_s} \quad (1)$$

The client cut-off can also derive from the following equation:

$$f_c = \frac{\sum_{i \in c} \sum_{l=1}^{L} n_{i,l}}{LH_c} \quad (2)$$

Where L is number of risk level, $i \in \{Server(s), Client(s)\}$ and $n_{i,l}$ is number (value) of detected vulnerabilities in each risk level l. Note that, both NetClarity and Nessus has "4" risk levels, so, generally, L is 4 in this paper. The third step is to normalize all detected vulnerabilities of each risk level by appropriated cut-off. Then, the number (value) of detected vulnerabilities in each risk level can be limited. If $n_{i,l}$ is less or equal to its type of cut-off f then the original value is maintained. Otherwise, $n_{i,l}$ will be set to its cut-off value. Let $\bar{n}_{i,l}^{(s)}$ and $\bar{n}_{i,l}^{(c)}$ represent new value of vulnerability after comparing with the server's cut-off and client's cut-off orderly. The new "Cut-off Normalize Table" can then be created from Eq.(3) and (4) as follows:

$$\bar{n}_{i,l}^{(s)} = \begin{cases} ln_{i,l} & , n_{i,l} \le f_i \\ lf_s & , n_{i,l} \le f_i \end{cases} \text{, where } i \in s \quad (3)$$

And,

$$\bar{n}_{i,l}^{(c)} = \begin{cases} ln_{i,l} & , n_{i,l} \le f_i \\ lf_c & , n_{i,l} \le f_i \end{cases} \text{, where } i \in c \quad (4)$$

In the fourth step, $\bar{n}_{i,l}^{(s)}$ and $\bar{n}_{i,l}^{(c)}$ will be weighted by weighted value ω_L (Table 2) from the following equation:

$$\acute{n}_{i,l}^{(s)} = \omega_L \times \bar{n}_{i,l}^{(s)} \quad \text{, where } i \in s \quad (5)$$

And,

$$\acute{n}_{i,l}^{(c)} = \omega_L \times \bar{n}_{i,l}^{(c)} \quad \text{, where } i \in c \quad (6)$$

The outcomes from Eq. (5) and (6) are new weighted value of vulnerability for each server host $\acute{n}_{i,l}^{(s)}$ and client host $\acute{n}_{i,l}^{(c)}$. Hence, the "Weighted Normalize Table" is created.

Finally, normalize risk for server $R_i^{(s)}$ and client $R_i^{(c)}$ can be calculated as follows:

$$R_i^{(s)} = \frac{\sum_{l=1}^{L} \acute{n}_{i,l}}{\sum_{l=1}^{L} l \cdot f_i} \times 100 \text{ , where } i \in s \quad (7)$$

And,

$$R_i^{(c)} = \frac{\sum_{l=1}^{L} \acute{n}_{i,l}}{\sum_{l=1}^{L} l \cdot f_i} \times 100 \text{ , where } i \in \text{c} \quad (8)$$

Table 2 Weighted value corresponding to risk level

Risk Level (L)	Serious	High	Medium	Low
Weight (ω_L)	4	3	2	1

As a result, the relative risk for server $\bar{R}_s$ and client $\bar{R}_c$ can be estimated from Eq. (9) and (10) as follows:

$$\bar{R}_s = \frac{\sum_{i \in s} R_i^{(s)}}{H_c} \tag{9}$$

And,

$$\bar{R}_c = \frac{\sum_{i \in c} R_i^{(c)}}{H_c} \tag{10}$$

Phase 2: Risk Evaluation

There are two products from the previous phase, active hosts (H_s *and* H_c) and relative risk ($\bar{R}_s$ and $\bar{R}_c$), that will be used as initial values to evaluate the total estimated risk. There are only 3 steps in this phase. Remind that the number of active hosts detected from each detection tools may not equal. Thus, it is not fair if we just add them together and divide by number of test tools. Hence, the first step of this phase is to find the "Probability of Trust" which related to ratio of detected servers and clients. The simple equation for "Probability of trust", *P_i(T)*, is as follows:

$$P_i(T) = \frac{H_i}{\sum_{\forall i} H_i} \quad ; i \in \{s, c\} \tag{11}$$

The second step is to find the possibility of risk $P_i(R)$ that detected vulnerabilities might be exploited. This can be made by applying the "Probability of trust" to the relative risk of server and client group with the following equation:

$$P_i(R) = \bar{R}_i \times P_i(T) \quad ; i \in \{s, c\} \tag{12}$$

Finally, in the last step, the total estimated risk in percentage could be easily calculated by adding both possibilities of risk together and times it with 100. The equation is as follows:

$$Total\ estimated\ risk\ (\%) = (P_s(R) + P_c(R)) * 100 \tag{13}$$

Total estimated risk is our goal in this paper as it represents final value from proposed "Network Risk Metric".

Experimental Results

As specify in the beginning of this paper, two network zone of RSU are discovered. However, only DMZ will be explained as an example. The rest can follow the same procedure. From DMZ, result of host searching and vulnerability detection by NetClarity and Nessus are shown in Table 3. In phase 1, the total number of active host is 26. Eighteen servers are detected as shown with bold letter (Table 3). The rest is considered as client hosts although some of them might be other network equipment e.g. router or firewall. Thus, the number of active server H_s is 18 and active client H_c is 8. There are 246 detected vulnerabilities on servers (Table 3) so $n_{i,l}^{(s)}$ is 246. In the same way, there are 130 detected vulnerabilities on clients so $n_{i,l}^{(c)}$ is 130. As mention above, the number of risk level (L) is 4. Therefore, from Eq. (1) and (2), the cut-off value for server f_s and client f_c are 3.417 and 4.063 respectively. Now, we can create the "Cut-off Normalize Table" from the appropriate cut-off (Table 4). In the fourth step, each cut-off normalize vulnerability, $\bar{n}_{i,l}^{(s)}$ and $\bar{n}_{i,l}^{(c)}$, will be weighted. From Eq. (5) and (6), the "Weighted Normalize Table" is created (Table 4).

Table 3 Result of host searching and vulnerability scanning from the RSU De-Militarized Zone (DMZ) by NetClarity (left) and Nessus (right)

	NetClarity	Risk Level			
	IP Address	Serious	High	Medium	Low
1	**xxx.yyy.184.2**	0	3	3	19
2	**xxx.yyy.184.9**	0	0	1	13
3	**xxx.yyy.184.11**	0	2	4	16
4	**xxx.yyy.184.22**	0	7	3	7
5	**xxx.yyy.184.28**	0	8	2	9
6	**xxx.yyy.184.51**	0	1	0	0
7	**xxx.yyy.184.52**	0	1	0	0
8	**xxx.yyy.184.54**	1	9	3	18
9	**xxx.yyy.184.55**	0	1	1	1
10	**xxx.yyy.184.56**	0	1	1	0
11	**xxx.yyy.184.57**	1	3	3	7
12	**xxx.yyy.184.58**	0	4	3	12
13	**xxx.yyy.184.59**	0	4	3	12
14	**xxx.yyy.184.60**	0	1	1	0
15	**xxx.yyy.184.61**	0	1	1	0
16	**xxx.yyy.184.62**	0	4	2	5
17	xxx.yyy.184.120	0	6	2	6
18	xxx.yyy.184.123	0	3	5	9
19	xxx.yyy.184.149	0	4	3	6
20	**xxx.yyy.184.151**	0	3	5	13
21	**xxx.yyy.184.152**	0	4	3	16
22	xxx.yyy.184.155	0	1	1	5
23	xxx.yyy.184.200	0	1	1	0
24	xxx.yyy.184.231	0	12	6	16
25	xxx.yyy.184.233	0	4	2	7
26	xxx.yyy.184.236	0	10	5	15
	TOTAL	**2**	**98**	**64**	**212**

	Nessus	Risk Level			
	IP Address	Serious	High	Medium	Low
1	xxx.yyy.184.1	0	0	0	0
2	**xxx.yyy.184.2**	0	0	7	0
3	**xxx.yyy.184.9**	1	1	4	1
4	**xxx.yyy.184.11**	0	1	5	2
5	**xxx.yyy.184.17**	0	1	2	1
6	**xxx.yyy.184.22**	1	1	1	2
7	**xxx.yyy.184.28**	1	9	14	2
8	**xxx.yyy.184.51**	0	0	0	0
9	**xxx.yyy.184.52**	0	0	0	0
10	**xxx.yyy.184.53**	0	0	0	0
11	**xxx.yyy.184.54**	1	3	5	1
12	**xxx.yyy.184.55**	0	0	0	0
13	**xxx.yyy.184.56**	0	0	0	0
14	**xxx.yyy.184.57**	1	3	4	1
15	**xxx.yyy.184.58**	0	1	4	1
16	**xxx.yyy.184.59**	0	1	4	1
17	**xxx.yyy.184.60**	0	0	0	0
18	**xxx.yyy.184.61**	0	0	0	0
19	**xxx.yyy.184.62**	1	0	1	0
20	xxx.yyy.184.97	0	0	3	0
21	xxx.yyy.184.120	0	0	16	5
22	xxx.yyy.184.123	0	1	6	3
23	xxx.yyy.184.149	0	0	0	0
24	**xxx.yyy.184.151**	0	0	0	0
25	**xxx.yyy.184.152**	0	0	0	0
26	xxx.yyy.184.155	0	0	0	0
	TOTAL	**6**	**22**	**76**	**20**

Finally, normalize risk for server $R_i^{(s)}$and client $R_i^{(c)}$ will be calculated from Eq. (7) and (8). The total weighted cut-off for server ($\sum_{l=1}^{L} l \cdot f_s$) and client ($\sum_{l=1}^{L} l \cdot f_c$) are 34.167 and 40.625 respectively. An example to calculate the normalize risk of the 1st server (xxx.yyy.184.2), $R_s^{(1)}$, is as follows;

$$R_s^{(1)} = \frac{\sum_{l=1}^{L} \acute{n}_{c,l}}{\sum_{l=1}^{L} l \cdot f_c} = \frac{18.417}{34.167} \times 100 = 53.902\%$$

The relative risk for server $\bar{R}_s$ and client $\bar{R}_c$ can be estimated from Eq. (9) and (10) as follows:

$$\bar{R}_s = \frac{\sum_{i \in s} R_i^{(s)}}{H_s} = \frac{(18.417+5.417+16.250+\cdots+19.667)}{18} = 34.167$$

And,

$$\bar{R}_c = \frac{\sum_{i \in c} R_i^{(c)}}{H_c} = \frac{(20.250+21.188+22.063+\cdots+24.375)}{8} = 40.625$$

Table 4 The "Cut-off Normalize Table" respected to the server and client cut-off (left) and The "Weighted normalize Table" respected to the server and client cut-off (right).

	NetClarity	Risk Level			
	IP Address	Serious	High	Medium	Low
1	**xxx.yyy.184.2**	0.000	3.000	3.000	3.417
2	**xxx.yyy.184.9**	0.000	0.000	1.000	3.417
3	**xxx.yyy.184.11**	0.000	2.000	3.417	3.417
4	**xxx.yyy.184.22**	0.000	3.417	3.000	3.417
5	**xxx.yyy.184.28**	0.000	3.417	2.000	3.417
6	**xxx.yyy.184.51**	0.000	1.000	0.000	0.000
7	**xxx.yyy.184.52**	0.000	1.000	0.000	0.000
8	**xxx.yyy.184.54**	1.000	3.417	3.000	3.417
9	**xxx.yyy.184.55**	0.000	1.000	1.000	1.000
10	**xxx.yyy.184.56**	0.000	1.000	1.000	0.000
11	**xxx.yyy.184.57**	1.000	3.000	3.000	3.417
12	**xxx.yyy.184.58**	0.000	3.417	3.000	3.417
13	**xxx.yyy.184.59**	0.000	3.417	3.000	3.417
14	**xxx.yyy.184.60**	0.000	1.000	1.000	0.000
15	**xxx.yyy.184.61**	0.000	1.000	1.000	0.000
16	**xxx.yyy.184.62**	0.000	3.417	2.000	3.417
17	xxx.yyy.184.120	0.000	4.063	2.000	4.063
18	xxx.yyy.184.123	0.000	3.000	4.063	4.063
19	xxx.yyy.184.149	0.000	4.000	3.000	4.063
20	**xxx.yyy.184.151**	0.000	3.000	3.417	3.417
21	**xxx.yyy.184.152**	0.000	3.417	3.000	3.417
22	xxx.yyy.184.155	0.000	1.000	1.000	4.063
23	xxx.yyy.184.200	0.000	1.000	1.000	0.000
24	xxx.yyy.184.231	0.000	4.063	4.063	4.063
25	xxx.yyy.184.233	0.000	4.000	2.000	4.063
26	xxx.yyy.184.236	0.000	4.063	4.063	4.063

	NetClarity	Risk Level			
	IP Address	Serious	High	Medium	Low
1	**xxx.yyy.184.2**	0.000	9.000	6.000	3.417
2	**xxx.yyy.184.9**	0.000	0.000	2.000	3.417
3	**xxx.yyy.184.11**	0.000	6.000	6.833	3.417
4	**xxx.yyy.184.22**	0.000	10.250	6.000	3.417
5	**xxx.yyy.184.28**	0.000	10.250	4.000	3.417
6	**xxx.yyy.184.51**	0.000	3.000	0.000	0.000
7	**xxx.yyy.184.52**	0.000	3.000	0.000	0.000
8	**xxx.yyy.184.54**	4.000	10.250	6.000	3.417
9	**xxx.yyy.184.55**	0.000	3.000	2.000	1.000
10	**xxx.yyy.184.56**	0.000	3.000	2.000	0.000
11	**xxx.yyy.184.57**	4.000	9.000	6.000	3.417
12	**xxx.yyy.184.58**	0.000	10.250	6.000	3.417
13	**xxx.yyy.184.59**	0.000	10.250	6.000	3.417
14	**xxx.yyy.184.60**	0.000	3.000	2.000	0.000
15	**xxx.yyy.184.61**	0.000	3.000	2.000	0.000
16	**xxx.yyy.184.62**	0.000	10.250	4.000	3.417
17	xxx.yyy.184.120	0.000	12.188	4.000	4.063
18	xxx.yyy.184.123	0.000	9.000	8.125	4.063
19	xxx.yyy.184.149	0.000	12.000	6.000	4.063
20	**xxx.yyy.184.151**	0.000	9.000	6.833	3.417
21	**xxx.yyy.184.152**	0.000	10.250	6.000	3.417
22	xxx.yyy.184.155	0.000	3.000	2.000	4.063
23	xxx.yyy.184.200	0.000	3.000	2.000	0.000
24	xxx.yyy.184.231	0.000	12.188	8.125	4.063
25	xxx.yyy.184.233	0.000	12.000	4.000	4.063
26	xxx.yyy.184.236	0.000	12.188	8.125	4.063

In comparison, Nessus also detected 26 active hosts which they are 20 servers and 6 clients (Table 3). NetClarity addresses 376 vulnerabilities while Nessus finds 124 vulnerabilities (Table 5). Thus, in this experiment, NetClarity has better detecting performance than Nessus 3.032 times approximately. Applying the same procedure to Nessus for both DMZ and internal zone, the overall summary is shown in Table 6.

Table 5 Detected Vulnerabilities in RSU DMZ

Vulnerability Test Tools	Active Hosts		Detected Vulnerabilities				Overall
	Server	Client	Serious	High	Medium	Low	
NetClarity	18	8	2	98	64	212	376
Nessus	20	6	6	22	76	20	124

Table 6 Summary of server and client risk from RSU network: DMZ and internal zone

RSU Network	Type	Active Host H_s and H_c		Cutoff f_s and f_c		Total Cut-off $\sum_{l=1}^{L} \boldsymbol{l \cdot f_i}$		Relative Risks $\bar{R}_s$ and $\bar{R}_c$	
		NetClarity	Nessus	NetClarity	Nessus	NetClarity	Nessus	NetClarity	Nessus
DMZ : xxx.yyy.184.1-255	Server	18	20	3.417	1.125	34.167	11.250	40.068	38.111
	Client	8	6	4.063	1.417	40.625	14.167	45.038	16.863
Internal Zone : xxx.yyy.118.1-255/24	Server	7	3	3.214	0.333	32.143	3.333	40.159	13.333
	Client	8	18	2.750	0.306	27.500	3.056	44.773	12.778

The result shows that the relative risks of NetClarity and Nessus are significantly different. For example, in RSU DMZ, the client relative risk of NetClarity is 45.038 while it is 16.863 for Nessus (Table 6). These are very disparate and it could be more ambiguous if many detection tools are applied. Security professionals have to desire which detection tool is the most appropriate one that fits the organization environment.

In phase 2, server and client group will be considered separately. For instance, NetClarity detected 18 servers so H_s from NetClarity is 18. In the meanwhile, Nessus detected 20 servers so H_s from Nessus is 20. Therefore, the probability of trust ($P_s(T)$) from Nessus will be slightly higher than NetClarity. From Eq. (11), the probability of trust ($P_s(T)$) for NetClarity is 18/(18+20) = 0.474 and the probability of trust ($P_s(T)$) for Nessus is 20/(18+20) = 0.526 . From phrase 1, we found that the relative risk of server ($\bar{R}_s$) from NetClarity and Nessus are 40.068 and 38.111 respectively. Then, from Eq. (12), we can calculate the possibility of server risk ($P_s(R)$) which they are 0.190 for NetClarity and 0.201 for Nessus. Ironically, the total estimated risk can be calculated by Eq. (13). Thus, the total estimated risk of server type in DMZ is (0.190+0.201) x 100 = 39.038 %. The overall result of risk evaluation for RSU DMZ and internet zone are shown in Table 7 and 8 respectively. Clearly, the total estimated risk in percentage can reasonably degrade the differentiation of relative risk derived from NetClarity and Nessus. It is not just the mean of two values. The proposed "Network Risk Metric" can be applied to various number of vulnerability detection tools. The key ingredients are number of found hosts and vulnerabilities that form the probability of trust, possibility of risks and, finally, the total estimated risk.

Table 7 Result of Risk Evaluation for RSU DMZ

RSU Network	Relative Risk $\bar{R}_s$ and $\bar{R}_c$		Active Host H_s and H_c		Prob. Of Trust $P_s(T)$and $P_c(T)$		Possibility of Risks $P_s(R)$or $P_c(R)$		Total Estimated Risk (%)
	NetClarity	Nessus	NetClarity	Nessus	NetClarity	Nessus	NetClarity	Nessus	$P_s(R) + P_c(R)$
Server	40.068	38.111	18	20	0.474	0.526	0.190	0.201	**39.038**
Client	45.038	16.863	8	6	0.571	0.429	0.257	0.072	**32.963**

Table 8 Result of Risk Evaluation for RSU Internal zone

RSU Network	Relative Risk $\bar{R}_s$ and $\bar{R}_c$		Active Host H_s and H_c		Prob. Of Trust $P_s(T)$ and $P_c(T)$		Possibility of Risks $P_s(R)$ or $P_c(R)$		Total Estimated Risk (%)
	NetClarity	Nessus	NetClarity	Nessus	NetClarity	Nessus	NetClarity	Nessus	$P_s(R) + P_c(R)$
Server	40.159	13.333	7	3	0.700	0.300	0.281	0.040	**32.111**
Client	45.038	16.863	8	18	0.308	0.692	0.138	0.088	**22.622**

Discussion and Conclusion

This paper investigates vulnerability detection performance between different test tools. From the experiment, NetClarity has approximately 3.032 times better detecting performance than Nessus by number of hosts but it is much slower. Both of them classify risk into 4 levels. However, they used different classification standard. Thus, numbers of detected vulnerability in each level are significantly varies depended on what detection tools are applied. Total risk analysis remains unclear. The security metric, including application, network, and software security, are still open. Therefore, this paper proposed "Network Risk Metric" to evaluate risk from various vulnerability detection tools. Two major ideas are applied: "Weight-Cut-off Severity Normalization" and "Probability of Trust" respected to detected hosts. Experimental results show that the total estimated risk (%) derived from "Network Risk Metric" is effective since it can compromise risk classification from distinct vulnerability test tools in more reasonable way not bias by any organizations or vendors. Fuzzy Logic will be applied to the "Network Risk Metric" in the future work for better performance.

Acknowledgements

This experiment is supported by Rangsit University Grants.

References

1. Tito Waluyo, Budi Rahardjo, and Kuspriyanto (2011) Security Metrics: A Brief Survey. IEEE, International Conference on Instrumentation, Communication, Information Technology and Biomedical Engineering.
2. Reijo M. Savola (2009) A Security Metrics Taxonomization Model for Software-Intensive Systems. Journal of Information Processing Systems, Vol. 5, No. 4
3. Wayne Jansen (2009) Research Direction in Security Metrics. In Processing of the 8th Annual Security Conference, Las Vegus, NV, USA
4. Golnaz Elahi, Eric Yu, and Nicola Zannone (2011) Security Risk Management by Qualitative Vulnerability Analysis. IEEE, Third International Workshop on Security Measurements and Metrics.
5. Peter Kok Keong Loh, and Deepak Subramanian (2010) Fuzzy Classification Metrics for Scanner Assessment and Vulnerability Reporting. IEEE Transaction on Information Forenics and Security, Vol. 5, No. 4.
6. DHS National Security Division, NIST, Web Application Vulnerability Scanners. Available: https://samet.nist.gov/index.php/Web_Application_Vulnerability_Scanners
7. D. Subramanian, H.T. Le, and P. K. K. Loh. F (2009) Fuzzy heuristic design for diagnosis of web-based vulnerabilities. In Processing of Fourth International Conference Internet Monitoring and Protection (ICIMP), Venice/Mestre, Italy.
8. Forum of Incident Response and Security Teams. http://www.first.org . 2007