Algorithm 6: Pseudocode for $evalCycleNode(D, N)$

Require: Parameters D, N, such that
$D\psi$d-separates all $n \in N$, and
$N\psi = \{n_0, n_1, ..., n_j\}$ are partial values for some node $n \in C\psi$

if $|N| > 1$ **then**
  return $\prod_{n \in N} evalCycleNode(D, \{n\})$
else if $N\psi = \{\overline{n}\}$ **then**
  return $1 - evalCycleNode(D, \{n\})$
else { $N\psi = \{n\}$, so that $N\psi$contains exactly one element, enabled }
  $P\psi \leftarrow$ partial path (set of attack-step nodes) leading to node instance $n\psi$
  $J\psi \leftarrow \{j\psi \mid j\psi \in D\}$ { All enabled nodes in $D\psi$}
  $K\psi \leftarrow \{k\psi \mid \overline{k}\psi \in D\}$ { All disabled nodes in $D\psi$}

  { If node in path is negated, $n\psi$cannot be reached by path $P\psi$}
  if $K\psi \cap P\psi \neq \emptyset$ **then**
    return 0
  end if

  { Discard any path nodes forced true }
  $P\psi \leftarrow P\psi \setminus J\psi$

  return $\prod_{p \in P} G_M[p]$
end if

Once all such fixed values have been accounted for, we know that the node is reachable along path P. Because the attack-step nodes in $P\psi$are treated independently, knowing the probability that all will jointly succeed gives us the likelihood that an attacker will succeed along path P. The algorithm therefore calculates the product of the component metric values for all remaining nodes in path P; this value is the probability that $n\psi$is true by path P, given set D.

5 Evaluation Results

We have implemented our cumulative metric algorithm in the Python language. To evaluate the effectiveness of using the metric model for risk assessment, we carried out three lines of study:

- Testing how the metrics can help making security hardening decisions.
- Evaluation on a production system to gain empirical experience of the metric model.
- Testing the scalability of metric computation.

5.1 Evaluating the use of metrics to guide hardening decisions

For this evaluation, we used the small example network in Figure 2. For this configuration, the cumulative metrics result is shown in the "Initial scenario" column of Table 1. In the table, the numbers indicate the likelihood various machines can be successfully compromised by an attacker.

Host	Initial scenario	Patch web server	Patch db server	Patch workstations	Change network access
Database server	0.47	0.43	0	0.12	0.12
Web server	0.2	0	0.2	0.2	0.2
Workstations	0.74	0.72	0.74	0	0.74

Table 1: Probabilities of compromise for hosts in Figure 2 (columns reflect different scenarios)

Consider again the sample network configuration (and associated attack graph) shown in Figure 2. When considering improvements in network security, a network administrator is constrained in terms of money and time. For example, some changes, though preferable, may not be feasible because of the time necessary to make the change and the system downtime that would occur while the change was made. Considering the network topology in this example, it is not immediately clear which of the vulnerabilities should be patched first, assuming that a fix is available for each of the three, or what other changes could be made to reduce security risk. The columns in Table 1 show new metric values based on various mitigation options: patching different vulnerabilities or changing the network access rules so that the user workstations cannot access the database server.

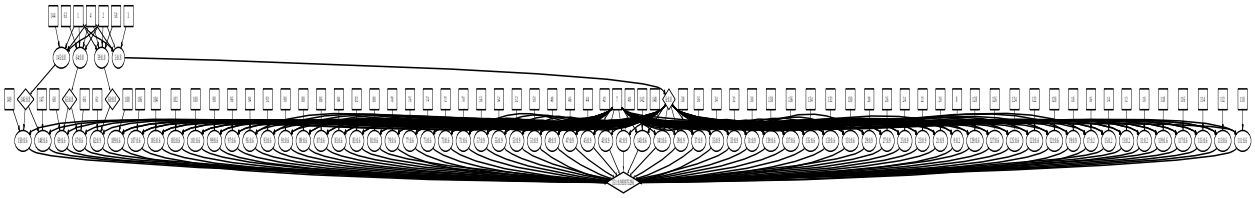
Patching the vulnerability on the web server would eliminate the known risk of compromise for the web server, but have little effect on the other two hosts. The web server does not contain sensitive information, so protecting this host first may not be the best choice.

Patching the vulnerability on the database server would eliminate the known risk of compromise for the database server, but have no effect on the risk in the other two hosts, since privileges on the database server do not enable new attacks on the other hosts. This option would secure the sensitive data on the database server, which may be most desirable, but at the cost of having a period of downtime on the database server which may affect business revenues.

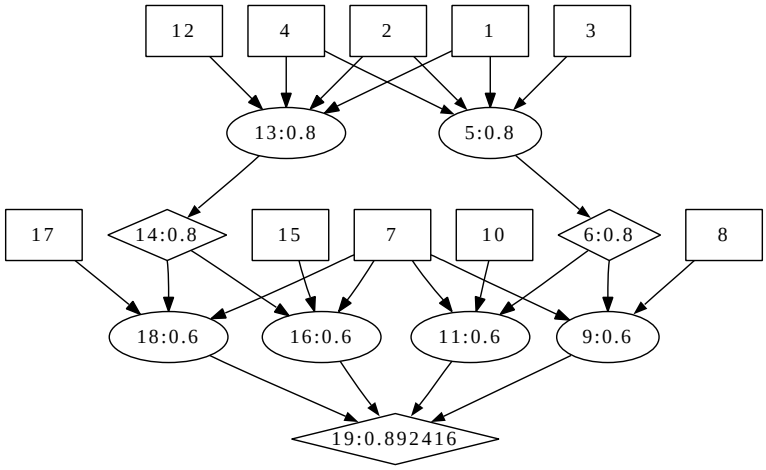
Patching the vulnerability on the user workstations would eliminate the risk to the workstations, as well as significantly reduce the risk to the database server, but the risk to the web server would remain unchanged. This may be a more feasible solution since downtime on the workstations is less costly than on the server, especially if the patching can be done outside of normal working hours.

Network configuration changes can also have drastic effects on the security risk. The final column in the table shows the effect of blocking network access from the workstations to the database server. This option eliminates an attack path to the database server that depends on privileges on the workstations, lowering the risk of compromise for the database server, but it leaves the web server and workstations vulnerable. Depending on other resource constraints and asset valuations, this may also be a viable solution.

There may not be a single "best" option for all organizations. Indeed, different administrators could easily make different choices, based on the perceived importance of the hosts and the expected time necessary to enact proposed changes, as well as human resources available. The quantitative security metrics make clear the effects emerging from each of these possible changes, thereby providing a network administrator with objective data beneficial for judging the relative value of each option. Our cumulative metrics could also be combined with quantitative asset values and costs of various mitigation options, fed into an optimization engine such as the one proposed by earlier works [11, 28, 38, 46], to automatically compute optimal hardening options.



(a)



(b)

Figure 6: Attack graphs from two production servers

5.2 Insights gained from evaluation on a production system

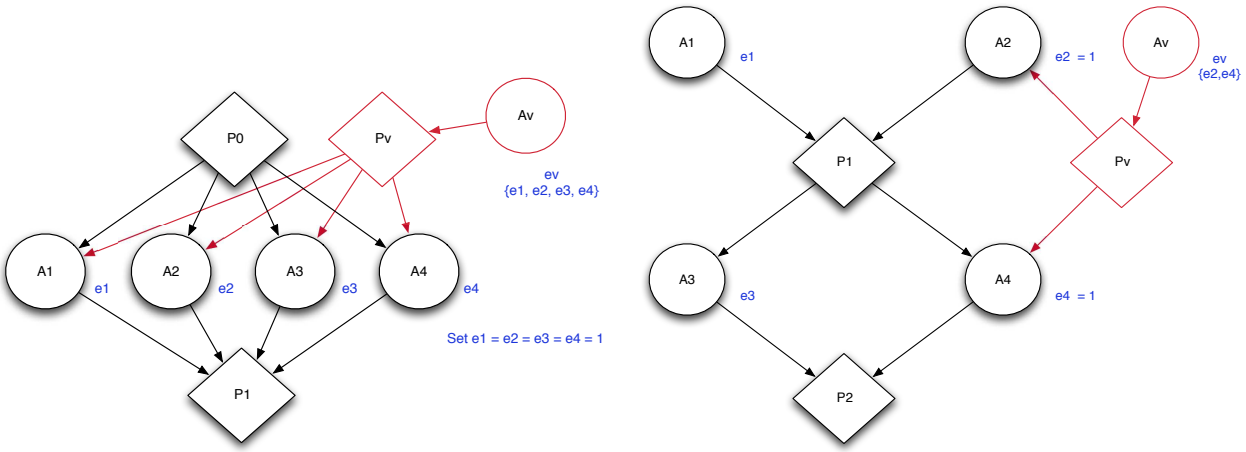
To study how the metric model works on production systems, we have conducted an OVAL vulnerability scan on all the Windows servers and workstations in the CIS departmental network of Kansas State University. OVAL[2] is part of the SCAP standard [36] for communicating security information. It is a language for reporting discovered known vulnerabilities on a host. The OVAL scan is performed periodically, and the vulnerability assessment reports are automatically sent to a central data repository, providing continuous fresh data for evaluating our metric model.

The departmental network has a fairly simple network topology. There is no firewall control in the internal network, so all the servers can talk to each other. The servers are well-managed so that most of the service program vulnerabilities have been patched. However, there are still a large number of client-side and local vulnerabilities on each machine. These vulnerabilities pose relatively low risk, since it is very unlikely that a user will access the server to launch those client programs, and as long as no user is compromised, the local vulnerabilities cannot pose any danger to the systems. For this reason, these systems are good candidates for evaluating the security metric methods — there is a significant amount of residual risk that needs to be quantified. The calculated security metrics can be used in comparison to the system administrator's rationale for delayed patching of these non-critical vulnerabilities.

As we ran our metric algorithm on the model, a problem quickly became obvious. It is best illustrated by the results in Figure 6, which shows the attack graphs for two servers.[3] Server (a) has many more vulnerabilities than (b), as can be seen immediately from the density of the attack graphs. The attack graph for (a) is so wide that it is shown almost like a line in the limited space on paper. However, when it comes

[2](http://oval.mitre.org/)

[3]The square nodes are configuration nodes which have been omitted in the previous attack-graph examples.



(a) (b)

Figure 7: Modeling artifacts for capturing hidden correlations

to cumulative security metrics results, the two do not show a significant difference: 0.99 vs. 0.89. Indeed, most of the machines we have evaluated had at least a dozen attack paths, which raises the likelihood of attack success to almost 1. This necessarily prompts the question: is this a realistic measurement of risk? We presented the result to the system administrator. His opinion was that it depends on the underlying differences in the existing vulnerabilities. A machine with 10 vulnerabilities in a single application has a lower risk than a machine with 10 vulnerabilities in 10 different applications, because vulnerabilities in the same application may not give an attacker significant advantage in exploiting them. The exploits for these vulnerabilities may not be truly independent: if the attacker lacks skill or experience in exploiting a specific application, the presence of more vulnerabilities in that application will not help a lot. On the other hand, if the 10 vulnerabilities are dispersed across 10 different applications, the chance that the attacker possesses the skill to exploit at least one of them will be significantly higher. This dependency among exploits based on the similarity of applications is not captured by the attack-graph structure. For the attack graph in (a), there are only four vulnerable applications but 67 distinct vulnerabilities, of which 62 are in the same application (Firefox). For (b), there are four vulnerabilities in two applications. Intuitively, (a) has a much higher risk than (b), more than the difference between the two calculated metric numbers indicate.

This observation has also led to another manifestation of the hidden-correlation problem. Suppose the same vulnerability appears on two different hosts and the attacker needs to exploit both of them in a multi-stage attack. If he has succeeded in exploiting the first one, he will very likely succeed with the second. This dependency also is not captured by the attack graph, which could lead to a lower evaluation of risk than really exists. If the chance of success for the attacker to exploit the vulnerability is 0.6, the likelihood for him to succeed in the two-stage attack chain should be very close to 0.6, since a successful attack in the first step will lead, with a likelihood of almost 1, to success in the second. Based on the attack-graph model, however, our metric algorithm will produce a result of 0.36, by multiplying the two probabilities.

Modeling artifacts for capturing hidden correlations To correctly account for such hidden correlations among attack steps, we introduced additional modeling artifacts in attack graphs so that the hidden correlations become explicit. Essentially, we grouped vulnerabilities for the same application and introduced a virtual modeling node to capture the case in which an attacker has succeeded in exploiting the application. This applies to both vulnerabilities on the same host (Figure 7.a) and on different hosts (Figure 7.b). The success likelihood of exploiting this vulnerability is associated with the added virtual exploit node A_v, and

the original exploit nodes are associated with a likelihood of 1. This makes the hidden correlation explicit in the graphical model. In (a), if the attacker fails in exploiting the vulnerability, he will fail on all four instances $A_1, ...A_4$, avoiding (incorrectly) bumping the success likelihood of P_1 to almost 1. In (b), if the attacker succeeds in exploiting the vulnerability, he will succeed in both attacks A_2 and A_4, avoiding (incorrectly) discounting the success likelihood of the two-step chain.

After this revision of the graphical model, our risk assessment algorithm produces significant different metrics for the two systems: 0.98 vs. 0.73; the difference between the two values is more consistent with the intuitive assessment provided by experienced system administrators. Moreover, the number of applications that enable an attacker to compromise the system also became obvious in the new attack-graph model. The results of this grouping are shown in Figure 8.

However, the new metric numbers would still imply that both hosts were at high risk and the vulnerabilities should be addressed. This was not adopted by the system administrator, after looking at the applications that had the vulnerabilities — most of them existed in client applications rarely used on the servers. Thus the likelihood a user will launch one of those vulnerable programs was very low. In our evaluation we used a fixed value (0.8) to represent this likelihood, resulting in the high metric values. This indicates that to obtain more realistic security metrics, we need to assign this input parameter based on the knowledge of whether and how often a client application is used, which we leave as future work.

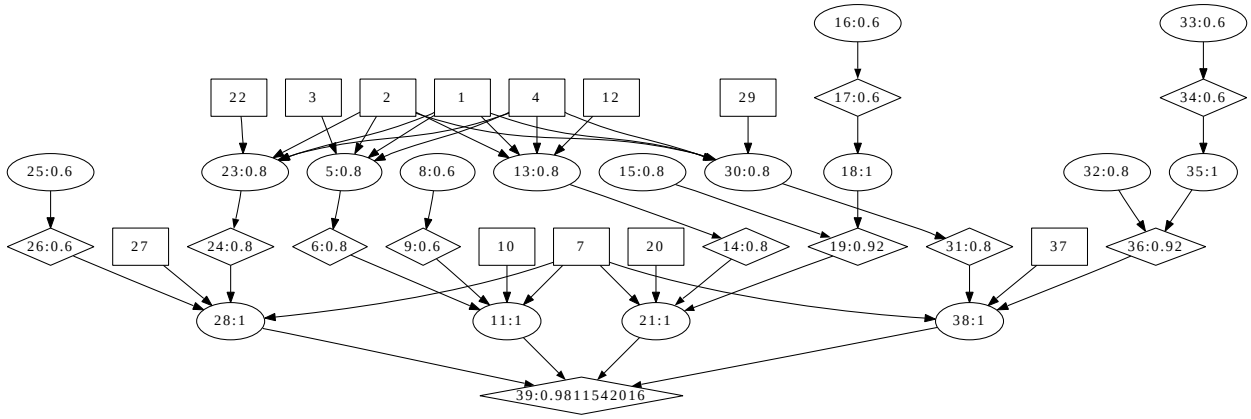
We would like to stress that *the reason we can refine our metric model and interpret the results based on such empirical observations is largely due to the fact that the metric calculation is sound.* This ensures that when we get a non-intuitive result, we can easily trace it back to the root cause, without having to wonder whether it was due to errors introduced in the calculation.

5.3 Testing scalability

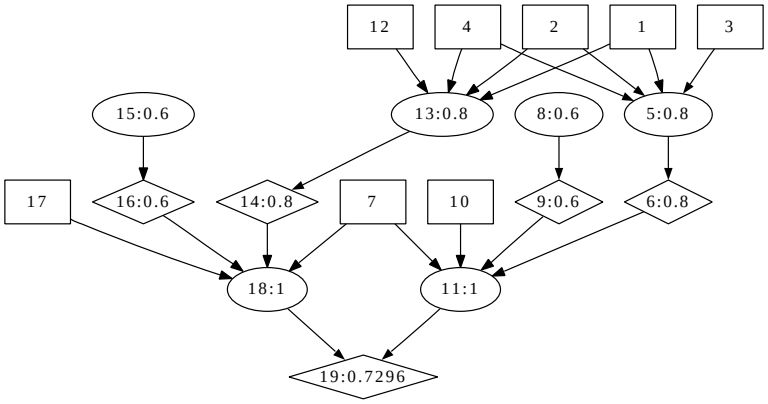
Table 2: **Network Reachability**

source	*destination*	*protocol*	*port*
Webservers	Database Servers	tcp	3306
Internet	Webservers	tcp	80
Workstations1	Internet	*	*
Workstations1	Fileservers	nfs	
Workstations2	Internet	*	*
Workstations2	Historians	*	*
Workstations2	Fileservers	nfs	
Workstations2	Mailservers	tcp	25
Mailservers	Internet	tcp	25

In order to test the scalability of our approach, we constructed several testing models based on networks of varying sizes and complexity, created MulVAL input files representing each network, and evaluated them with the current implementation of our algorithm. Figure 9 indicates the general network topology of the hypothetical network configurations used in our testing. The reachability information between various groups of machines is given in Table 2 (* is wild card). The vulnerability information can be found in Table 3. We performed our tests based on abstracted network models. In an abstracted model, each host represents a group of hosts having the same network reachability and similar configuration features (*e.g.*, they may be under the same software package management server). Therefore, every machine in Table 2 and Figure 9 represents a group of hosts. The run-time for metric computation of this network model was extremely short (less than a second).



(a)



(b)

Figure 8: Attack graphs with modeling artifacts for capturing hidden correlations

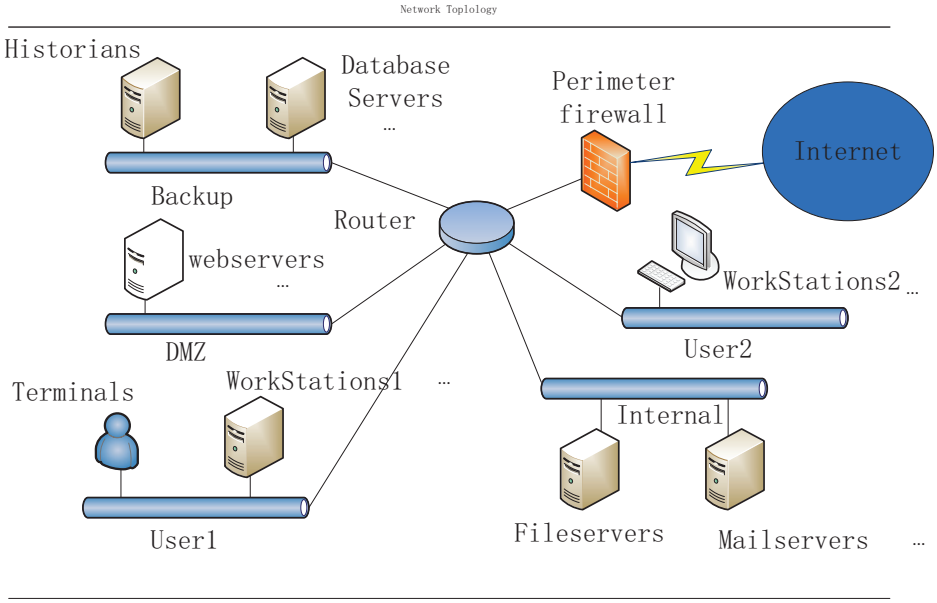


Figure 9: General network topology

Table 3: **Host vulnerabilities**

Host	# of vulnerabilities		
	Local	Remote Client	Remote Server
Webservers	0	1	1
Database servers	0	0	1
workstations1	0	1	1
Fileservers	1	0	1
Workstations2	0	1	1
Historians	0	0	1
Mailservers	0	0	1

To further test the limit of the algorithm's scalability, we picked 10 and 100 as the number of host groups in each subnet, and every machine can reach another subnet. Each time, we ensure that the host group with the deepest attack path (the one having the largest number of inter-subnet hops from the initial location of the attacker) is in the set of attack goals. The deepest goal will take the longest running time on the metric calculation. We also added 10 vulnerabilities per host with all three types (local, remote client, and remote server). The running time of the algorithm for these scenarios is shown in Table 4.

Table 4: **Scalability of Risk Assessment**

Num of host groups per subnet \ depth of the deepest inter-subnet attack hops	1	2	3
10	2s	2s	3s
100	1m38s	16m14s	46m36s

The limiting factors in the current algorithm and implementation are the size of the d-separating set (the number of nodes which must be marginalized in calculating conditional probability values) and the number of paths that must be considered in the calculation of each multi-predecessor node within a cycle. As either of these increases, the number of recursive calls made by the algorithm increases, and the evaluation time grows correspondingly. In the worst case, the computational increase could be exponential. However, as Table 4 shows, for realistic network settings, our algorithm can finish metric calculation for sufficiently large network configurations. The biggest case in the configuration consists of 100 host groups per subnet, 3 inter-subnet attack steps in the longest attack path, and 10 vulnerabilities of all three types per host. We believe this is a reasonable estimation on the large cases the tool will need to handle in reality. It is believed that most enterprise intrusions will take no more than three inter-subnet steps. After grouping and abstraction, 100 host groups per subnet and 10 vulnerabilities with all types per host represents a significantly large scenario for risk assessment analysis. For the worst-case scenario, our implementation of the algorithm can finish computation in less than an hour.

6 Conclusion

We have presented an approach to aggregating vulnerability metrics in an enterprise network through attack graphs. Our approach is sound in that, given component metrics which characterize the likelihood that individual vulnerabilities can be successfully exploited, the model computes a numeric value representing the cumulative likelihood for an attacker to succeed in gaining a specific privilege or carrying out an attack in the network. Our method handles both cyclic and shared dependencies in attack graphs correctly, surpassing

previous efforts on this problem. Preliminary testing results show the effectiveness and practicality of the approach and how it can be used to help system administrators decide between risk mitigation options.

References

[1] Ehab Al-Shaer, Latif Khan, and M. Salim Ahmed. A comprehensive objective network security metric framework for proactive security configuration. In *ACM Cyber Security and Information Intelligence Research Workshop*, 2008.

[2] Paul Ammann, Duminda Wijesekera, and Saket Kaushik. Scalable, graph-based network vulnerability analysis. In *Proceedings of 9th ACM Conference on Computer and Communications Security*, Washington, DC, November 2002.

[3] Zahid Anwar, Ravinder Shankesi, and Roy H. Campbell. Automatic security assessment of critical cyber-infrastructures. In *Proceedings of the 38th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, July 2008.

[4] Davide Balzarotti, Mattia Monga, and Sabrina Sicari. Assessing the risk of using vulnerable components. In *Proceedings of the 2nd ACM workshop on Quality of protection*, 2005.

[5] Steven Bellovin. On the brittleness of software and the infeasibility of security metrics. *IEEE Security & Privacy*, 2006.

[6] Marc Dacier, Yves Deswarte, and Mohamed Kaâniche. Models and tools for quantitative assessment of operational security. In *IFIP SEC*, 1996.

[7] J. Dawkins and J. Hale. A systematic approach to multi-stage network attack analysis. In *Proceedings of Second IEEE International Information Assurance Workshop*, pages 48 – 56, April 2004.

[8] Rinku Dewri, Nayot Poolsappasit, Indrajit Ray, and Darrell Whitley. Optimal security hardening using multi-objective optimization on attack tree models of networks. In *14th ACM Conference on Computer and Communications Security (CCS)*, 2007.

[9] Marcel Frigault and Lingyu Wang. Measuring network security using Bayesian network-based attack graphs. In *Proceedings of the 3rd IEEE International Workshop on Security, Trust, and Privacy for Software Applications (STPSA'08)*, 2008.

[10] Marcel Frigault, Lingyu Wang, Anoop Singhal, and Sushil Jajodia. Measuring network security using dynamic Bayesian network. In *Proceedings of the 4th ACM workshop on Quality of protection*, 2008.

[11] John Homer and Xinming Ou. SAT-solving approaches to context-aware enterprise network security management. *IEEE JSAC Special Issue on Network Infrastructure Configuration*, 2009.

[12] Nwokedi Idika and Bharat Bhargava. Extending attack graph-based security metrics and aggregating their application. *IEEE Transactions on Dependable and Secure Computing*, 9(1), 2012.

[13] Kyle Ingols, Matthew Chu, Richard Lippmann, Seth Webster, and Stephen Boyer. Modeling modern network attacks and countermeasures using attack graphs. In *25th Annual Computer Security Applications Conference (ACSAC)*, 2009.

[14] Kyle Ingols, Richard Lippmann, and Keith Piwowarski. Practical attack graph generation for network defense. In *22nd Annual Computer Security Applications Conference (ACSAC)*, Miami Beach, Florida, December 2006.

[32] Xinming Ou, Sudhakar Govindavajhala, and Andrew W. Appel. MulVAL: A logic-based network security analyzer. In *14th USENIX Security Symposium*, 2005.

[33] Joseph Pamula, Sushil Jajodia, Paul Ammann, and Vipin Swarup. A weakest-adversary security metric for network configuration security analysis. In *Proceedings of the 2nd ACM workshop on Quality of protection*, 2006.

[34] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufman, 1988.

[35] Cynthia Phillips and Laura Painton Swiler. A graph-based system for network-vulnerability analysis. In *NSPW '98: Proceedings of the 1998 workshop on New security paradigms*, pages 71–79. ACM Press, 1998.

[36] Stephen Quinn, David Waltermire, Christopher Johnson, Karen Scarfone, and John Banghart. *The Technical Specification for the Security Content Automation Protocol (SCAP): SCAP Version 1.0*. The National Institute of Standards and Technology Special Publication 800-126, 2009.

[37] Diptikalyan Saha. Extending logical attack graphs for efficient vulnerability analysis. In *Proceedings of the 15th ACM conference on Computer and Communications Security (CCS)*, 2008.

[38] Reginald Sawilla and Craig Burrell. Course of action recommendations for practical network defence. Technical Report TM-2009-130, Defence Research and Development Canada, 2009.

[39] Reginald Sawilla and Xinming Ou. Identifying critical attack assets in dependency attack graphs. In *13th European Symposium on Research in Computer Security (ESORICS)*, Malaga, Spain, October 2008.

[40] Oleg Sheyner. *Scenario Graphs and Attack Graphs*. PhD thesis, Carnegie Mellon, April 2004.

[41] Oleg Sheyner, Joshua Haines, Somesh Jha, Richard Lippmann, and Jeannette M. Wing. Automated generation and analysis of attack graphs. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, pages 254–265, 2002.

[42] Laura P. Swiler, Cynthia Phillips, David Ellis, and Stefan Chakerian. Computer-attack graph generation tool. In *DARPA Information Survivability Conference and Exposition (DISCEX II'01)*, volume 2, June 2001.

[43] Robert Tarjan. Depth-first search and linear graph algorithms. *SIAM Journal on Computing*, 1(2):146–160, 1972.

[44] T. Tidwell, R. Larson, K. Fitch, and J. Hale. Modeling Internet attacks. In *Proceedings of the 2001 IEEE Workshop on Information Assurance and Security*, West Point, NY, June 2001.

[45] Lingyu Wang, Tania Islam, Tao Long, Anoop Singhal, and Sushil Jajodia. An attack graph-based probabilistic security metric. In *Proceedings of The 22nd Annual IFIP WG 11.3 Working Conference on Data and Applications Security (DBSEC'08)*, 2008.

[46] Lingyu Wang, Steven Noel, and Sushil Jajodia. Minimum-cost network hardening using attack graphs. *Computer Communications*, 29:3812–3824, November 2006.

[47] Lingyu Wang, Anoop Singhal, and Sushil Jajodia. Measuring network security using attack graphs. In *Third Workshop on Quality of Protection (QoP)*, 2007.

Appendices

A Sample Symbolic Computations

A.1 Computation for Acyclic Graph

We will now work through the sample graph shown in Figure 3, demonstrating our approach and showing its effectiveness at recognizing and correctly handling shared dependencies within the graph. This graph is acyclic, so we can recursively calculate the probability value for each node, utilizing previously calculated individual probability values and computing joint probabilities only as needed. A table showing all calculated values will be included at the end of this example.

We begin with the root node, P_0. The probability that node P_0 is true is G_V which is assumed to be 1. Thus $\phi(P_0) = 1$. As the root node, P_0 has no preceding nodes, so $\chi(P_0) = \{\ \}$ and $\delta(P_0) = \{\ \}$.

Now that we have calculated $\phi(P_0)$, we can calculate for either A_1 or A_6. Let us next calculate for A_6. A_6 has exactly one predecessor, P_0. So, $\phi(A_6) = G_M[A_6] \cdot \phi(P_0) = e_6 \cdot 1 = e_6$; furthermore, $\chi(A_6) = \delta(A_6) = \{P_0\}$.

We cannot yet evaluate node P_2, because not all of its predecessors have been evaluated. We will return, then, to evaluate node A_1. Similar to the calculation for A_6, $\phi(A_1) = G_M[A_1] \cdot \phi(P_0) = e_1$; $\chi(A_1) = \delta(A_1) = \{P_0\}$.

We can now evaluate node P_1. Node P_1 also has only one predecessor. Thus, $\phi(P_1) = 1 - \phi(\bar{A}_1) = \phi(A_1) = e_1$; $\chi(P_1) = \delta(P_1) = \{P_0\}$.

From this point, we could evaluate either A_2 or A_3. Let us next calculate for A_2. $\phi(A_2) = G_M[A_2] \cdot \phi(P_1) = e_2 \cdot e_1 = e_1e_2$; $\chi(A_2) = \delta(A_2) = \{P_0, P_1\}$.

Now both predecessors to P_2 have been solved and we can calculate for this node. Proposition 4.1 specifies the calculation for a privilege node with multiple predecessors. So, $\phi(P_2) = 1 - \phi(\{\overline{A}_2, \overline{A}_6\})$. In previous cases with single predecessors, we already knew the probability of the predecessor, but in this case we do not yet know the joint probability of $\phi(\{\overline{A}_2, \overline{A}_6\})$ and so must solve for it. To calculate $\phi(\{\overline{A}_2, \overline{A}_6\})$, we must find a d-separating set for these two nodes so that we can utilize Theorem 3.1. One such set can be found by taking the intersection of the χ-sets for these nodes, so that $D = \chi(A_2) \cap \chi(A_6) = \{P_0\}$. D contains all branch nodes that diverge to paths leading to A_2 and A_6, which should be sufficient to d-separate the nodes (Definition 1). Using the set D, we can now solve for $\phi(\{\overline{A}_2, \overline{A}_6\})$:

$$
\begin{aligned}
&= \sum_{P_0} \ (\{P_0\}, \overline{A}_2, \overline{A}_6)\phi(\{P_0\}) \\
&= \sum_{P_0} \ (\{P_0\}, \overline{A}_2) \ \ (\{P_0\}, \overline{A}_6)\phi(\{P_0\}) \\
&= \ \ (\{P_0\}, \overline{A}_2) \ \ (\{P_0\}, \overline{A}_6)\phi(\{P_0\}) + \\
&\quad\ (\{\overline{P}_0\}, \overline{A}_2) \ \ (\{\overline{P}_0\}, \overline{A}_6)\phi(\{\overline{P}_0\}) \\
&= (1 - e_1e_2)(1 - e_6)(1) + (1)(1)(0) \\
&= 1 - e_1e_2 - e_6 + e_1e_2e_6
\end{aligned}
$$

Then, $\phi(P_2) = 1 - \phi(\{\overline{A}_2, \overline{A}_6\}) = 1 - (1 - e_1e_2 - e_6 + e_1e_2e_6) = e_1e_2 + e_6 - e_1e_2e_6$. Also, $\chi(P_2) = \chi(A_2) \cup \chi(A_6) = \{P_0, P_1\}$, and $\delta(P_2) = \delta(A_2) \cap \delta(A_6) = \{P_0\}$.

Nodes A_3, A_4, A_5, P_3 are calculated very similarly to nodes we've already seen here, so we will skip over the details of these. The resulting values are in Table 5.

Finally, we evaluate for node P_4, a privilege node with multiple predecessors, so again we will apply Proposition 4.1: $\phi(P_4) = 1 - \phi(\{\overline{A}_4, \overline{A}_5\})$. The d-separating set for $\{A_4, A_5\}$ is $D = \chi(A_4) \cap \chi(A_5) = \{P_0, P_1\}$. Using the set D, we can now solve for $\phi(\{\overline{A}_4, \overline{A}_5\})$:

$$
\begin{aligned}
&= \sum_{P_0,P_1} \quad (\{P_0,P_1\},\overline{A}_4) \;\; (\{P_0,P_1\},\overline{A}_5)\phi(\{P_0,P_1\}) \\
&= \quad (\{P_0,P_1\},\overline{A}_4) \;\; (\{P_0,P_1\},\overline{A}_5)\phi(\{P_0,P_1\}) + \\
&\quad\; (\{P_0,\overline{P}_1\},\overline{A}_4) \;\; (\{P_0,\overline{P}_1\},\overline{A}_5)\phi(\{P_0,\overline{P}_1\}) + \\
&\quad\; (\{\overline{P}_0,P_1\},\overline{A}_4) \;\; (\{\overline{P}_0,P_1\},\overline{A}_5)\phi(\{\overline{P}_0,P_1\}) + \\
&\quad\; (\{\overline{P}_0,\overline{P}_1\},\overline{A}_4) \;\; (\{\overline{P}_0,\overline{P}_1\},\overline{A}_5)\phi(\{\overline{P}_0,\overline{P}_1\}) \\
&= (1 - e_4(e_2 + e_6 - e_2e_6))(1 - e_3e_5)(e_1) + \\
&\quad (1 - e_4e_6)(1)(1 - e_1) + (1)(1)(0) + (1)(1)(0) \\
&= 1 - e_1e_2e_4 + e_1e_2e_4e_6 - \\
&\quad e_1e_3e_5 - e_4e_6 + e_1e_3e_4e_5(e_2 + e_6 - e_2e_6)
\end{aligned}
$$

Then, $\phi(P_4) = 1 - \phi(\overline{A}_4, \overline{A}_5) = e_4e_6 + e_1(e_2e_4 + e_3e_5) - e_1e_2e_4e_6 - e_1e_3e_4e_5(e_2 + e_6 - e_2e_6)$. Also, $\chi(P_4) = \chi(A_4) \cup \chi(A_5) = \{P_0, P_1\}$, and $\delta(P_2) = \delta(A_2) \cap \delta(A_6) = \{P_0\}$.

We have now solved for the probability of each node in the graph. The computed ϕ values for individual nodes are shown in Table 5, together with the χ and δ sets for each node. In our implementation, joint and conditional probability values are calculated only as needed, to reduce the amount of computation performed. We also apply dynamic programming techniques to cache the calculated values to avoid repeating the same computation.

Table 5: Risk assessment calculations for Figure 3

N	$\phi(N)$	$\delta(N)$	$\chi(N)$
P_0	1	$\{\}\leftarrow$	$\{\}\leftarrow$
P_1	e_1	$\{P_0\}\leftarrow$	$\{P_0\}\leftarrow$
P_2	$(e_1e_2 + e_6 - e_1e_2e_6)$	$\{P_0\}\leftarrow$	$\{P_0, P_1\}\leftarrow$
P_3	e_1e_3	$\{P_0, P_1\}\leftarrow$	$\{P_0, P_1\}\leftarrow$
P_4	$(e_4e_6 + e_1(e_2e_4 + e_3e_5) - e_1e_2e_4e_6 - e_1e_3e_4e_5(e_2 + e_6 - e_2e_6))$	$\{P_0\}\leftarrow$	$\{P_0, P_1\}\leftarrow$
A_1	e_1	$\{P_0\}\leftarrow$	$\{P_0\}\leftarrow$
A_2	e_1e_2	$\{P_0\}\leftarrow$	$\{P_0, P_1\}\leftarrow$
A_3	e_1e_3	$\{P_0, P_1\}\leftarrow$	$\{P_0, P_1\}\leftarrow$
A_4	$e_4(e_1e_2 + e_6 - e_1e_2e_6)$	$\{P_0\}\leftarrow$	$\{P_0, P_1\}\leftarrow$
A_5	$e_1e_3e_5$	$\{P_0, P_1\}\leftarrow$	$\{P_0, P_1\}\leftarrow$
A_6	e_1	$\{P_0\}\leftarrow$	$\{P_0\}\leftarrow$

A.2 Computation for Cyclic Graph

In the previous example, we showed that the probabilities for graph nodes depend on the probabilities of their predecessors, so a recursive approach can be employed for this calculation. Within a cycle, however, recursing backward through predecessor sets will create an infinite loop. It is clear that a different approach is needed for cyclic nodes.

We will now work through the sample graph shown in Figure 4, demonstrating our approach and showing its effectiveness at recognizing and correctly handling cycles within the graph. A table showing all calculated

values will be included at the end of this example.

Nodes P_0, A_1, P_1, A_2, A_3 will be calculated very much as demonstrated in Section A.1 and so we will not go through the detailed calculations for those nodes. Once these have been calculated, however, the remaining graph nodes comprise a cycle and therefore must be handled differently.

Table 6: Risk assessment calculations for Figure 4

N	$\phi(N)$	$\delta(N)$	$\chi(N)$
P_0	1	$\{\}\leftarrow$	$\{\}\leftarrow$
P_1	e_1	$\{\}\leftarrow$	$\{\}\leftarrow$
P_2	$e_1e_2 + e_1e_3e_4 - e_1e_2e_3e_4$	$-\leftarrow$	$-\leftarrow$
P_3	$e_1e_3 + e_1e_2e_5 - e_1e_2e_3e_5$	$-\leftarrow$	$-\leftarrow$
A_1	e_1	$\{\}\leftarrow$	$\{\}\leftarrow$
A_2	e_1e_2	$\{P_1\}\leftarrow$	$\{P_1\}\leftarrow$
A_3	e_1e_3	$\{P_1\}\leftarrow$	$\{P_1\}\leftarrow$
A_4	$e_4(e_1e_3 + e_1e_2e_5 - e_1e_2e_3e_5)$	$-\leftarrow$	$-\leftarrow$
A_5	$e_5(e_1e_2 + e_1e_3e_4 - e_1e_2e_3e_4)$	$-\leftarrow$	$-\leftarrow$

First, we will trace all acyclic paths through the cycle, to determine all valid ways that these nodes can be reached. This trace essentially performs a logical unfolding of the graph, marking unique passes through each node. The acyclic paths through this cycle are:

$$P_{2A} = \{A_2\} \qquad P_{3A} = \{A_2, P_{2A}, A_4\}$$
$$P_{3B} = \{A_3\} \qquad P_{2B} = \{A_3, P_{3B}, A_5\}$$

There are two unique instances of node P_2 in this logical unfolding of the graph, P_{2A} and P_{2B}. The probability that node P_2 is true will equal the probability that at least one of these instances is true, or $\phi(P_2) = 1 - \phi(\overline{P}_{2A}, \overline{P}_{2B})$.

To calculate $\phi(\overline{P}_{2A}, \overline{P}_{2B})$, we must calculate the joint probability of the set of entry nodes $\{A_2, A_3\}$ and we will also need to identify a d-separating set $D\not$within the cycle, to ensure that the instances of P_2 are conditionally independent. A cyclic d-separating set can be found by intersecting the sets of possible paths leading to the node instances and identifying common attack-step nodes within the cycle. In this case, a cyclic d-separating set is not needed for nodes P_{2A} and P_{2B}; because the cycle is so small, these partial paths are already conditionally independent, given the entry points into the cycle. The formula of computation is shown below.

$$
\begin{aligned}
& \phi(\overline{P}_{2A}, \overline{P}_{2B}) \\
= & \sum_{A_2, A_3} \phi(\{A_2, A_3\}) \ \ (\{A_2, A_3\}, \overline{P}_{2A}) \ \ (\{A_2, A_3\}, \overline{P}_{2B}) \\
= & \ \phi(\{A_2, A_3\}) \ \ (\{A_2, A_3\}, \overline{P}_{2A}) \ \ (\{A_2, A_3\}, \overline{P}_{2B}) + \\
& \ \phi(\{A_2, \overline{A}_3\}) \ \ (\{A_2, \overline{A}_3\}, \overline{P}_{2A}) \ \ (\{A_2, \overline{A}_3\}, \overline{P}_{2B}) + \\
& \ \phi(\{\overline{A}_2, A_3\}) \ \ (\{\overline{A}_2, A_3\}, \overline{P}_{2A}) \ \ (\{\overline{A}_2, A_3\}, \overline{P}_{2B}) + \\
& \ \phi(\{\overline{A}_2, \overline{A}_3\}) \ \ (\{\overline{A}_2, \overline{A}_3\}, \overline{P}_{2A}) \ \ (\{\overline{A}_2, \overline{A}_3\}, \overline{P}_{2B}) \\
= & \ (0) + (0) + (e_1 e_3 (1 - e_2))(1)(1 - e_4) + \\
& \ (e_1 (1 - e_2)(1 - e_3) + 1 - e_1)(1)(1) \\
= & \ 1 - e_1 e_2 - e_1 e_3 e_4 + e_1 e_2 e_3 e_4
\end{aligned}
$$

So, $\phi(P_2) = 1 - (1 - e_1e_2 - e_1e_3e_4 + e_1e_2e_3e_4) = e_1e_2 + e_1e_3e_4 - e_1e_2e_3e_4$. By a similar calculation, $\phi(P_3) = e_1e_3 + e_1e_2e_5 - e_1e_2e_3e_5$. Once these have been solved, it is easy to see that $\phi(A_4) = G_M[A_4] \cdot \phi(P_3) = e_4(e_1e_3 + e_1e_2e_5 - e_1e_2e_3e_5)$ and $\phi(A_5) = G_M[A_5] \cdot \phi(P_2) = e_5(e_1e_2 + e_1e_3e_4 - e_1e_2e_3e_4)$.

The full results are shown in Table 6. Nodes P_2, P_3, A_4, A_5 do not have $\chi\psi$ or $\delta\psi$ sets, because these values are not used for evaluation within a cycle. Once a cyclic node set is calculated, however, successor nodes can include a virtual node representing the cycle in their $\chi\psi$ and $\delta\psi$ sets.

Responses to Reviewers' Comments

REVIEWER 3

The cited work on attack graphs should probably include more recent work from MIT/LL. E.G. "Modeling Modern Network Attacks and Countermeasures Using Attack Graphs" Dec. 2009.

We have added a citation for this work on attack graphs.

When the attack graph semantics are introduced on page 5 with figure 2 there is no explanation for the number following the colon in the graph nodes. This is explained later, but there should be a brief explanation here, since it is in the figure.

We added an explanatory sentence and reorganized some text within the subsection "Attack Graph Semantics" to explain this more clearly

Page 6. Component metrics: "The input to the metric model is the component metrics" Should metrics be singular? (or change "is" to "are"?)

This has been corrected (changed to "are")

Page 7 mentions the determination of a minimum d-separated set. In Bayesian Networks, this is known as the "Markov Blanket".

We added a reference to Markov Blanket after Definition 1.

Also, the statement: "The metric aggregation problem on attack graphs is a more specific problem than generic Bayesian Networks" implies that this approach is a specific application of Bayesian Networks. Is it? On page 21, there is some discussion of the relationship to Bayesian Networks and this approach. But it is not apparent whether it would be correct to say this is a specific application of Bayesian Networks. If that is not a correct statement, what is it about this approach that is contrary to the rules of Bayesian Networks.

We added a section (3.3.3) to clarify the relationship.

For example, it seems to me that once cycles have been removed to form a DAG, the same rules for calculating the probability of a given event would be used that is used for a BN. It isn't clear exactly how this approach relates to BN: application? refinement? Departure?

The difficulty is that one cannot remove cycles from an attack graph without exponentially blowing up the resulting DAG's size (discussed in section 3.3.2). To have a more efficient calculation, we chose to design a customized algorithm to aggregate probability metrics on attack graphs based on Bayesian rules. We hope the newly added section 3.3.3 and some revision of the texts in the previous sections help to clarify these issues.

Section 3.1: To improve readability I suggest changing "Function 1" to "Function ?" and similarly for functions 2-4.

Unfortunately, this suggestion appears to have been obscured when posted online, so we are unable to respond to it and the function numbering remains unchanged.

Page 16. I suggest Figure 6a, be expanded vertically somewhat. It is very difficult to recognize it as a graph in that form.

The figure has been expanded vertically and is now more recognizable as a graph (although one with many nodes)

Page 16 first paragraph: "there is quite amount of residual risk" should be reworded.

This has been reworded as, "there is a significant amount of residual risk"

Page 16: The following statement: "A machine with 10 vulnerabilities in a single application has a lower risk than a machine with 10 vulnerabilities in 10 different applications." may be true but I'm not convinced. I claim it is not necessarily true. I'd like to see a better argument.

An explanation of the reasoning behind this statement is given in the paper. We make this statement on the strength of empirical data gathered from a variety of sources. We acknowledge that the statement is debatable; if future experiences dispute this perspective, our approach to risk assessment in such situations can be easily adapted.

General comment: This paper provides an interesting approach that could be quite useful for decision making IF credible numbers can be obtained for the component metrics (probability a given vulnerability will be exploited) Unfortunately it is very difficult to obtain credible numbers of that type. That is my main reservation about the significance of the paper. I think it is valuable to have the theoretical foundation this paper provides.

We believe that there is value in setting forth a well-defined theoretical framework to risk assessment. We agree that component metrics can be difficult to verify, but it is important to take steps toward a reliable measurement. We believe that a well-defined approach can itself be used to calibrate the input parameters provided, based on results obtained over time in realistic scenarios.

REVIEWER 5

This paper presents a method to aggregate vulnerability metrics so that a more informed decision could be made for security hardening of enterprise systems unlike using the standard path-based metrics from attack graphs. The main contribution of the paper is the handling of cycles in attack graphs, and the dependency between the various paths in attack graphs. The vulnerability metric aggregation that is complicated by shared dependency between paths is handled by a simple application of Bayes Theorem.

While the paper has several innovative aspects, it suffers from poor presentation of the material. Starting with the title, the paper makes several awkward remarks and self-congratulatory statements. Some facts are presented in a very confusing way. The literature review could also be improved. The paper needs a complete rewriting to bring it to the level of an archival publication. Below are some specific comments.

1) The title is highly inappropriate. Both the title and abstract overuse the word "sound." A more appropriate title could be "Aggregating Attack Graph Based Vulnerability Metrics in Enterprise Networks." The usage of the word "sound" to describe the authors' approach implies that all existing metrics are ad hoc and useless (a strong statement). A metric needs to be sound to begin with. Otherwise, it is useless or misleading. A model or a metric that is not sound is not a model or a metric at all. So, trying to sell the contribution of the paper as a "sound approach" is an unnecessary stretch for a technical paper.

The paper title has been altered to, "Aggregating Vulnerability Metrics in Enterprise Networks using Attack Graphs". We agree that soundness is the basic requirement of any metric model, and we recognize that there are significant challenges to create a sound probabilistic metric model for the purpose of enter-

prise network security assessment. Our main contribution is to tackle these challenges, to create a metric model with well-defined probabilistic semantics, and to design a set of algorithms that calculate the metrics correctly based on the semantics. We have revised/re-organized multiple parts of the introduction section to better explain this contribution.

2) The writing at many places is ambiguous, speculative and awkward. See the following instances. In page 2, "To accurately reflect security risks vulnerabilities bring to an enterprise network, both measurement of individual vulnerabilities' properties and the context within which they appear must be taken into account." What do you mean by context here? Does your aggregation technique take this context information into consideration more than the existing attack graph based evaluation systems? To a large extent, attack graphs do consider context by way of incorporating the "system configuration" information and the "attacker skills." Therefore, some rewriting of this material is necessary.

The purpose of this statement is to explain the limitation of existing vulnerability metrics (such as CVSS), and this leads into the next paragraph that explains why combining attack graphs and individual vulnerability metrics is a good approach. Our approach actually brings in the context information through attack graphs, so attack graph is a critical component in our approach, rather than an alternative.

We did revise multiple other parts of the introduction to better explain the motivation of our approach. We hope these revisions help clarify some of the issues mentioned in the comments.

On page 3, Sec. 1.1.1, the statement "we believe there is value in capturing such vague assessment through a sound model" is incorrect. The classification of risk as high, medium and low is not vague. This is a typical way of doing risk assessment and is quite meaningful. It may be more instructive to say "....quantification of risk as illustrated in this paper will provide more clarity to the risk assessment."

We accept the standard high/med/low classifications and did not intend to imply that they are not meaningful, but were trying to address a perceived imprecision. Section 1.1.1 has been rewritten to emphasize that our contribution is a clear semantic model for aggregating over vulnerability metrics, both utilizing existing metrics and providing a feedback loop for refinement over time.

The final statement at the end of Section 4.3 is highly speculative. Without a detailed complexity analysis, how could one make such a statement? You are better off not making that statement at all.

This statement has been removed, as suggested.

3) On page 3, "attack exposure" is essentially the "attack surface" characterized in prior research in the area of computer security. Using known terminology with a proper reference will make the writing tight.

Text altered to the term "attack surface", with appropriate references

4) In Section 1.2, the discussion of cycles in attack graphs "such attack paths should be excluded from the calculations" is confusing and is also incorrect. Cycles in attack graphs are inevitable. For example, an adversary may take a "walk" in the graph to acquire "information" (come back to where he started and proceed further with the acquired information and then exploit a vulnerability using the knowledge so gained). This kind of analysis will lead to a cyclic representation. So, the current discussion in Section 1.2 is misleading. As can be seen in later sections, you are not excluding cycles. You treat them in a different way (using the concept of d-separation, for instance). So, some rewriting is necessary to remove confusion.

Section 1.2 has been altered to avoid this confusion, including the removal of the specific phrase "such attack paths should be excluded from the calculations". We also revised section 3.3.2 to better explain why

cycles are problematic in metric calculation.

5) Figure 2 is not a full attack graph for the given physical network. This must be stated early on (later on you say that the configuration nodes are not included).

This is now more clearly stated, both in the figure label and the text.

6) Why all the algorithms are pushed to the appendix? Why aren't they important? I suggest that you move back the main algorithm to the body of the paper and perhaps push the secondary algorithms to the appendix. In order to conserve space, you may move some of the examples (call them concept illustrations) which show the step-by-step derivations to the appendix.

As suggested, the algorithms now appear in the body of the paper and some sample computations have been moved to the appendix.

7) The example in Section 3.4 is actually not just an example. It has some theory on how to handle cycles, a key concept of the paper. I suggest that you move the theory part to Section 2.3 and beef up this section.

Section 2.3 (now 3.3) is meant as a high-level overview, addressing the need to correctly handle cycles. Section 4.3 now uses this example to further explain how cycles are handled in our algorithms.

8) Finally, I would like to bring to the authors' attention the following work by a group of researchers - Nwokedi Idika, Bharat Bhargava, "Extending Attack Graph-Based Security Metrics and Aggregating Their Application," IEEE Transactions on Dependable and Secure Computing, vol. 99, no. 1, pp. , , 5555. This paper is in pre-print stage and can be obtained from IEEE digital library. Though the authors have no way of knowing someone else's unpublished work, the fact is that this paper by Idika and Bhargava will be appearing in print soon and is highly relevant to the paper by the current authors. Because of the closeness of these two papers, I strongly suggest that the authors include a reference to this upcoming paper (if it appears in print while revising your paper) and situate your work. A comparison of your metrics to the metrics of Idika and Bhargava, if possible will add value to your work.

We have added a paragraph in the Related Works section addressing this work, and briefly comparing it to our own approach.

REVIEWER 6

There are two papers here. One on the theory for attack graphs, and one on experimental results.

The theory part is pretty good, with the following changes.

1. in the abstract, quantify in some way how the new methods for accounting for non-independence improve the theory/model of attack graphs.

We have included in the introduction an intuitive example that demonstrates the significant difference in metric results introduced by non-independence. We hope this helps explain why accounting for such non-independence is important.

2. move the related work section to be section 2. readers like to know the full situating context of your work before they dive in to the details.

This section has been moved, as suggested.

3. soundness is a loaded word. When I hear it, I think: http://en.wikipedia.org/wiki/Soundness Please use a different word. Well-defined? Yes. I think what you're looking for is face validity. http://en.wikipedia.org/wiki/Face_validity

Soundness may not be the best word overall. We have rephrased this as "clear semantics" at several points in the paper, and we stress that the algorithm produces correct metric result with respect to the defined semantics.

4. Be clear about the accuracy or validity improvement over the current approach. can you say anything quantitative? like a formula derived from an abstract model? or even for a specific simple class of attack graphs? that would be very useful and insightful.

We have used a small simple attack graph (Figure 1) to illustrate the validity improvement (see comment 1 above).

For the Experimental part, as far as I can tell you didn't run experiments. You plugged in some data. What are the hypotheses that you're testing? Read Gauch on Scientific Method in Practice (9780521017084). Please rethink/rewrite and separate from the theory paper.

On the other hand, you could use the operational data to demonstrate how they quantitative benefits of your approach are expressed in the particular instance of your data. But that's not an experiment.

This section has been relabeled as "Evaluation Results"; at other points, "experiment" has been changed to "test" or "evaluation". This phrasing is more accurate.