Metrics Suite for Network Attack Graph Analytics

Steven Noel
The MITRE Corporation*
McLean, Virginia USA
+1 703.983.2468
snoel@gmu.edu

Sushil Jajodia
Center for Secure Information Systems
George Mason University
Fairfax, Virginia USA
+1 703.993.1653
jajodia@gmu.edu

ABSTRACT
We describe a suite of metrics for measuring network-wide cyber security risk based on a model of multi-step attack vulnerability (attack graphs). Our metrics are grouped into families, with family-level metrics combined into an overall metric for network vulnerability risk. The *Victimization* family measures risk in terms of key attributes of risk across all known network vulnerabilities. The *Size* family is an indication of the relative size of the attack graph. The *Containment* family measures risk in terms of minimizing vulnerability exposure across protection boundaries. The *Topology* family measures risk through graph theoretic properties (connectivity, cycles, and depth) of the attack graph. We display these metrics (at the individual, family, and overall levels) in interactive visualizations, showing multiple metrics trends over time.

Categories and Subject Descriptors
C.2.0 [**Computer-Communication Networks**]: General – *security and protection.*

General Terms
Management, Measurement, Security.

Keywords
Attack graphs, topological vulnerability analysis, security metrics.

1. INTRODUCTION
Modeling and analysis of network attack graphs has reached a fair level of maturity. Analysis tools such as Cauldron [1] are able to merge data from a variety of network data sources to build graphs of all known vulnerability paths through a network. Attack graphs provide a rich framework for new kinds of metrics for network attack risk. There is a critical need for such metrics, to summarize operational status at a glance, to compare security options, and to understand network health over time.

We describe a suite of metrics for measuring overall network security risk, based on a comprehensive model of multi-step attack vulnerability. Our metrics span different complementary dimensions of enterprise security. These metrics are grouped into families, which are combined into an overall risk metric for the network at a point in time, based on vulnerabilities and policies.

CISR '14, April 08–10, 2014, Oak Ridge, TTN, USA
ACM 978-1-4503-2812-8/14/04.
http://dx.doi.org/10.1145/2602087.2602117

2. SYSTEM ARCHITECTURE
Figure 1 depicts our system for computing security metrics from network attack graphs. This system leverages data sources that are commonly deployed within enterprise networks, such as vulnerability scanners and firewall configuration files.

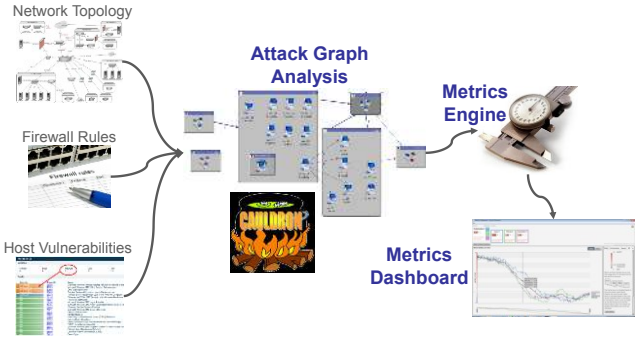

Figure 1. Attack graph metrics suite.

Cauldron builds a model of network attack vulnerability from various scan tools and other data sources. It correlates model cyber attack vulnerabilities and environmental metadata, and applies network access policy rules. Based on assumed threat sources and mission-critical assets to protect, Cauldron finds all known paths of vulnerability. The metrics engine then computes metrics from the attack graph, which we visualize over time.

3. ATTACK GRAPH METRICS
3.1 Metrics Families
The metrics engine computes individual metrics that each capture different aspects of overall security. We group related metrics into families, as shown in Figure 2.

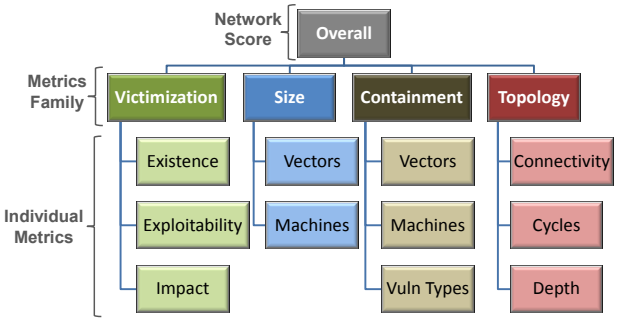

Figure 2. Attack graph metrics families.

We combine individual metrics into family scores, and then combine those into an overall network score. The metrics are mapped to a common scale of zero to 10 (least risk to most risk), as for the Common Vulnerability Scoring Sysem (CVSS) [3].

* The author's affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE's concurrence with, or support for, the positions,opinions or viewpoints expressed by the author. MITRE Approved for Public Release; Distribution Unlimited. 14-1377.

We treat the individual metrics as independent (orthogonal) components of a multi-dimensional vector. We then compute the Euclidean norm (magnitude) of the k-vector as the combined effect of k metrics (either individual or family).

3.2 Vicimization Family

Individual vulnerabilities and exposed services each have elements of risk. The *Victimization* family scores the entire network across these victimization dimensions.

The *Existence* metric is the relative number of network services that are vulnerable, on the standard scale of (0,10). In particular, for s_v vulnerable and s_n non-vulnerable services across the network, the Existence metric is $10\, s_v/(s_v + s_n)$.

The *Exploitability* metric is the average value of the CVSS Exploitability score (the relative ease of exploitation), averaged over all vulnerabilities over all hosts, on the scale of (0,10). For $\text{CVSS}_{\text{Exploitability}}(u_i)$ for vulnerability u_i and $|U|$ total vulnerabilies, the exploitability for the entire network is $\sum_i^{|U|} \text{CVSS}_{\text{Exploitability}}(u_i)/|U|$.

The *Impact* metric is the average value of the CVSS Impact score (relative impact of exploitation), taken over all vulnerabilities over all hosts, on the scale of (0,10). For $\text{CVSS}_{\text{Impact}}(u_i)$ for vulnerability u_i and $|U|$ total vulnerabilities, the impact for the entire network is $\sum_i^{|U|} \text{CVSS}_{\text{Impact}}(u_i)/|U|$.

3.3 Size Family

The size of an attack graph is a prime indication of risk. The larger the graph, the more ways you can be compromised. The *Size* family measures risk in terms of the attack graph size.

The *Attack Vectors* metric is the number of single-step attack vectors, relative to the total possible number for the network, on the scale of (0,10). As shown in Figure 3, we must consider two kinds of attack vectors: implicit (within protection domains) and explicit (across domains). Here, *protection domains* is a set of network machines that have unrestricted access to each others' vulnerabilities (a shaded box in the figure).

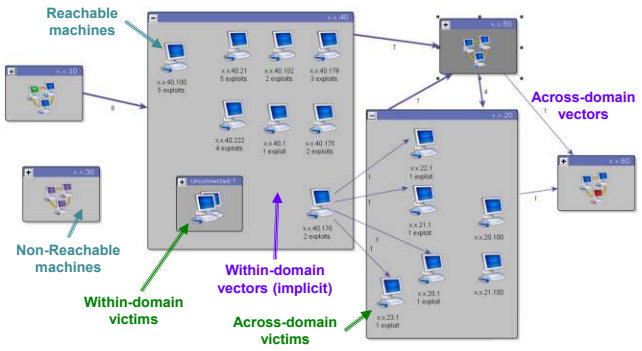

Figure 3. Components of an attack graph.

The total number of attack vectors is then the sum of the implicit and explicit attack vectors. To map this raw number of attack vectors to the scale (0,10), we must normalize by the total possible number of attack vectors, in terms of the number of open ports across all machines. The Attack Vectors metric is then $10\sqrt{v_a/v_p}$ for v_a attack vectors and v_p possible attack vectors.

The *Reachable Machines* metric is the number of machines in the attack graph, relative to the total number of machines in the network, on the scale of (0,10). As shown in Figure 3, we must consider the machines that are in the attack graph (reachable by an attacker through some number of attack steps) as well as machines that are in the network model but not in the attack graph.

That is, for r_i reachable machines in protection domain i, with d domains, the total number of reachable machines is $r = \sum_i^d r_i$. For n_i non-reachable machines (i.e., in the network but not in the attack graph), the total number of non-reachable machines is $n = \sum_i^d n_i$. The Reachable Machines metric is then $10\, r/(r + n)$.

3.4 Containment Family

Networks are generally administered in pieces (subnets, domains, etc.). Risk mitigation should aim to reduce attacks across such boundaries, to contain attacks. The *Containment* family measures risk in terms of the degree to with the attack graph contains attacks across such network protection domains.

The *Vectors Containment* metric is the number of attack vectors across protection domains, relative to the total number of attack vectors, on the scale of (0,10). As shown in Figure 3, the attack vectors across domains are explicit, and are counted across all domain pairs. The attack vectors within protection domains are implicit (all machine vulnerabilities are reachable within a domain). The Vectors Containment metric is then $10\, v_c/v_a$ for v_c attack vectors across domains and v_c total attack vectors.

The *Machines Containment* metric is the number of machines in the attack graph that are victims of attacks from other domains, relative to the total number of attack graph machines, on the scale of (0,10). As shown in Figure 3, the victim machines across domains are those machines that have no incoming incident edge in the domain-to-domain attack graph. The remaining machines are within-domain victims only. The Machines Containment metric is then $10\, m_a/(m_a + m_w)$ for m_a across-domain victim machines and m_w within-domain victim machines.

The *Vulnerability Types* metric is the number of unique vulnerability types that are victims of attacks from other domains, relative to the total number of vulnerability types across the entire attack graph, on the scale of (0,10). As shown in Figure 3, the across-domain vulnerability types are victimized across domains. The remaining vulnerability types are victimized within domains only. The idea is that multiple instances of the same vulnerability type are less costly to mitigate. The Vulnerability Types metric is then $10t_a/(t_a + t_w)$ for t_a across-domain vulnerability types and m_w within-domain vulnerability types.

3.5 Topology Family

Certain graph theoretic properties (i.e., connectivity, cycles, and depth) of an attack graph (at the domain-to-domain level) reflect how graph relationships enable network penetration. The *Topology* family measures risk in terms of these properties.

The *Connectivity* metric is the number of weakly connected components in the domain-level attack graph, relative to the best (most secure) and worst (least secure) cases possible, on the scale of (0,10). The intuition is that it is better to have an attack graph that is disconnected parts versus a connected whole.

To map the Connectivity metric to the $(0,10)$ scale, we need the largest and smallest possible values for weak connectivity (at the protection domain level). The worst case (least secure) is a single weakly connected component. The best case (most secure) is completely disconnected, i.e., d weakly connected components for d domains. These ranges of possible numbers of components

need to be mapped to the $(0,10)$ scale, consistent with the definition of zero as best case (most secure) and 10 as best case (least secure).

We seek a function that maps the best case (d components) to the number zero (most secure), and the worst case (one component) to the number 10 (least secure). This is accomplished by a linear transformation that shifts the number of weak components w to the left by one, divides by the range $d-1$, reverses the order by multiplying by negative one, and then shifts to the right by one. The resulting transformation maps the best case (d components) to zero and the worst case (one component) to 10. This yields the Connectivity metric $10\left(1-\frac{w-1}{d-1}\right)$.

Figure 4 shows an example computation of the Connectivity metric. In this example, there are three attack graphs, shown at the protection-domain level. Each attack graph has the same set of domains, but different sets of domain-to-domain edges, resulting in different numbers of weakly connected components.

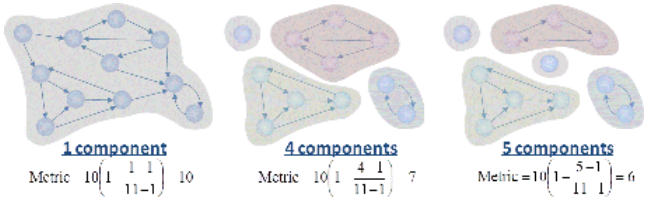

Figure 4. Example of Connectivity metric.

As shown in the example, an attack graph comprised of a single weakly connected component has the highest (most risky) Connectivity score. The Connectivity score decreases (is less risky) as the number of weakly connected components increases.

The *Cycles* metric is the number of strongly connected components in the domain-level attack graph, relative to the best (most secure) and worst (least secure) cases possible, on the scale of (0,10). The intuition is that for a (weakly) connected attack graph, it is better to avoid cycles within it.

To map the Cycles metric to the $(0,10)$ scale, we need the largest and smallest possible values for strong connectivity (at the protection domain level). The extremes for strong connectivity are the same as for weak connectivity in Figure 4. That is, the worst case is a single strongly connected component, and the best case is d strongly connected components for d domains.

These numbers of components need to be mapped to the (0,10) scale, consistent with the definition of zero as best case (most secure) and 10 as best case (least secure). Thus, for computing the Cycles metric, we apply the same formulas as for computing the Connectivity metric. The difference is that we count strongly connected components (attack sub-graphs that are all reachable from each other) versus weakly connected components.

Figure 5 shows an example computation of the Cycles metric. In this example, there are three attack graphs, shown at the protection domain level. Each attack graph has the same set of domains, but different sets of domain-to-domain edges, resulting in different numbers of strongly connected components. As shown in the example, an attack graph with fewer components (cyclic reachability within each component) has higher (more risky) Cycles score. The Cycles score decreases (is less risky) as the number of strongly connected components increases.

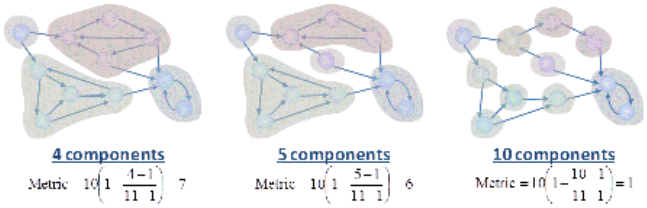

Figure 5. Example of Cycles metric.

The *Depth* metric is the length of the maximum shortest path in the domain-level attack graph, relative to the best (most secure) and worst (least secure) cases possible, on the scale of (0,10). In particular, this is the maximum shortest path over all possible attack graph vertex pairs, also known as the graph diameter. The intuition is that it is better to have attack graph that is deeper versus shallower, i.e., requiring more attack steps to penetrate the entire network.

To map the Depth metric to the $(0,10)$ scale, we need the largest and smallest possible values for the attack graph diameter (at the protection domain level). The worst case (least secure) is a diameter (maximum shortest path) of one. The best case (most secure) is a diameter that is one less than the number of domains d. These ranges of possible diameters need to be mapped to the $(0,10)$ scale, consistent with the definition of zero as best case (most secure) and 10 as best case (least secure). The Depth metric needs to consider the potential impact of connectivity on graph diameter. In particular, if a graph is not (weakly) connected, then the graph diameter applies to each (weakly) connected component separately.

We seek a function that maps the best case (diameter of one less than the full size c of the domain-level component) to the number zero (most secure), and the worst case (diameter of one) to the number 10 (least secure). This linear transformation shifts the diameter s to the left by one, divides by the range c, reverses the order by multiplying by negative one, and then shifts to the right by one. The resulting transformation maps the best case (diameter $c-1$ for component size c) to zero and the worst case (diameter one) to 10. This needs to be done for all n connected components of the domain-level attack graph, for d domains, with the diameter s_i for component i having size c_i. Thus, we have the Depth metric $\frac{10}{nd}\sum_i^n c_i\left(1-\frac{s_i-1}{c_i}\right)$.

Figure 6 shows an example computation of the Depth metric. In this example, there are three attack graphs, shown at the protection domain level. As shown in the example, an attack graph with larger diameter(s) relative to its connected component(s) has a lower (less risky) Depth score.

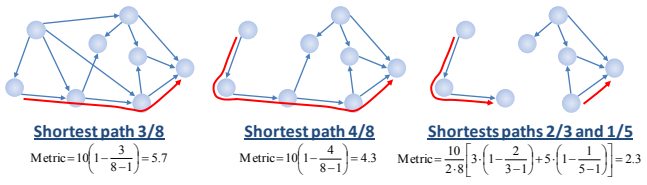

Figure 6. Example of Depth metric.

4. ILLUSTRATIVE EXAMPLE

As an illustrative example, consider the sequence of five attack graphs in Figure 7. This represents the exposed vulnerabilities for a network for a sequence of network hardening operations (software patches and firewall rule changes) over time.

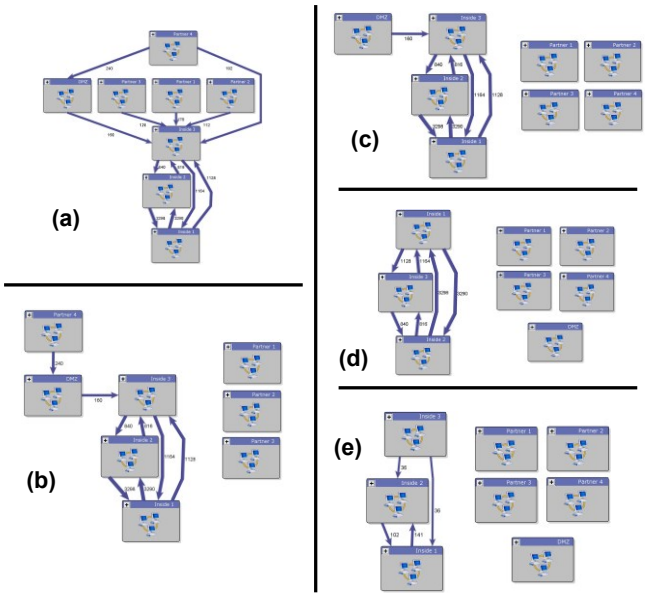

Figure 7. Attack graphs for metrics computation.

Figure 7(a) is the attack graph for the baseline network, before any software patches or firewall rule changes. Figure 7(b) is the result of blocking vulnerable ports from an outside partner network, although there is still vulnerable exposure from the outside into the DMZ. Figure 7(c) is the result of blocking the exposure into the DMZ, although there is still vulnerable exposure from the DMZ itself into the internal network. Figure 7(d) is the result of blocking the exposure from the DMZ to the inside. In Figure 7(e), patches for the two most frequently exposed vulnerabilities are applied to internal hosts, greatly reducing across-domain vulnerabilities in the internal network.

Figure 8 shows the overall and family metrics for the attack graphs in Figure 7. The metrics are shown in time order, from the baseline network through each successive risk reduction. In this case, despite the significant reduction in exposed vulnerabilities over time, the Victimization metric is relatively unchanged. This metric family captures charcteristics of the endpoint host vulnerabilities, independent of their exposure.

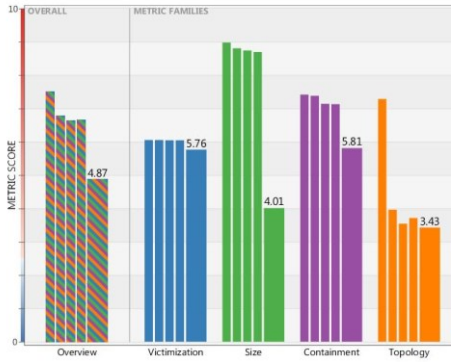

Figure 8. Metrics for attack graphs in Figure 7.

The Size family has modest reduction for attack graphs (a) through (d), but much stronger reduction for (e). This is because of the reduction of internal exposures when patches are finally applied. There is a similar pattern for the Containment family. For the Topology family, the largest reduction is the firewall rule change from (a) to (b), separating the graph into four components.

5. RELATED WORK

Security metrics have been proposed from a wide range of criteria, including intrusion detection, security policy, security incidents, game theory, dependability theory, and statistical methods. An in-depth survey of security metrics is given in [7].

A number of proposed security metrics employ attack graph models, including percentage of compromised hosts [2], the weakest adversary required to compromise a network [5], attack likelihood [4], and resilience to zero-day attacks [8].

Security metrics standardization efforts such as CVSS [3] and the NIST guidelines for security metrics [6] consider the relative severity of individual vulnerabilities in isolation, and do not consider the overall impact of combined vulnerablities.

6. SUMMARY

Our attack graph metrics suite has a number of distinct advantages. It incorporates a straightforward model with clear semantics, which helps lower barriers for acceptance. The grouping of metrics into families and an overall score helps reduce the cognitive burden of dealing with multiple scores. Also, our metrics fit within the larger framework of the established Cauldron tool for attack graph analysis. Preliminary experimental results suggest that our metrics are consistent with intuitive notions of attack risk across a network.

7. ACKNOWLEDGEMENTS

The work of Sushil Jajodia was supported in part by the Army Research Office under MURI awards number W911NF-13-1-0421 and W911NF-09-1-0525, and by the Office of Naval Research under award number N000141210461.

8. REFERENCES

[1] Jajodia, S., Noel, S., Kalapa, P., Albanese, M., Williams, J., "Cauldron: Mission-Centric Cyber Situational Awareness with Defense in Depth," 30th Military Communications Conference (MILCOM), November 2011.

[2] Lippmann, R., Ingols, K., Scott, C., Piwowarski, K., Kratkiewicz, K., Artz, M. , Cunningham, R., "Validating and Restoring Defense in Depth using Attack Graphs," IEEE Conference on Military Communications (MILCOM), 2006.

[3] NIST, NVD Common Vulnerability Scoring System (CVSS), http://nvd.nist.gov/cvss.cfm.

[4] Noel, S., Jajodia, S., Wang, L., Singhal, A., "Measuring Security Risk of Networks Using Attack Graphs," *International Journal of Next-Generation Computing*, 2010.

[5] Pamula, J., Jajodia, S., Ammann, P., Swarup, V. "A Weakest-Adversary Security Metric for Network Confguration Security Analysis," 2nd ACM Workshop on Quality of Protection, 2006.

[6] Swanson, M., Bartol, N., Sabato, J., Hash, J., Graffo, J., *Security Metrics Guide for Information Technology Systems*, NIST Technical Report 800-55, July 2003.

[7] Verendel, V., "Quantified Security is a Weak Hypothesis: A Critical Survey of Results and Assumptions," ACM New Security Paradigms Workshop, 2009.

[8] Wang, L., Jajodia, S., Singhal, A., Cheng, P., Noel. S., "k-Zero Day Safety: A Network Security Metric for Measuring the Risk of Unknown Vulnerabilities," *IEEE Transactions on Dependable and Secure Computing*, 2013.