A Graph theoretical approach to Network Vulnerability Analysis and Countermeasures

Dr.Thaier Hamid
University of Bedfordshire, UK

Prof. Carsten Maple,
University of Bedfordshire, UK

ABSTRACT

Computer networks are certainly vulnerable as long as they deliver services from different machines. An attack graph is a security model representing the chains of vulnerability exploits in a network displays the ways an attacker can compromise a network or host. A number of researchers have admitted attack graph visual complications and a large amount of source data must be assembled to accurately build an attack graph, the difficulty scaling to large, enterprise-size networks with tens of thousands of hosts and the lack comprehensive understanding. Information on vulnerabilities is present in public vulnerability databases, such as the National Vulnerability Database and Nessus. But current attack graph tools are reserved to only limited attributes. The automatic formation of vulnerability information has been troublesome and vulnerability descriptions were created by hand or based on limited information. Much vulnerability has still not been discovered and many others without patches or solutions Our approach to developing a cost metric exploits the Markov's model using combinations well known vulnerabilities (the Common Vulnerability Scoring System, CVSS) and Risk Assessment Values (RAV) and using ranking algorithms (similar to V. Mehta et al. 2006 and kijsanayothin, 2010) but instead of using vulnerabilities. For each host we have developed a cost rank Markov's model reducing the complexity in the attack graph, representing the network topology and dipping the problem of visibility.

General Terms

Graph theoretical, Algorithms, Experimentation

Keywords

Ranking attack graph, Network security, Security metrics..

1. INTRODUCTION

To define the security effect as applications and operating systems vulnerabilities on a specific network, one must study communications among several network components. For a vulnerability analysis tool to be useful, two features are essential. First, the model used in the analysis must be able to automatically integrate recognized vulnerability conditions from the public vulnerability databases community. Second, the investigation must be capable of measuring networks with thousands of machines relying on software providers that are subject to fault, making them susceptible to malicious attacks. Although the internet, which has brought many benefits to organizations and individuals, it has also increased the risks of having hosts compromised without the need of physical access. Network vulnerabilities refer to exploitable problems in configurations (e.g., ports and services enabled) or the software implemented to provide network services (e.g., Apache Chunked-Code on Apache web servers, buffer overflow on Windows XP SP2, Vista and 7 other operating environments, including TNS Listener on Oracle software for database servers). Figure 1 reflects the number of published vulnerabilities reported by Secunian [6], and the number of Common Vulnerabilities and Exposures (CVEs) disclosed per year since 2005 with a breakdown of the solution status ("Unpatched", "patched", "total"). As shown in figure 1, on average there were 4,464 CVEs per year from 2005 to 2009.

An extrapolation of the data of the first half of 2011 leads us to expect that 2011 will exceed the number of CVEs of 2009, but not the average of the last five years. It should be noted that older vulnerabilities are more likely to have a patch available than recently found vulnerabilities. We can see that from 2005 to 2011 we had a sharp increase in the number of vulnerabilities. In 2005-2006 and from 2006-2007 and 2008-2009, there was an insignificant decrease in 2006-2007 and from 2008-2009, of 1.8%. This was due in part to the release of new operating systems at the time. Nevertheless, in 2008 alone, a total of 4,800 vulnerabilities were published. As shown in Figure 1. Consequently, this volume of vulnerabilities requires management from organizations to determine which of the daily reported vulnerabilities apply to their situation according to the software they have installed, their version and configuration.

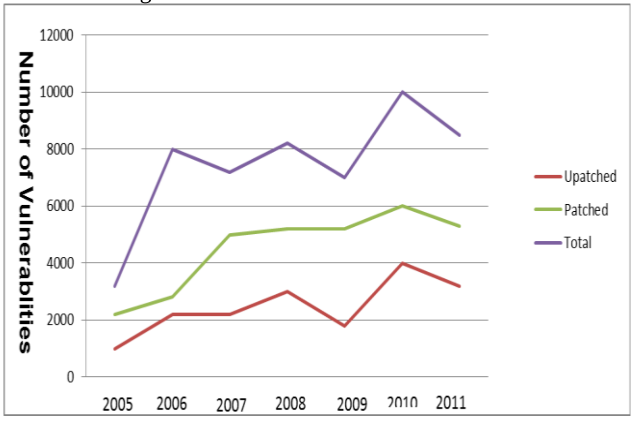

Fig. 1 Number of published vulnerabilities from 2005-2011.

All in all, solving every vulnerability, for example by patching nonnative services and hosts, is never a solution for multi-step attacks. There are basically two types of attack graphs. In the first type, each vertex represents the entire network state and the arcs represent state transitions caused by an attacker's actions. Examples are Sheyner's scenario graph based on model checking [9], and the attack graph in Swiler and Phillips' work [15]. This type of attack graph is sometimes called a state enumeration attack graph [9]. In the second type of attack graph, a vertex does not represent the entire state of a system but rather a system condition in some form of logical sentence. The arcs in these graphs represent the interconnection relations between the system conditions. We call this type of attack graph a dependency attack graph. Examples are the graph structure used by Ammann et al, state enumeration attack graph in general tend to suffer from the drawback of the state explosion problem [15]. It means the complexity of the graph generated grows exponentially O (2n), in terms of the size of the state space n. The key difference between the two types of attack graphs lies in the semantics of their vertices. While each vertex in a state enumeration attack graph encodes all the conditions in the network, a vertex in a dependency attack graph encodes a single attack asset of the network. A path s1 →s2→ s3 in a state enumeration attack graph means that the system's state can be transitioned from s1 to s2 and then to s3 by an attacker. But the condition that enables the transition s2→ s3 may have already become true in a previous state, say s1. The reason the attacker can get to state s3 is encoded in some state variables in s2, but the arcs in the graph do not directly show where these conditions were first enabled. In a dependency attack graph, however, the dependency relations among various assets are directly represented by the arcs.

1.1Query independent Link based Ranking-PageRank

We give the following dentition of PageRank ($\Re$). Let u be a web page (a vertex in the graph). Then let Fu be the set of forward links of u and let Bu be the set of backlinks of u. Let also du = |Fu| be the out-degree of u. Let also $\Re$ u be the $\Re$ of page u. Then the $\Re$ values of the pages are calculated as follows. Initially, each page is assigned a $\Re$ value of 1/n where n is the number of pages.

$$\Re v = \quad \sum\nolimits_{u \in Bv} \frac{\Re u}{du} \qquad for \forall u \in V$$

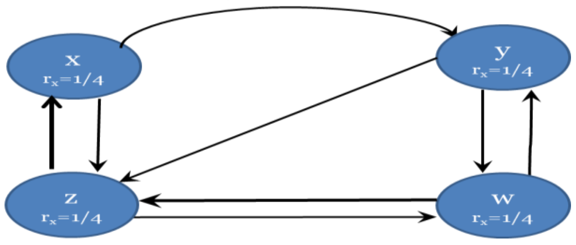

Fig. 3 A simple web graph with vertex weight.

Let P be a row stochastic matrix corresponding to the graph G of the web pages, assuming that all edges have at least one outgoing edge. If there is a link from page x to page y, then let the matrix entry P_{xy} have the value 1/di. Let all other entries have the value 0.

P	x	y	z	w
x	0	1/2	1/2	0
y	0	0	1/2	1/2
z	1/2	0	0	1/2
w	0	1/2	1/2	0

Table 1: transition probability matrix

Then, one iteration of the $\Re$ computation corresponds to the vector matrix multiplication [$\Re$] 1n x [P]nn. Repeated multiplication of r by P yields the dominant eigenvector r of the matrix P. Since P corresponds to the stochastic transition matrix of the graph G, PageRank, which is denoted by r, This r can be viewed as the stationary probability distribution over pages induced by a random walk on the web graph [10]. Because P is stochastic, its dominant eigenvalue is one. This leads to the following equation:

$$\Re^{(t)} = \Re^{(t-1)} * P$$

$$\Re^{(0)} \qquad * \qquad P = \qquad\qquad \Re^{(1)}$$

$$[\ ¼\ ¼\ ¼\ ¼\ ] * \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix} = [\ \frac{1}{8} \quad \frac{2}{8} \quad \frac{3}{8} \quad \frac{2}{8}\ ]$$

Since $\Re$ models the random walk of a web surfer over the web pages, the calculation mentioned above also corresponds to finding a stationary probability distribution for a Markov chain in which web pages represents the states and the matrix P represents the state transition matrix for the states in the chain. Discrete time steps in the chain and corresponds to the iterations of $\Re$ calculation. As we see in figure-4 the ranking after 13 iterations is **x**=0.16, **y**=0.22, **z**=0.33, **w**=0.27, z in this case it is the most important web site.

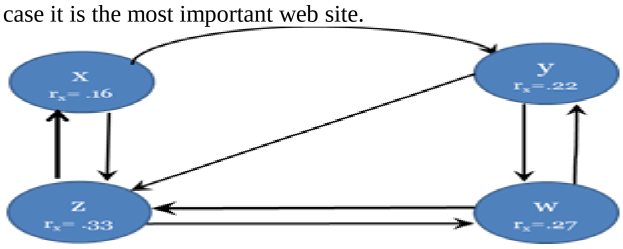

Fig. 4 A simple web graph with vertex weight after 13 iterations.

1.2 Security Metrics

The Common Vulnerability Scoring System, CVSS, calculator [15, 22], maintained The Forum of Incident Response and Security Teams, which was launched in 2004 and is currently on its second version. The CVSS is composed of three metrics groups: base, temporal and environmental, Base: represents the basic and fundamental characteristics of vulnerability that are constant over time and user environments. The Access Vector, Access Complexity, and Authentication Metrics capture how the vulnerability is accessed and whether or not extra conditions are required to exploit it. Risk Assessment Values (RAV) is the computation of security operations [7], controls, and limitations, which represents the current state of protection. The RAV is a derivative from three categories defined within the scope: Operational Security, Loss Controls and Security Limitations. We added the Protocol Complexity Cost to RAV Components as a measure of difficulty that an attacker encounters when trying to bypass channels implementing different protocols (e.g., FTP, SMTP, SSH, VPN) to progress across network nodes.

$$OP\sec_{sum} = P_V + P_A + P_T + P_P$$

$$OP\sec_{sum} = \log^2(1 + 100 + OP\sec_{sum})$$

Our methodology combines the CVSS score with the Risk Assessment Values. A RAV is the computation of security operations, controls, and limitations, for that we added to the cost metric calculator CV base as follows:

$CV_{Base} =$ round_to_1_decimal (((0.6*Impact)+(0.4*Exploitability)–1.5)*f(Impact)).

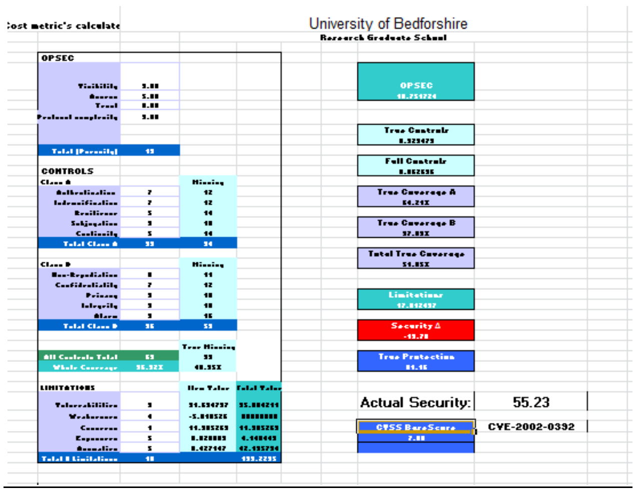

Fig. 5 RAV cost calculator.

2. METHODOLOGIES FOR DEVELOPING COST-CENTRIC APPROACH

Instead of specifying a state by network attributes, we propose cost-centric model checking, in which each state is specified by the attributes of a single host. We refer to the corresponding attack graphs, generated with model checking algorithms, as cost-centric attack graphs.

In our model, the attributes of any host X include:

1- Attacker's access privilege on host X: Privilege level, i.e., root or system administrator (2), privileged user (1), user, guest, or none(0).

2- Security Metrics cost for each host : represent the CVSS score with the Risk Assessment Values(RAV) for each host.

3- Exploit mode: The locality of an attacker performing an exploit.

To illustrate the proposed approach in details, consider a simple network as shown in figure-6 taken from V. Mehta et al. (2006) and used by kijsanayothin (2010) to compare the results , where there are two service hosts: IP1 and IP2, and an attacker's workstation, Attacker, connecting to each of the servers via a central router.

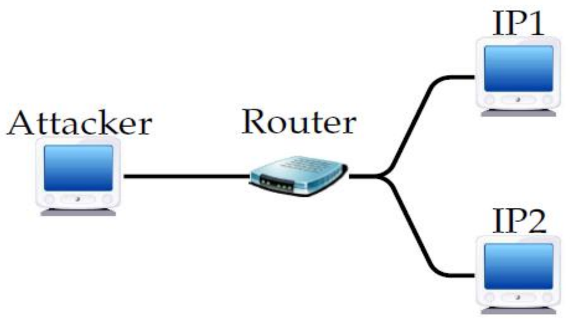

Fig. 6 a simple network taken from V. Mehta et al. 2006.

(Att , 2)	(IP 1 , 2)	(IP 1 , 1)	(IP 2, 1)	(IP 2 , 2)
S0	S1	S2	S3	S4

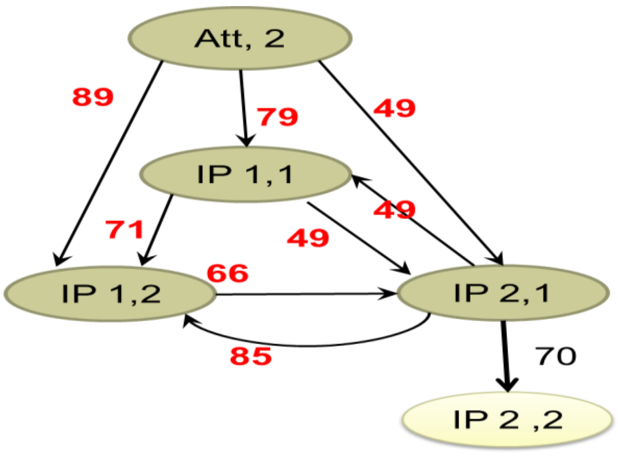

Fig. 7 cost attack graph with arc weight.

To get the normalized cost matrix we used the following formula:

$$\psi(u,v)=\frac{c(u,v)}{\sum_{\forall w\in Ov} c(u,w)}$$

Where $Ov \rightarrow \quad \forall w \in out(u)$

For example

$$\psi_{s0\rightarrow s1}=\frac{89}{89+79+49}=0.410138$$

Table 2. Personation cost matrix

	S0	S1	S2	S3	S4
S0	0	0.410138	0.364055	0.225806	0
S1	0	0	0	1	0
S2	0	0.591667	0	0.408333	0
S3	0	0.416667	0.240196	0	0.343137
S4	0	0	0	0	1

let δv be the probability of intrusion of attack state v at time t and d be a damping factor representing the probability of an attacker to continue penetrating the current path of the attack. Thus, in the context of network.

$$\forall v \delta_{(i+1)}(v)=\frac{1-d}{N}+d\sum_{\forall u\in Bv}\delta(u)*\psi(u,v)$$

When v is an initial state.

$$\forall v \qquad \delta_{(i+1)}(v)=d\sum_{\forall u\in Bv}\delta(u)*\psi(u,v) \qquad Otherwise$$

Where Bv represent the set of pages pointing to v, in(v).

Table 3. Comparing the Results

state	Mehta et al.'s approach	Kijsanayothin approach	Our approach
S0	0.1500	0.1500	0.1500
S1	0.1450	0.1287	0.1809
S2	0.1020	0.1658	0.0929
S3	0.2090	0.2548	0.1290
S4	0.3940	0.3007	0.4472

The Table shows the ranking result obtained when we applied the experiential, that is- s4, s1, s0, s3, s2 (i.e., our approach) . Otherwise, s4, s3, s0, s1, s2 is obtained when Mehta et al.'s approach is applied and s4, s3, s2, s0, s1 when Kijsanayothin approach is applied..

2.1 Initial results and discussion

The ranking is in the order of likelihood of intrusion based on exploitability. For example, all results suggest that s4 has the highest (relative) likelihood of being attacked (i.e., highest exploitability and most vulnerable). To further compare the ranking results, if we omit s0 in all ranking lists, both ranking orders generally agree except a conflicting case of ranking order between s1, s2 and s3.

1- { s1, s2} :Consider attackers from the initial state. As shown in page 21 to reach state s1 (e.g., from s0, s2 or s3) requires exploiting cost 89 71 85, where as to reach state s2 (e.g., from s0 or s3) requires exploiting cost 79 85. However, according to simple calculations s1 is more vulnerable than s2, in addition that reach to s1 mean the attacker has the ability to compromise a host and reach to the root of IP1 this is not the case for s2. Therefore, s1 should rank higher than s2.
2- {s2, s3}: starting from an initial state s0, to reach s4 through s2 requires three applications of costs of exploitability value 79 49 70 , whereas to reach s4 through s3 only requires two applications of cost s of exploitability value 49 70 . Therefore, s3 should rank higher than s2.
3- {s1, s3} :Consider attackers from the initial state. To reach state s1 (e.g., from s0, s2 or s3) requires exploiting cost 89 71 85, where as to reach state s3 (e.g., from s0 or s2 or s1) requires exploiting cost 49 49 66. Therefore, s1 should rank higher than s3 because s1 is more vulnerable than s3.
4- {s3, s4}. If there is only one way to reach s4 by an attack path from s0 to s3 to s4 then s3 ranks higher than s4. However, we can reach s4 by another attack path from s0 to s2 to s4 and attack path s0 to s3 to s4 Thus, s4 ranks higher than s3.

From the results above the system administrator should give first priority to s4 (IP 2,2) { compromised the root of IP2} as it has the highest (relative) likelihood of being attacked. It has the highest exploitability and is the most most vulnerable. Solutions should be immediately implemented then in the second place s2(IP 1,2) { compromised the root of IP1} then s3(IP 2,1) then s2(IP 1, 1) as we explained.

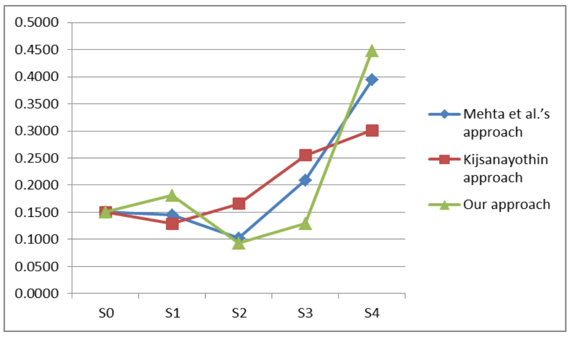

Fig.8 Initial ranking results

3. INVESTIGATES

In the following example network, the attacker is initially located in host A and wants to reach either host E that is protectedby a firewall.

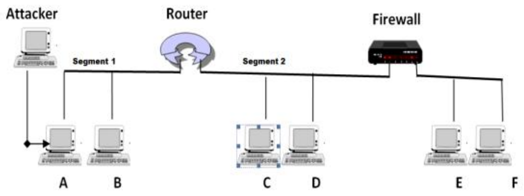

Fig. 9 An example network, modified from [16]

In this example network, the attacker is initially located in host A and wants to reach host E that is protected by a firewall. The firewall only allows traffic from hosts C or D to host E. Router (Ro) connects segment 1 & 2. Additionally, all hosts have a CVE-2003-0818: Microsoft Windows ASN.1 Library Integer vulnerability in the single service they provide. Successful exploitation of this vulnerability results in a buffer overflow condition allowing the attacker to execute arbitrary codes with administrative (system) privileges, potentially causing a complete impact on its confidentiality (C), integrity (I) and availability (A).

Table-4 below shows the ranking result obtained when we applied the experimental approach:

s12, s11, s10, s7, s9, s6, s8, s5, s3, s2, s1,s4.

Table -4. Ranking Results

S0	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12
0.0769	0.0441	0.0515	0.0520	0.0414	0.0544	0.0715	0.0904	0.0687	0.0846	0.1178	0.1573	0.1663

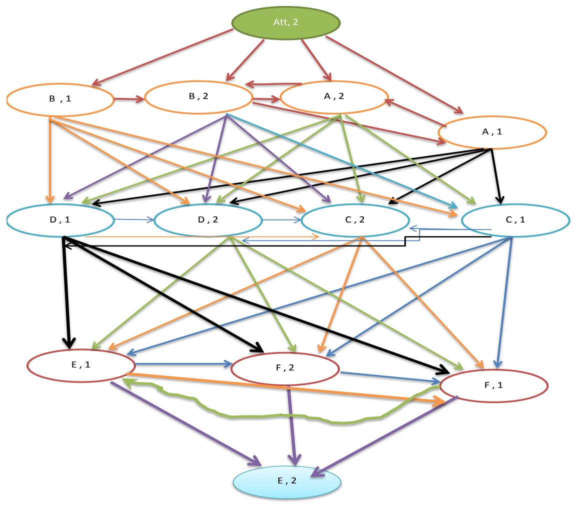

Fig. 9 A worst case attack graph.

4. DISCUSSIONS OF THE RESULTS

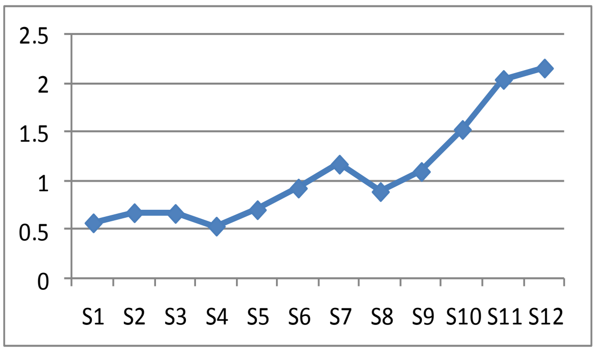

Fig.10 Cost ranking result

1. {s12, s11} we can reach s12 by another attack path to s4 thus, s12 ranks higher than s11.
2. {s11, s10} $\sum$ cost to reach s11: 87 85 87 88 72 69 69= 557, but the total cost to reach s10 from initial state=482 Therefore, s11 should rank higher than s10 (same segment).
3. {s9, s7} $\sum cost$ to reach s9= 374 and to s7 =528 from initial state, Therefore, s7 should rank higher than s9.
4. {s9, s8} because s9 (f1) is in the same segment while s7(c1) in other segment2 across, the firewall only allows traffic from hosts C or D to host E. Therefore, s9 should rank higher than s8.
5. {s6, s7, s8} total cost $\sum cost$ to reach s6= 434, and the total cost to reach s7 from initial state=528, the total cost to reach s8 from initial state=407 Therefore s7(cm 2), should rank higher than s6, s8 and s6 rank higher from s8 (same segment).
   {s6,s5} total cost to reach s6= 434, while the total cost to reach s5 from initial state=334 Therefore, s6 should rank higher than s5, specially s6= (d, 2) compromised the root. (same segment).
6. {s5, s4} total cost to reach s4= 209 and to s5 =334 from initial state, Therefore, s5 should rank higher than s4.
7. {s3, s2} from initial state, Therefore, s3 should rank higher than s2.
8. {s1, s2} total cost to reach s2= 257 and to s1 = 222 from initial state, Therefore, s2 should rank higher than s1.
9. {s1, s2} total cost to reach s2= 257 and to s1 = 222 from initial state, Therefore, s2 should rank higher than s1.

4. CONCLUSIONS

1. Advance a new methodology to represent attack graph with cost metrics.
2. Develop new methodology to calculate and represent the cost for each host.
3. Model the problem's parameters into a mathematical framework.
4. Model manual cost calculator to be used in our experiments .
5. Analyze and compare the results of our ranking algorithms with those of Mehta et al. and Kijsanayothin approach respectively.
6. Analyze and compare the results of our ranking algorithms to a medium size network.

5. REFERENCES

[1] L. He and N. Bode. Network Penetration Testing. In EC2ND 2005: Proc. of the First European Conference on Computer Network Defense, pages 3-12, London,2006. Springer-Verlag.
[2] E. W. Dijkstra. A Discipline of Programming. Prentice Hall, Upper Saddle River, NJ, USA, 1976.
[3] C. S. Wright. A Taxonomy of Information Systems Audits, Assessments and Reviews. SANS Institute, June 2007.
[4] B. Schneier. Attack trees: Modeling security threats. Dr. Dobb's Journal, December 1999.
[5] W. E. Wesely, F. F. Goldberg, N. H. Roberts, and D. F. Haasl. Fault Tree
[6] Secunia Half Year Report 2010, http://secunia.com/gfx/pdf/Secunia_Half_Year_Report_2010.pdf.
[7] OSSTMM 3 – The Open Source Security Testing Methodology Manual, Creative Commons 3.0 Attribution-NoDerivs 2001-2010, ISECOM, http://www.isecom.org.
[8] Mehta, V., C. Bartzis, H. Zhu, E. M. Clarke, and J. M. Wing (2006). Ranking attack graphs. In D. Zamboni and C. Kr ¨ugel (Eds.), Recent Advances in Intrusion Detection, Volume 4219 of Lecture Notes in Computer Science, pp. 127–144. Springer.
[9] Phongphun Kijsanayothin, Network Security Modeling with Intelligent and Complexity Analysis, 2010.
[10] ISEGCOM, SCARE 0.1 - The Source Code Analysis Risk Evaluation 15. November 2007, www.isecom.org.
[11] D. Geer and J. Harthorne. Penetration Testing: A Duet. In ACSAC'02: Proc. of the 18th Annual Computer Security Applications Conference, page 185, Washington, DC, USA, 2002. IEEE Computer Society.
[12] Python. Programming language. http://www.python.org/.
[13] Tenable network security: The Nessus Security Scanner. http://www.nessus.org. Visited 10-July-2008.
[14] S. Noel, M. Jacobs, P. Kalapa, and S. Jajodia. Multiple Coordinated Viewsfor Network Attack Graphs. In VIZSEC'05: Proc. of the IEEE Workshops on Visualization for Computer Security, page 12, Washington, DC, USA, 2005. IEEE Computer Society.
[15] S. Noel and S. Jajodia. Understanding Complex Network Attack Graphs through Clustered Adjacency Matrices. In ACSAC '05: Proceedings of the 21st Annual Computer Security Applications Conference, pages 160{169, Washington, DC, USA, 2005. IEEE Computer Society.

[16] R. Sawilla and X. Ou. Googling Attack Graphs. Technical Report TM-2007-205, Defense Research and Development Canada, September 2007.

[17] Brin, S. and L. Page (1998). The anatomy of a large-scale hypertextual web search engine. Compute. Netw. ISDN Syst. 30(1-7), 107–117.

[18] Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. J. ACM 46(5), 604–632.

[19] Gy¨ongyi, Z., H. Garcia-Molina, and J. Pedersen (2004). Combating web spam with trustrank. In VLDB '04: Proceedings of the Thirtieth international conference on Very large data bases, pp. 576–587. VLDB Endowment.

[20] ISEGCOM, SCARE 0.1 - The Source Code Analysis Risk Evaluation 15. November 2007, www.isecom.org.

[21] D. Geer and J. Harthorne. Penetration Testing: A Duet. In ACSAC'02: Proc. of the 18th Annual Computer Security Applications Conference, page 185, Washington, DC, USA, 2002. IEEE Computer Society.

[22] Tenable network security: The Nessus Security Scanner. http://www.nessus.org. Visited 10-July-2008.