A Suite of Metrics for Network Attack Graph Analytics

Steven Noel and Sushil Jajodia

Abstract This chapter describes a suite of metrics for measuring enterprise-wide cybersecurity risk based on a model of multi-step attack vulnerability (attack graphs). The attack graphs are computed through topological vulnerability analysis, which considers the interactions of network topology, firewall effects, and host vulnerabilities. Our metrics are normalized so that metric values can be compared meaningfully across enterprises. To support evaluations at higher levels of abstraction, we define family groups of related metrics, combining individual scores into family scores, and combining family scores into an overall enterprise network score. The *Victimization* metrics family measures key attributes of inherent risk (existence, exploitability, and impact) over all network vulnerabilities. The *Size* family is an indication of the relative size of the vulnerability attack graph. The *Containment* family measures risk in terms of minimizing vulnerability exposure across security 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.

1 Introduction

Modeling and analysis of network attack graphs has reached a fair level of maturity. A variety of tools are able to merge network data from various sources to build graphs of attack vulnerability through networks [1–7]. Such vulnerability-based 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.

S. Noel (✉)
The MITRE Corporation, McLean, VA, USA
e-mail: snoel@mitre.org

S. Jajodia
Center for Secure Information Systems, George Mason University, Fairfax, VA 22030-4444, USA

L. Wang et al., *Network Security Metrics*,
https://doi.org/10.1007/978-3-319-66505-4_7

This chapter describes 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.

Section 2 describes system architecture for computing our suite of attack graph metrics. In Sect. 3, we describe these metrics in detail. Section 4 shows how we portray our metrics as they evolve over time, through interactive visualizations. Section 5 examines our metrics suite through a case study that tracks the metrics over time as they are applied to a network that is progressively hardened. Section 6 describes related work in this area, and Sect. 7 summarizes our results and concludes this chapter.

2 System Architecture

Figure 1 depicts our system for computing security metrics from vulnerability-based network attack graphs. This system imports data from sources that are commonly deployed within enterprise networks, such as vulnerability scanners and firewall configuration files. The system then maps all the exposed vulnerabilities between pairs of hosts, which it organizes as an attack graph.

In this architecture, one option for computing such vulnerability-based attack graphs is the Cauldron tool [6]. Cauldron applies Topological Vulnerability Analysis (TVA) [8], analyzing network attack vulnerability from scan tools and other data sources. It correlates cyber vulnerabilities and environmental metadata, and applies network access policy (firewall) rules.

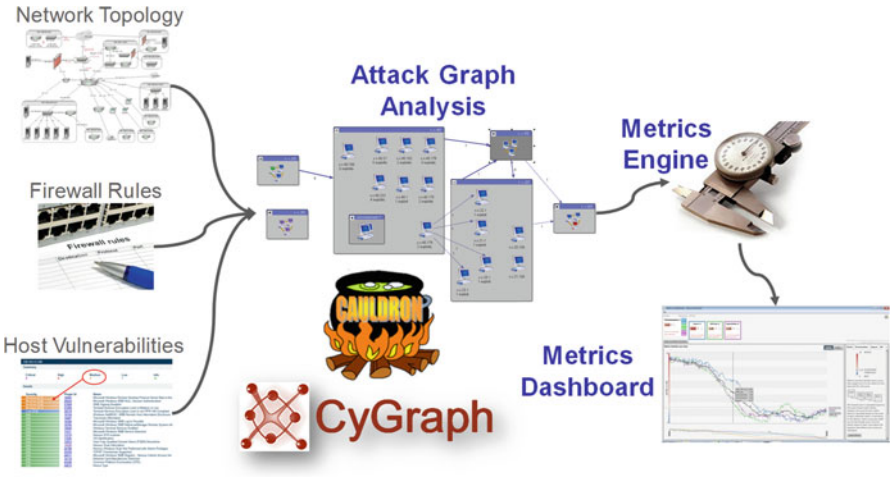

Fig. 1 Attack graph metrics suite

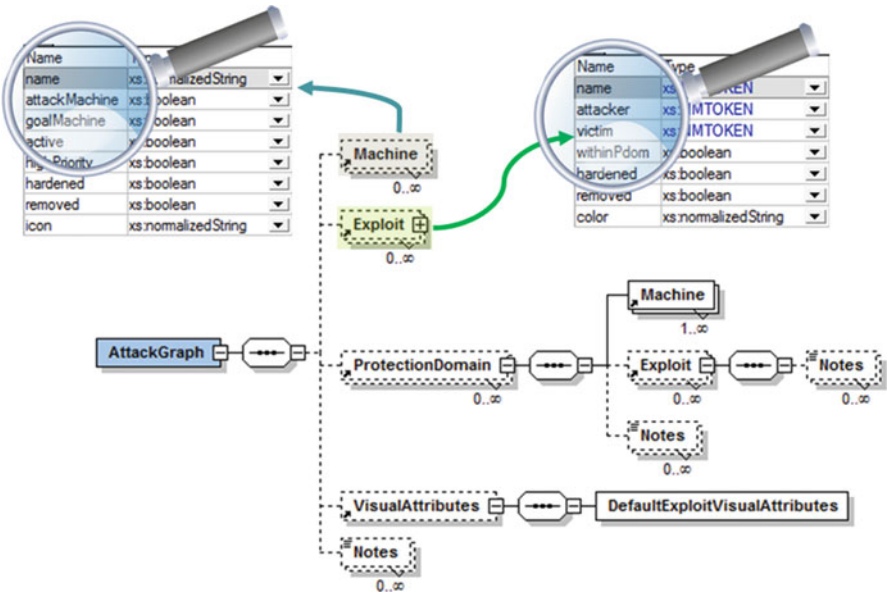

Fig. 2 Structure of attack graph model

By considering all source and destination host pairs, and testing reachability to other host vulnerabilities through the network topology and firewall rules, Cauldron finds each exposed host-to-host vulnerability vector, which it combines into an overall vulnerability attack graph. Vulnerability-based attack graphs computed in this way have quadratic complexity, i.e., $O(n^2)$ for n network hosts. For scaling to larger networks, we can apply the CyGraph tool [1, 2], which leverages big-data architecture.

In the architecture of Fig. 1, the attack graph engine (e.g., Cauldron or CyGraph) produces an attack graph representing attack reachability at a given time. Each attack graph instance is logged as input to the metrics computational engine, for the analysis of the metrics values over time.

Figure 2 shows the structure of an attack graph passed to the metrics engine. An attack graph is composed of a set of security protection domains, which contain host machines and exploits.

In the TVA attack graph model, protection domains are sets of machines that implicitly have reachability to each other's vulnerabilities. Optionally, machines (and exploits between them) can exist outside of protection domains. There other model elements for visual attributes (e.g., marking intrusion alerts). Figure 2 shows the model attributes for machines, including flags for attack start/goal, priority, and hardening state, as well as attributes for exploits, e.g., attacker and victim machines and indication of being within a protection domain (versus across domains).

Figure 3 shows the structure of the log produced by the metrics engine and consumed by the metrics dashboard. A metrics log begins with a definition of each

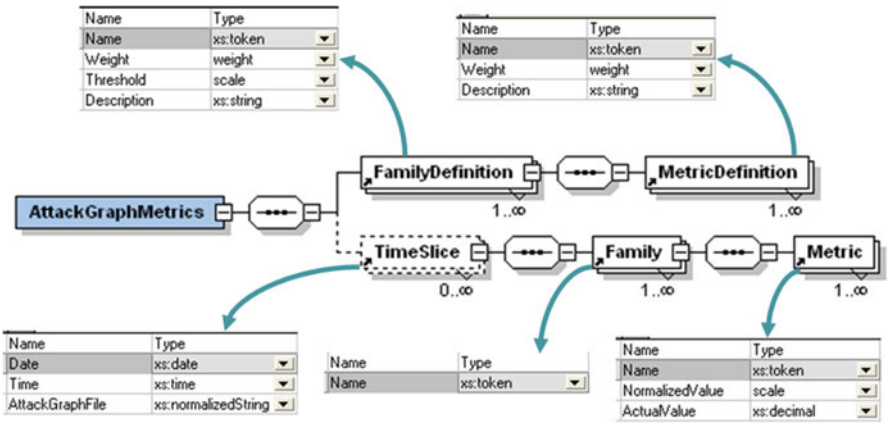

Fig. 3 Structure of attack graph metrics over time

metrics family, along with a definition of each metric within a family. The log then contains a sequence of time slices. Each time slice is the full set of metrics (family and individual) for a single point in time, derived from an attack graph of vulnerability paths through a network at that time.

Figure 3 includes the model attributes for a metrics family definition, including family name, relative weight (across families), threshold, and description. It also includes the attributes for an individual metric within a family, including metric name, relative weight (within the family), and description. The attributes for a metrics time slice include date/time and the corresponding attack graph file. For a metric family at a time slice, the attribute is the family name. The attributes for an individual metric are metric name and the normalized and actual metric values.

3 Attack Graph Metrics

The metrics engine computes individual metrics that each capture different aspects of overall security. We group related metrics into families, as shown in Fig. 4.

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 ten (least risk to most risk), as for the Common Vulnerability Scoring System (CVSS) [9].

We treat the individual metrics asindependent (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).

The following subsections describe each of our metrics families and the individual metrics within them, i.e., the Victimization family (Sect. 3.1), the Size family (Sect. 3.2), the Containment family (Sect. 3.3) and the Topology family (Sect. 3.4).

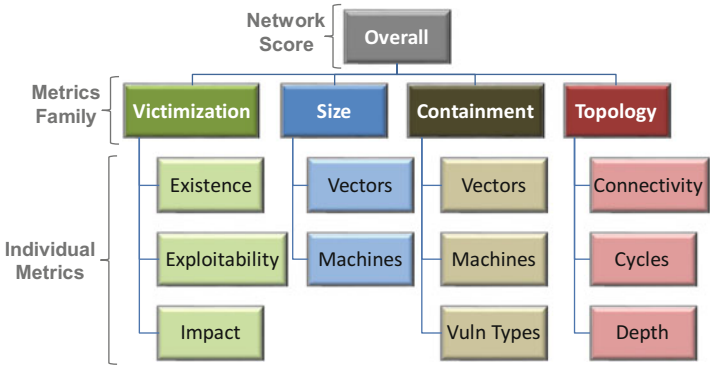

Fig. 4 Attack graph metrics families

3.1 Victimization Family

Individual vulnerabilities and exposed services each have elements of risk, independent of network topology and firewall access policy. These are risk dimensions inherent to the vulnerabilities themselves, in the sense of how they can be victimized by attackers. The *Victimization* metric family scores the entire enterprise network, as a summary of all vulnerabilities across these victimization dimensions.

The following subsections describe each of the individual metrics within the Victimization family, i.e., Existence (Sect. 3.1.1), and the two CVSS-based metrics (Exploitability and Impact) in Sect. 3.1.2. In Sect. 3.1.3, we describe how we combine these individual metrics into an overall metric score for the Victimization family.

3.1.1 Existence Metric

The Victimization family includes a direct measurement of the relative number of vulnerable network services. In particular, 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 $m_{\text{existence}}$ is simply the number of vulnerable network services (that have one or more vulnerabilities), relative to the total number of services:

$$m_{\text{existence}} = \frac{10 s_v}{s_v + s_n}.$$

3.1.2 CVSS-Based Metrics

The Victimization family also includes average scores (over all network vulnerabilities) of the two major components of CVSS Base Metrics—Exploitability and

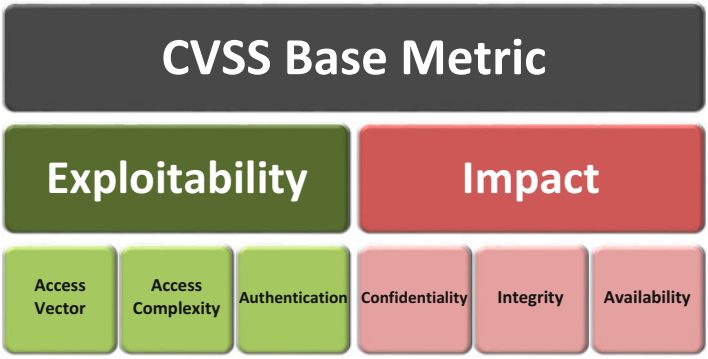

Fig. 5 Components of CVSS (Version 2) base metric

Impact. These components account for the two elements of risk, i.e., likelihood and impact. Figure 5 shows the components and sub-components of the CVSS Base Metric (for CVSS Version 2).

The following subsections these two CVSS-based metrics in more detail, i.e., Exploitability (Sect. 3.1.2.1) and Impact (Sect. 3.1.2.2).

Exploitability Metric

CVSS Exploitability measures the relative difficulty of exploiting a vulnerability. It includes the Access Vector (network, adjacent network, local, or physical) required for exploitation, Access Complexity (high or low) indicating level of attacker effort required, and Authentication (none, single, multiple) for the number of times the attacker must authenticate to a target. Our enterprise-wide *Exploitability* metric is the average value of the CVSS Exploitability score, averaged over all host vulnerabilities, on the scale of [0,10]. Given a vulnerability u_i, we denote its CVSS Exploitability as $\text{CVSS}_{\text{Exploitability}}(u_i)$. Then, for $|U|$ total vulnerabilities over all hosts in the network, the Exploitability metric $m_{\text{exploitability}}$ for the entire network is

$$m_{\text{exploitability}} = \frac{\sum_i^{|U|} \text{CVSS}_{\text{Exploitability}}(u_i)}{|U|}.$$

Impact Metric

The Impact component of CVSS measures the severity of impact upon exploitation of a vulnerability. It includes impact on data confidentiality, system integrity, and system availability, with each type denoted either complete, partial, or no impact. Our enterprise-wide *Impact* metric is the average value of the CVSS Impact score, taken over all vulnerabilities over all hosts, on the scale of [0,10]. Given a

vulnerability u_i, we denote its CVSS Impact as $\text{CVSS}_{\text{Impact}}(u_i)$. Then, for $|U|$ total vulnerabilities over all hosts in the network, the Impact metric m_{impact} for the entire network is

$$m_{\text{impact}} = \frac{\sum_i^{|U|} \text{CVSS}_{\text{Impact}}(u_i)}{|U|}.$$

3.1.3 Victimization Family Metric

Finally, we compute the metric $m_{\text{victimization}}$ for the entire Victimization family as the weighted Euclidean norm of the Victimization components:

$$m_{\text{victimization}} = \sqrt{\frac{(w_{\text{existence}}\, m_{\text{existence}})^2 + (w_{\text{exploitability}}\, m_{\text{exploitability}})^2 + (w_{\text{impact}}\, m_{\text{impact}})^2}{w_{\text{existence}}^2 + w_{\text{exploitability}}^2 + w_{\text{impact}}^2}}.$$

This treats the three individual Victimization metrics as components of a three-dimensional Euclidean space. The overall Victimization metric is then the norm (magnitude) of the vector with weighted Victimization components. Here, the weights $w_{[\text{existence,exploitability,impact}]}$ are (optional) user-defined weights for assigning relative strengths of the Victimization family components.

3.2 Size Family

The size of an attack graph is a prime indication of risk. Intuitively, the larger the graph, the more ways you can be compromised (in the sense of attack surface [10]). The *Size* metric family measures enterprise network risk in terms of the attack graph size.

The following subsections describe each of the individual metrics within the Size family, i.e., Attack Vectors (Sect. 3.2.1) and Reachable Machines (Sect. 3.2.2). In in Sect. 3.2.3, we describe how we combine these individual metrics into an overall metric score for the Size family.

3.2.1 Attack Vectors Metric

Within the Size family, we define the *Attack Vectors* metric as 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 Fig. 6, we must consider two kinds of attack vectors: implicit (within protection domains) and explicit (across domains). Here, as defined in TVA, a *protection domain* (shaded box in the figure) is a set of network machines that have unrestricted access to one another's vulnerabilities.The total number of attack vectors is the sum of the implicit and explicit attack vectors. That is, for

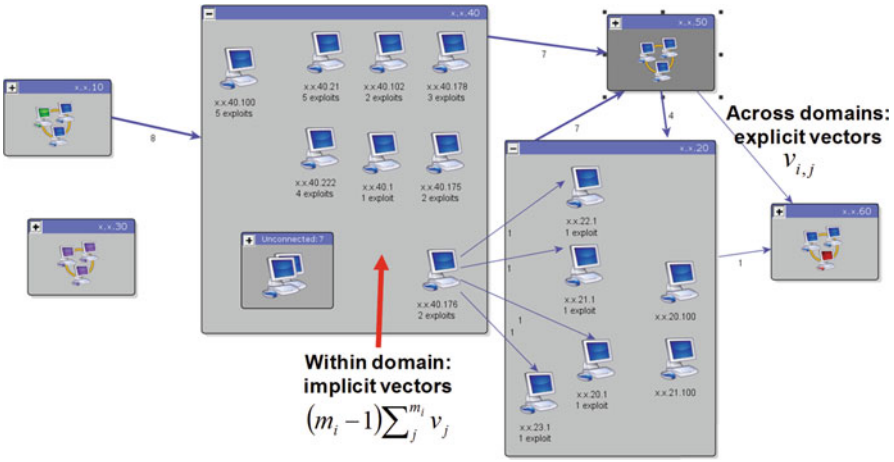

Fig. 6 Counting the single-step attack vectors

m_i vulnerable machines in protection domain i, v_j vulnerabilities on machine j, $v_{i,j}$ vulnerable (explicit) vectors domain i to domain j, and d domains, the total number of attack vectors v_a is

$$v_a = \sum_i^d (m_i - 1) \sum_j^{m_i} v_j + \sum_{i,j}^d v_{i,j}.$$

To map this raw number of attack vectors to the scale [0,10], we must normalize by the total possible number of attack vectors, i.e., in terms of all network services (both vulnerable and not vulnerable) across all machines. So, given m machines and s_i services on machine i, the total possible number of attack vectors v_p is then

$$v_p = (m - 1) \sum_i^m s_i.$$

The Attack Vectors metric, mapped to the scale [0,10] is then

$$v_{\text{attackVectors}} = 10\sqrt{\frac{v_a}{v_p}}.$$

Here, we apply the square root as a nonlinear compression that reduces dynamic range of the typically large difference between the number of possible and actual attack vectors.

3.2.2 Reachable Machines Metric

Also in the Size family is the *Reachable Machines* metric. This 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 Fig. 7, we must consider the machines that are in the attack graph (reachable by an attacker through some number of attack steps) as

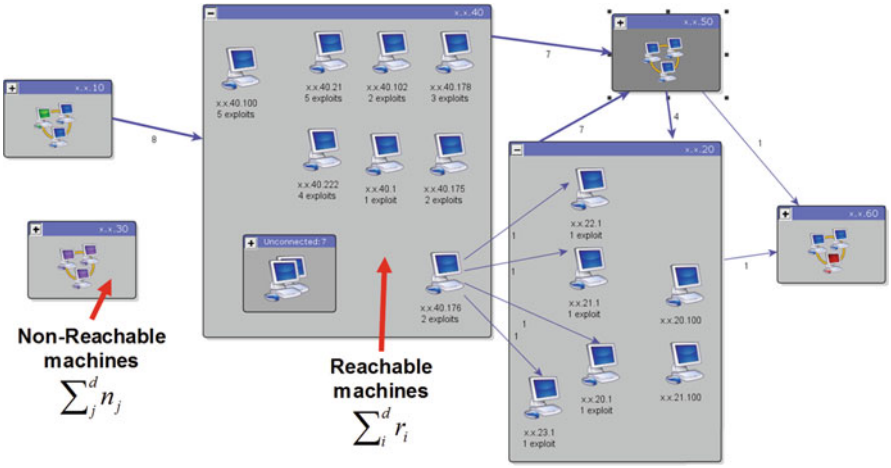

Fig. 7 Counting the number of attacker reachable machines

well as machines that are in the network but not in the attack graph. 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 n of non-reachable machines is

$$n = \sum_i^d n_i.$$

The Reachable Machines metric $m_{\text{reachableMachines}}$, mapped to the scale [0,10] is then

$$m_{\text{reachableMachines}} = 10\frac{r}{r+n}.$$

3.2.3 Metric for Size Family

The overall metric for the Size family m_{size} is then the weighted Euclidean norm of the Size metric components:

$$m_{\text{size}} = \sqrt{\frac{(w_{\text{attackVectors}}\ \ m_{\text{attackVectors}})^2 + (w_{\text{reachableMachines}}\ \ m_{\text{reachableMachines}})^2}{w^2_{\text{attackVectors}} + w^2_{\text{reachableMachines}}}}.$$

This treats the two individual Size metrics as components of a two-dimensional Euclidean space, with the overall Size metric as the norm (magnitude) of this vector. The weights $w_{[\text{attackVectors,reachableMachines}]}$ are (optional) user-defined weights for relative strengths of the Size family components.

3.3 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 which the attack graph contains attacks across security protection domains.

The following subsections describe each of the individual metrics within the Containment family, i.e., Vectors Containment (Sect. 3.3.1), Machines Containment (Sect. 3.3.2), and Vulnerability Types Containment (Sect. 3.3.3). In Sect. 3.3.4, we describe how we combine these individual metrics into an overall metric score for the Containment family.

3.3.1 Vectors Containment Metric

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 Fig. 6, the attack vectors across domains are explicit, and are simply counted across all domain pairs. The attack vectors within protection domains are implicit, i.e., all machine vulnerabilities are directly reachable within the domain. That is, the number of attack vectors across domains v_c is

$$v_C = \sum_{i,j}^{d} v_{i,j}.$$

The total number of attack vectors v_a, both across and (implicit) within domains is

$$v_a = \sum_{i}^{d} (m_i - 1) \sum_{i}^{d} v_i + \sum_{i,j}^{d} v_{i,j}.$$

The Vectors Containment metric m_{vecsC} is then

$$m_{\text{vecsC}} = 10 \cdot \frac{v_c}{v_a}.$$

3.3.2 Machines Containment Metric

Next, 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 Fig. 8, the victim

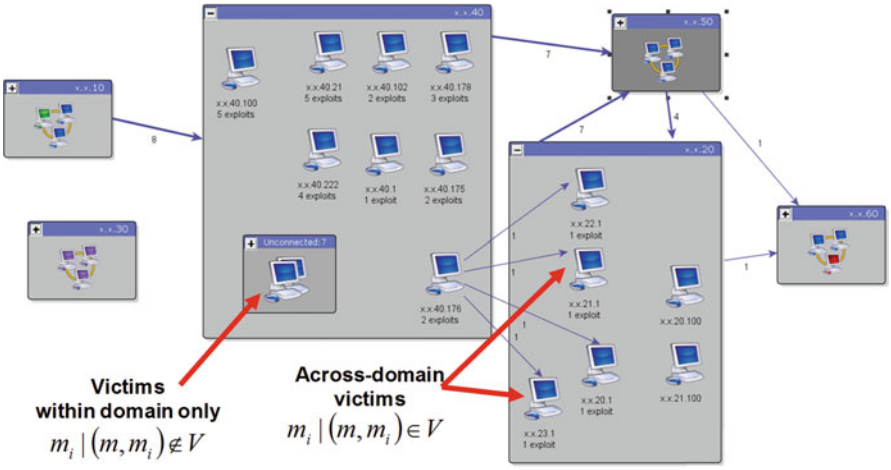

Fig. 8 Counting the number of attack graph victim machines

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. That is, the total number of across-domain victim machines m_a is

$$m_a = \sum_i^d \{m_i \mid (m, m_i) \in V\}.$$

The total number of within-domain victim machines m_w is

$$m_w = \sum_i^d \{m_i \mid (m, m_i) \notin V\}.$$

The Machines Containment metric m_{machsC} is then

$$m_{\text{machsC}} = 10 \cdot \frac{m_a}{m_a + m_w}.$$

3.3.3 Vulnerability Types Metric

The *Vulnerability Types* metric is the number of unique vulnerability types in the attack graph 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 Fig. 9, the across-domain vulnerability types are on hosts 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 compared to the same number of instances of differing vulnerability types.That is, the total number of across-domain victim machines t_a is

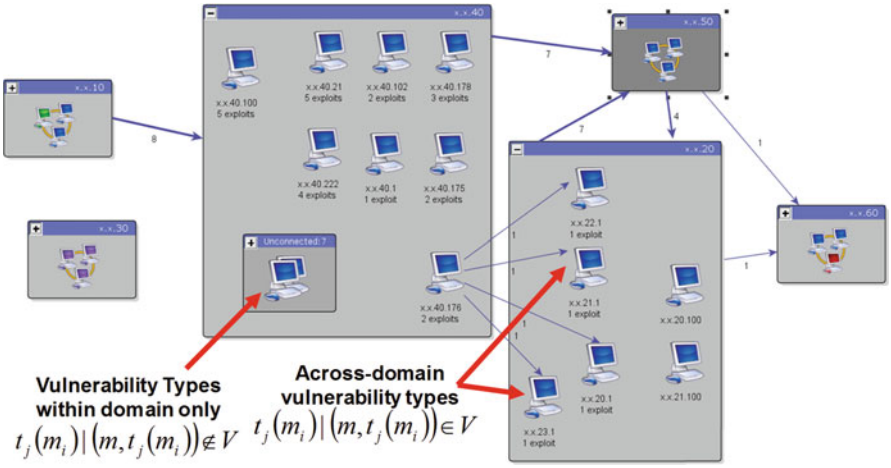

Fig. 9 Counting the number of attack graph vulnerability types

$$t_a = \sum_i^d \{t_i(m_i) | (m, t_i(m_i)) \in V\}.$$

The total number of within-domain victim machines t_w is

$$t_w = \sum_i^d \{t_i(m_i) | (m, t_i(m_i)) \notin V\}.$$

The Vulnerability Types metric m_{typesC} is then

$$m_{\text{typesC}} = 10 \cdot \frac{t_a}{t_a + t_w}.$$

3.3.4 Metric for Containment Family

The overall metric for the Containment family $m_{\text{containment}}$ is then the weighted Euclidean norm of the Containment metric components:

$$m_{\text{containment}} = \sqrt{\frac{(w_{\text{vecs C}} m_{\text{vecs C}})^2 + (w_{\text{machs C}} m_{\text{machs C}})^2 + (w_{\text{types C}} m_{\text{types C}})^2}{w_{\text{vecs C}}^2 + w_{\text{machs C}}^2 + w_{\text{types C}}^2}}.$$

This treats the three Containment metrics as components of a Euclidean space, with the overall Containment metric as the norm (magnitude) of this vector. The weights $w_{[\text{vecsC,machsC,typesC}]}$ are (optional) user-defined weights for relative strengths of the Containment family components.

3.4 Topology Family

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

The following subsections describe each of the individual metrics within the Topology family, i.e., Connectivity (Sect. 3.4.1), Cycles (Sect. 3.4.2), and Depth (Sect. 3.4.3). In Sect. 3.4.4, we describe how we combine these individual metrics into an overall metric score for the Topology family.

3.4.1 Connectivity Metric

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]. As shown in Fig. 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 standard [0,10] scale, we need the largest and smallest possible values for weak connectivity (at the protection domain level). This is shown in Fig. 11. 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 ten as best case (least secure).

As suggested by Fig. 12, to map to the [0,10] scale, we need to define a function that linearly maps the best case (d components) to the number zero (most secure), and the worst case (one component) to the number ten (least secure). This function is the following sequence of linear transformations:

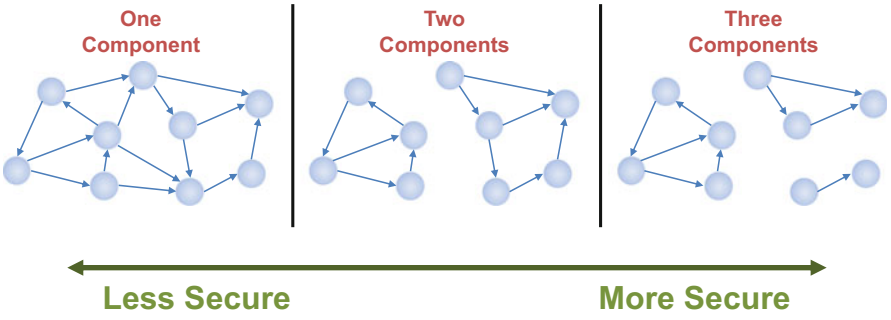

Fig. 10 Motivation for Connectivity metric

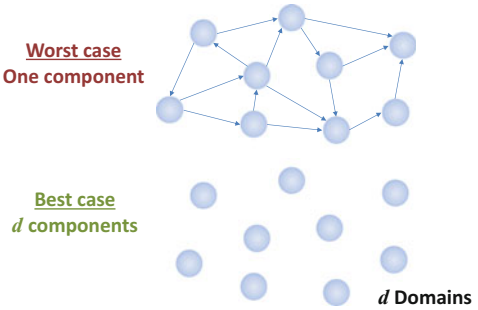

Fig. 11 Worst and best cases for weakly connected components

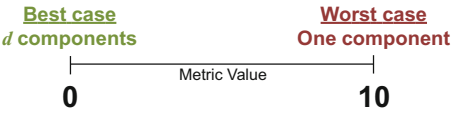

Fig. 12 Mapping extremes of weak connectivity to standard scale

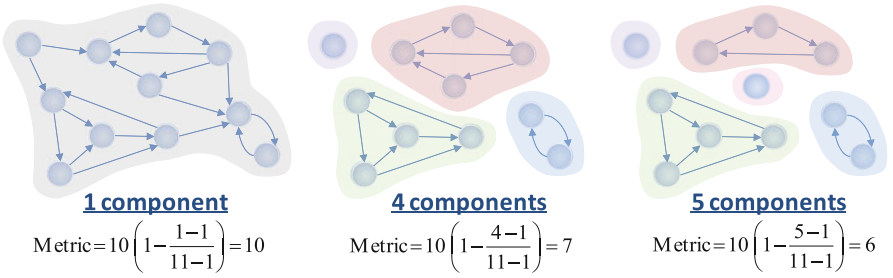

Fig. 13 Example Connectivity metric scores

1. Subtract the number of (weakly) connected components w_{weak} by unity, shifting them to the left by one
2. Divide by the range $d - 1$, normalizing the values to [0,1] (worst to best)
3. Multiply by negative unity, reversing the order to [-1,0] (best to worst)
4. Add unity, shifting to the right by one to [0,1] (best to worst)
5. Multiply by 10, yielding the scale [0,10] (best to worst)

The resulting transformation maps the best case (d components) to zero and the worst case (one component) to 10. This yields the Connectivity metric $m_{\text{connectivity}}$:

$$m_{\text{connectivity}} = 10\left(1 - \frac{w_{\text{weak}} - 1}{d - 1}\right).$$

Figure 13 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.

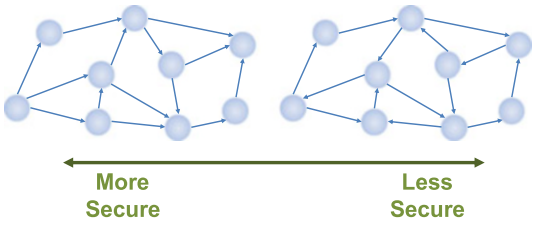

Fig. 14 Motivation for Cycles metric

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

3.4.2 Cycles Metric

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]. As shown in Fig. 14, the intuition is that for a (weakly) connected attack graph, it is better to avoid cycles within it (i.e., strongly connected components).

Comparing the two attack graphs in Fig. 14, they both have the same number of domains and domain-to-domain edges, and each graph has a single weakly connected component. However, the upper graph is more secure in the sense that all edges generally flow from left to right, so that attacker reachability is limited to that directional flow. On the other hand, the lower graph is less secure because the flow is cyclic. In fact, each domain is reachable from all other domains (i.e., cycle connecting all domains).

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 Fig. 11. That is, the worst case is a single strongly connected component, and the best case is d strongly connected components for d domains. As before, 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). Thus, for computing the Cycles metric, we apply the same formulas as for computing the Connectivity metric $m_{\text{connectivity}}$. The difference is that we count strongly connected components w_{strong} (attack sub-graphs that are all reachable from each other), versus weakly connected components w_{weak} as for $m_{\text{connectivity}}$. We thus have the Cycles metric m_{cycles}:

$$m_{\text{cycles}} = 10\left(1 - \frac{w_{\text{strong}} - 1}{d - 1}\right).$$

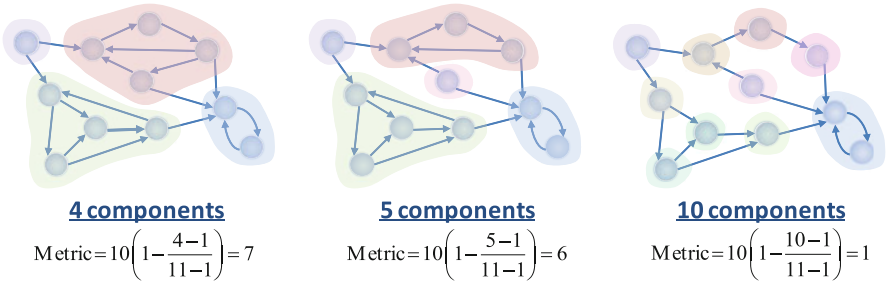

Fig. 15 Example Cycles metric scores

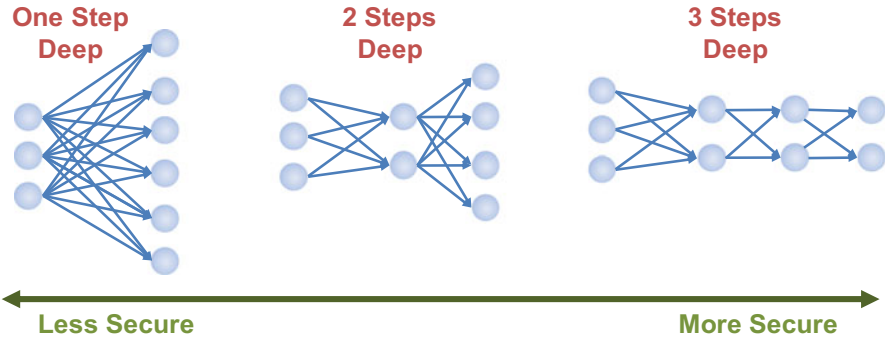

Fig. 16 Motivation for Depth metric

Figure 15 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 (riskier) Cycles score. The Cycles score decreases (is less risky) as the number of strongly connected components increases.

3.4.3 Depth 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. As shown in Fig. 16, 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.

Comparing the attack graphs in Fig. 16, they all have the same number of protection domains (graph nodes). In addition, each graph has a single weakly

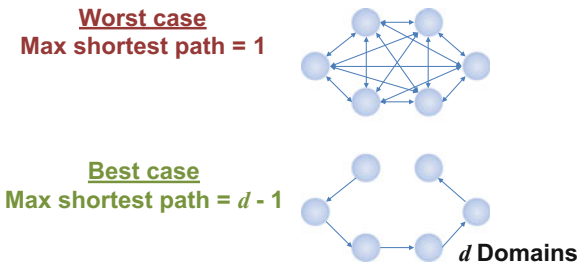

Fig. 17 Worst and best cases for graph diameter

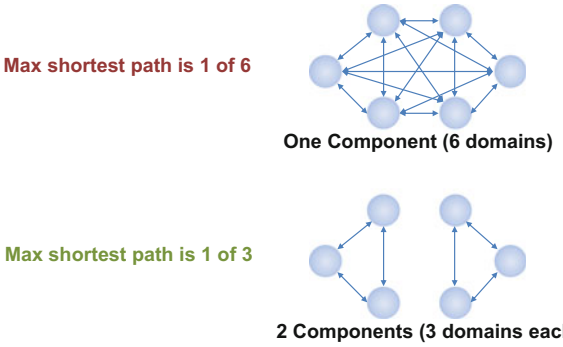

Fig. 18 Depth is relative to the size of connected components

connected component, and the maximum possible number of strongly connected components. However, the graph on the left side is less secure, in the sense that all domains are reachable in one attack step. On the other hand, the other graphs are more secure in the sense that more attack steps are needed before all domains are reached.

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). This is shown in Fig. 17. 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 ten as best case (least secure).

As shown in Fig. 18, 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.

For example, the upper attack graph in Fig. 18 has a single connected component, while the lower attack graph has two connected components. In each case, the graph diameter is one. However, a diameter of one is a different relative score compared to the maximum possible of five (upper graph) versus three (lower graph). We must compute diameter for each connected component, map to standard scale, then combine scores for each component according to relative component size.

As suggested by Fig. 19, we need a function that linearly maps the best case (diameter of one less than the full size c of the domain-level component) to the

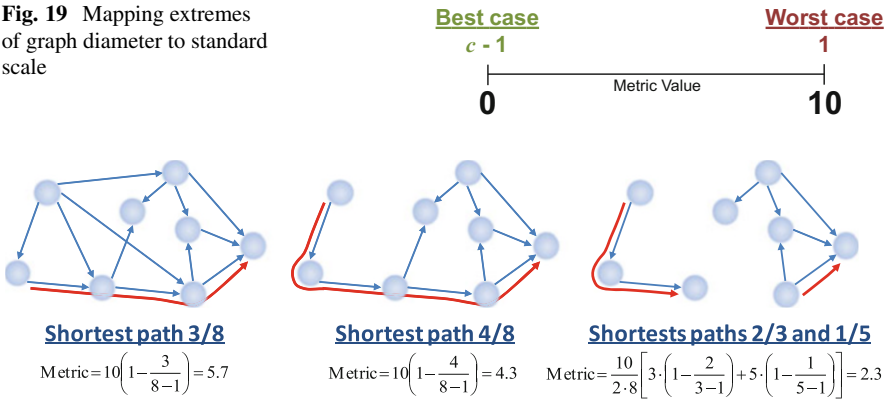

Fig. 19 Mapping extremes of graph diameter to standard scale

Fig. 20 Example Depth metric scores

number zero (most secure), and the worst case (diameter of one) to the number ten (least secure). This linear transformation does the following:

1. Shift the diameter s to the left by one (subtract unity)
2. Divide by the range c, normalizing the values to [0,1] (worst to best)
3. Multiply by negative unity, reversing the order to [-1,0] (best to worst)
4. Add unity, shifting to the right by one to [0,1] (best to worst)
5. Multiply by 10, yielding the scale [0,10] (best to worst)

The resulting transformation maps the best case (diameter $c - 1$ for component size c) to zero and the worst case (diameter one) to ten. 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. This yields the Depth metric m_{depth}:

$$m_{\text{depth}} = \frac{10}{nd} \sum_{i}^{n} c_i \left(1 - \frac{s_i - 1}{c_i}\right).$$

Figure 20 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.

3.4.4 Metric for Topology Family

The overall metric for the Topology family m_{topology} is then the weighted Euclidean norm of the Topology metric components:

$$m_{\text{topology}} = \sqrt{\frac{\left(w_{\text{connectivity}}\ m_{\text{connectivity}}\right)^2 + \left(w_{\text{cycles}}\ m_{\text{cycles}}\right)^2 + \left(w_{\text{depth}}\ m_{\text{depth}}\right)^2}{w_{\text{connectivity}}^2 + w_{\text{cycles}}^2 + w_{\text{depth}}^2}}.$$

This treats the three Topology metrics as components of a Euclidean space, with the overall Topology metric as the norm (magnitude) of this vector. The weights $w_{[\text{connectivity,cycles,depth}]}$ are (optional) user-defined weights for relative strengths of the Topology family components.

4 Metrics Visualization

We visualize our enterprise network security risk metrics in a dashboard for tracking and analyzing metrics values over time. The dashboard design presents the overall enterprise risk metric as the primary view, with drilldown into the details of the component metrics families.

Figure 21 shows the initial screen for the metrics dashboard visualization. In this view, the quick look for Overall is pressed; this causes the overall metric to be highlighted in the timeline (the individual families are diminished). The dashboard shows the initial and most recent date/time for the selected timeline. The overall and family quick looks show current (most recent) values, and changes with respect to the initial values. The quick looks also show how the current values compare to the threshold acceptable value. They also show the relative weights for each metrics family.

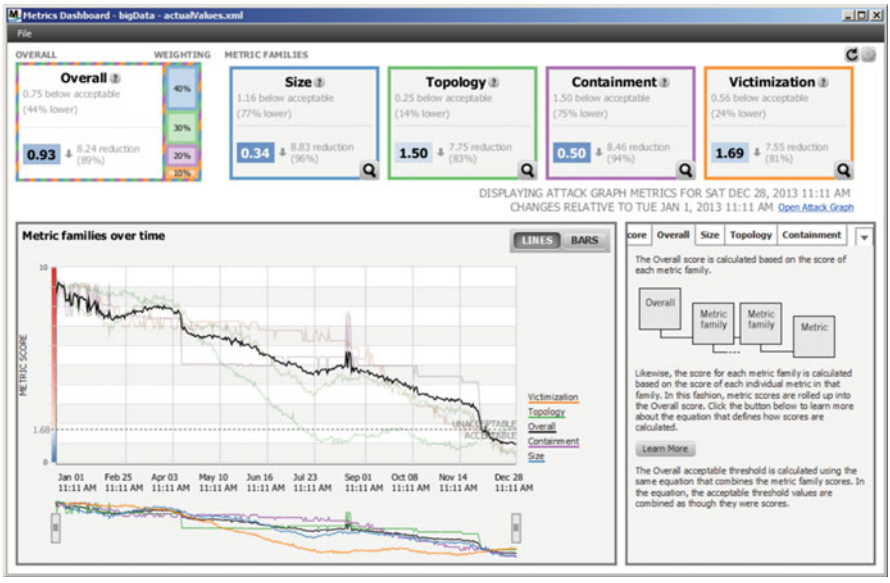

Fig. 21 Overall timeline for metrics dashboard

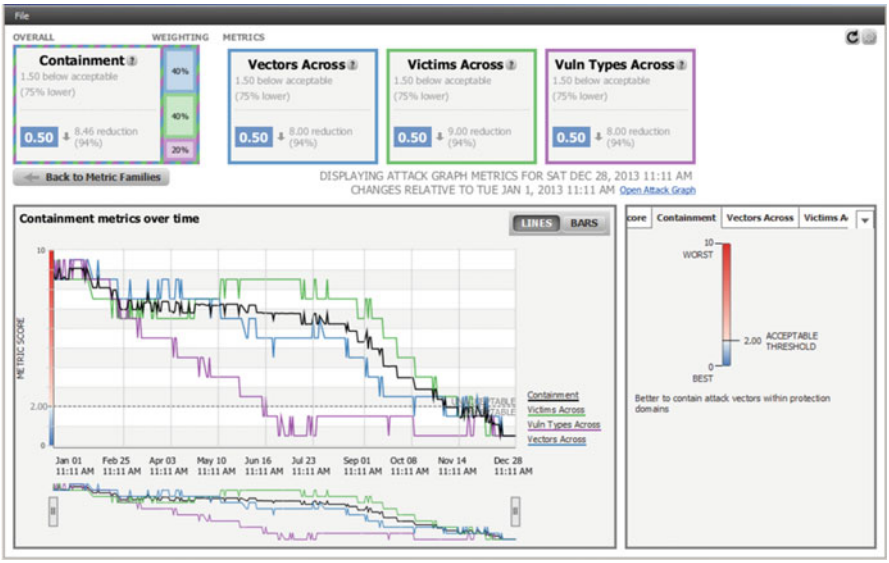

Fig. 22 Dashboard details for selected metrics family (Containment)

In Fig. 21, clicking on a magnifying glass on one of the family quick looks causes the display to show the details for the selected family. This is shown in Fig. 22, for the Containment family. The quick looks and timeline now show the overall family metric, as well as the individual metrics within the family. The selected time range persists in its current setting across overall to family view changes. The dashboard supports different thresholds for each family (and the overall), so that the threshold line changes accordingly.

The metrics dashboard also supports customization of certain settings, including metrics acceptability thresholds and relative weights. This is shown in Fig. 23. Thresholds and weights (for computing overall metric) can be selected for each family.

The dashboard allows the selection of time scale, as shown in Fig. 24. The slider along the bottom allows selection of starting and ending time to be displayed in the timeline. The time slider can also be panned backward and forward in time to display metrics values for a sliding time window. As shown in the figure, one can also hover over the timeline to display all the metrics values for a single point in time.

The dashboard also includes bar chart displays that summarize metrics trends. This is shown in Fig. 25. A bar chart is a binning of a corresponding sequence of metrics over time. In this way, an arbitrarily long history of metrics is displayed in a fixed number of bins (bars). The plus sign over each bar allows drilldown to the underlying metrics for that bar.

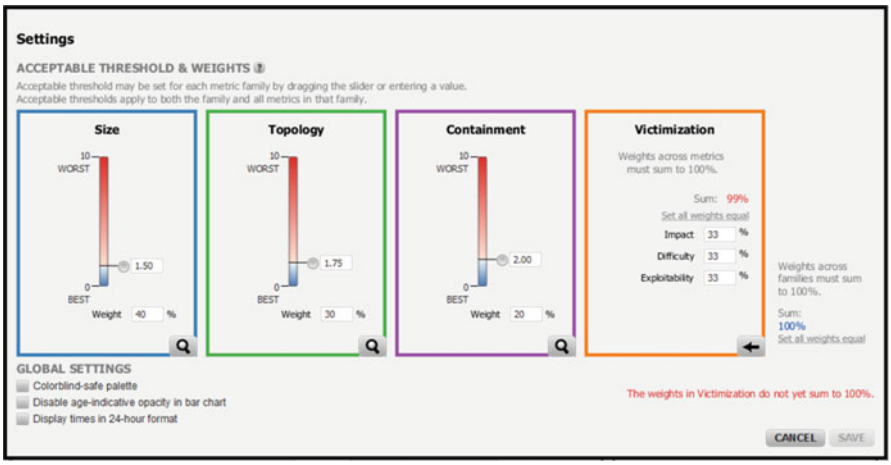

Fig. 23 Dashboard user-adjustable settings

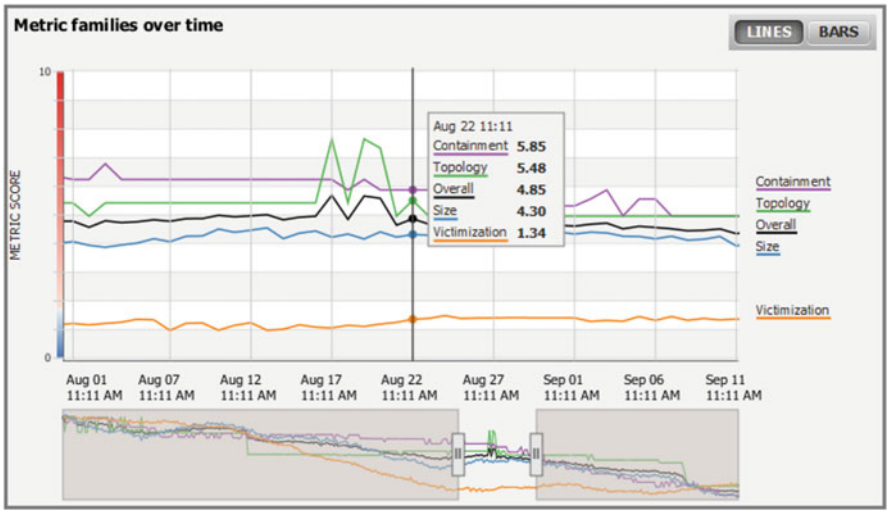

Fig. 24 Dashboard selection of time scale

5 Case Study

As a case study of our enterprise network security risk metrics, we consider a sequence of attack graphs representing the exposed vulnerabilities for a network, for a sequence of network hardening operations (software patches and firewall rule changes). We apply our metrics to track changes in enterprise risk over time.

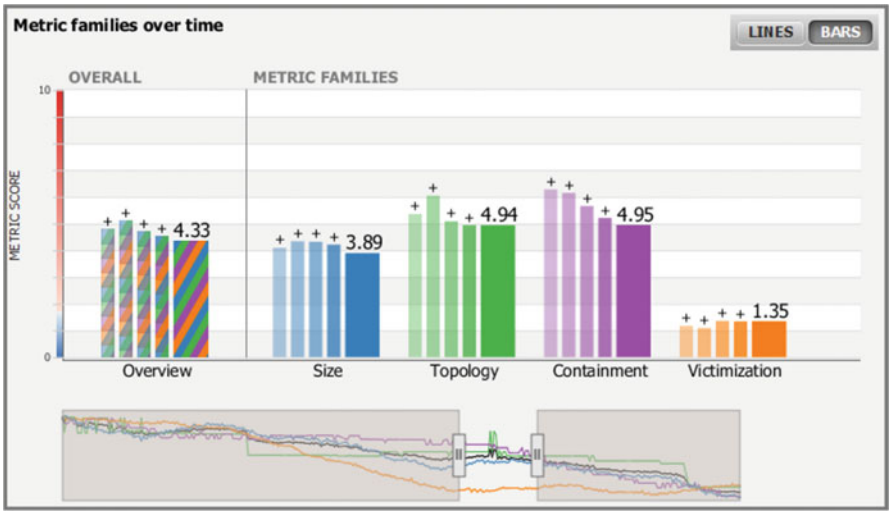

Fig. 25 Dashboard summary (binned) bar charts

In Sect. 5.1, we describe the network, hardening steps, and resulting attack graphs in our case study. Section 5.2 then computes our metrics for each of these attack graphs, and examines how these metrics quantify security risk.

5.1 Attack Graphs

Figure 26 shows a network topology for our case study, in which we generate attack graph metrics for different network configurations. This network contains eight security protection domains, with a multitude of machines within each domain. The enterprise to be protected has three internal domains and a DMZ domain. The enterprise DMZ is protected by a firewall. There is also an internal firewall, which protects the internal domains.

The enterprise allows some access from a partner organization, which has four domains. The primary defensive goal is to protect the internal domains against vulnerable exposures from either the partner domains or the DMZ.

In this case study, a Cauldron attack graph is generated for a baseline network configuration. The attack graph analysis identifies the critical exposures of vulnerable machines across domains. Mapping of vulnerable exposures to corresponding firewall rules leads to tightened policy based on mission requirements, which eliminate many of the vulnerable exposures.

Subsequent analysis indicates that the remaining exposed vulnerabilities are all among the internal enterprise domains. In that case, software patches are applied to

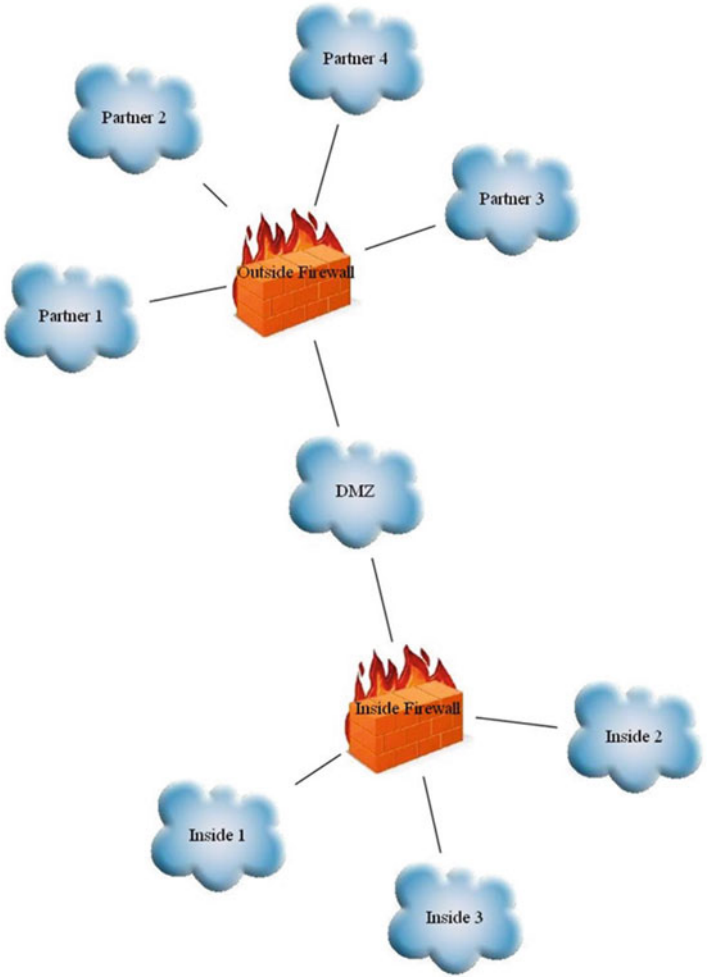

Fig. 26 Network topology for case study

remove the vulnerabilities, so that additional firewall blocking within the internal network is not needed to reduce risk.

Figure 27 shows the attack graph for the baseline network configuration, before any firewall rule changes or software patches have been applied. This attack graph shows that there are exposed vulnerabilities from the partner protection domains into the internal network, i.e., to the *Inside 3* domain. There are also exposed vulnerabilities from *Partner 4* to *DMZ*, and from *DMZ* to *Inside 3*.

An examination of the firewalls for rules permitting access into *Inside 3* reveals that there is a rule in both firewalls that allow access to all ports of certain machines. These machines are web servers that need to be accessed by the partners. However,

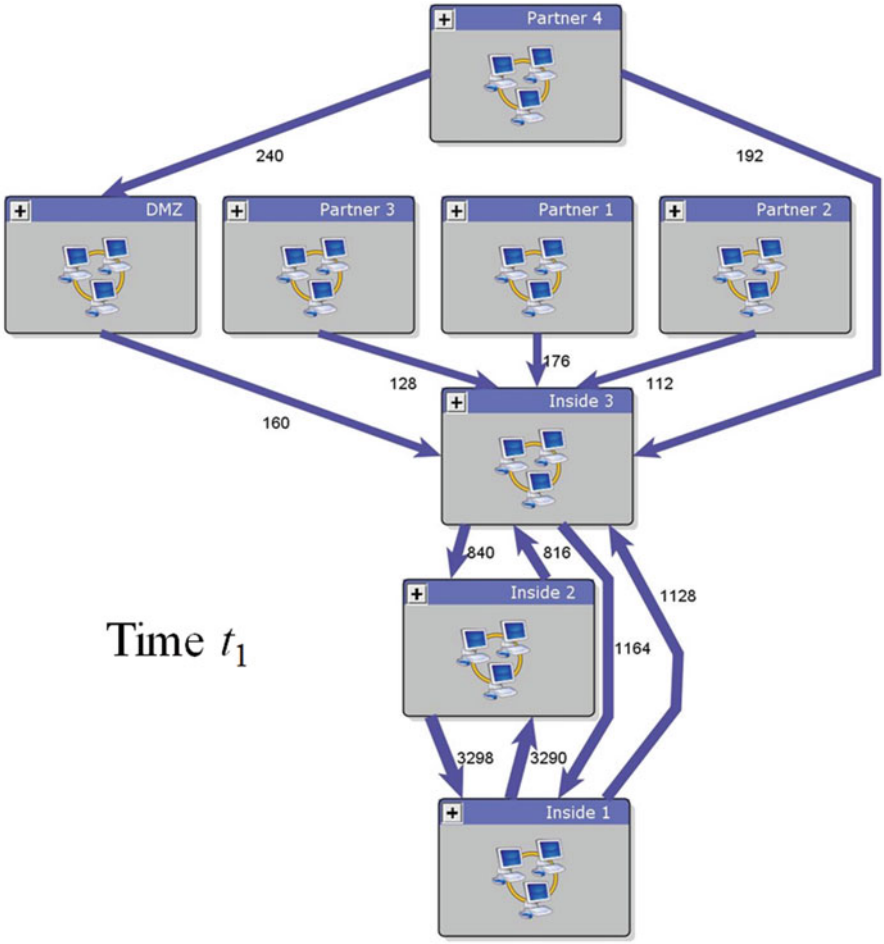

Fig. 27 Attack graph for baseline network

the vulnerabilities are actually on ports other than the HTTP port (80) needed by the mission. We therefore change the firewalls to allow access to port 80 only on these *Inside 3* machines. Figure 28 shows the resulting attack graph.

Figure 28 shows that there are still exposed vulnerabilities from *Partner 4* to *DMZ*. Examining the rules on the outside firewall for rules permitting access into *DMZ*, we see that there is a rule that allows access to all ports of the web servers in the *DMZ*. Again, the vulnerabilities are actually on ports other than the HTTP port (80) needed by the mission. We therefore change the outside firewall to allow access to port 80 only on these DMZ machines. No rule change is needed for the inside firewall, since it does not filter traffic from *Partner 4* to *DMZ*. Figure 29 shows the resulting attack graph.

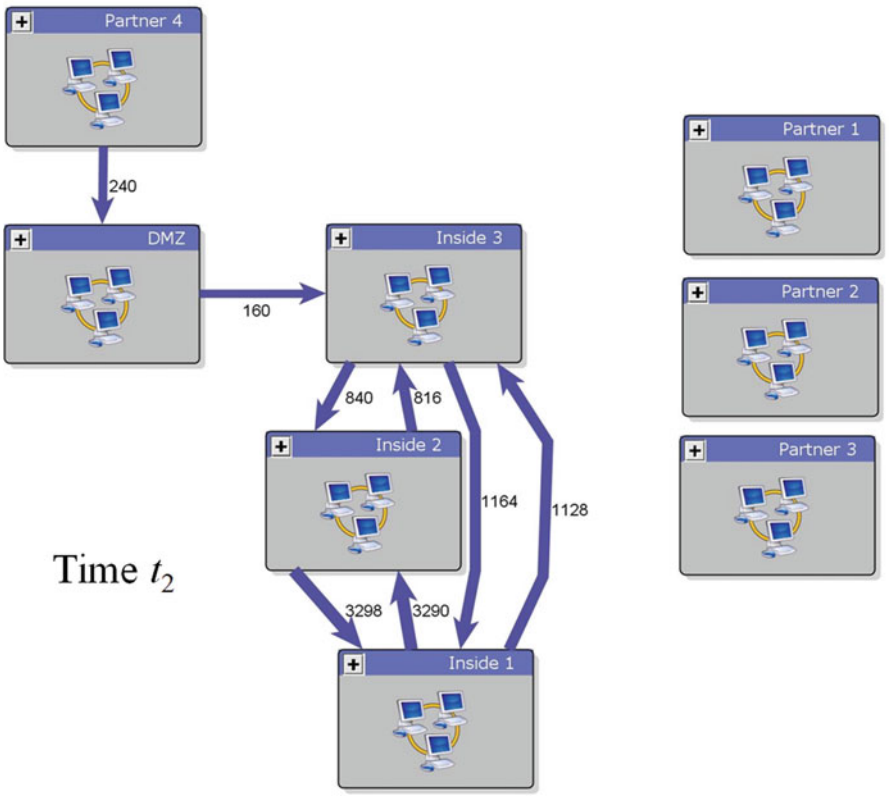

Fig. 28 Attack graph for restricting access to port 80 only from partners to *Inside 3*

Figure 29 shows there are still exposed vulnerabilities from *DMZ* to *Inside 3*. Examining the rules on the inside firewall, we see that there is a rule that allows access to all ports of web servers in *Inside 3*. Yet again, the vulnerabilities are on ports other than the HTTP port (80) needed by the mission. We therefore change the inside firewall to allow access to port 80 only on these *Inside 3* machines. No rule change is needed for the outside firewall, since it does not filter traffic from *DMZ* to *Inside 3*. Figure 30 shows the resulting attack graph.

Figure 30 shows that there are still exposed vulnerabilities among the inside protection domains. Under the assumption that further restriction of access within the inside domains will affect the mission, we consider the possibility of applying software patches.

Figure 31 shows the network vulnerabilities ranked (in Cauldron) by frequency of across-domain exposure for the attack graph in Fig. 30. This shows that two vulnerabilities are actually responsible for a large portion of the exposure instances. Figure 32 shows the resulting attack graph after all instances of those two vulnerabilities are patched.

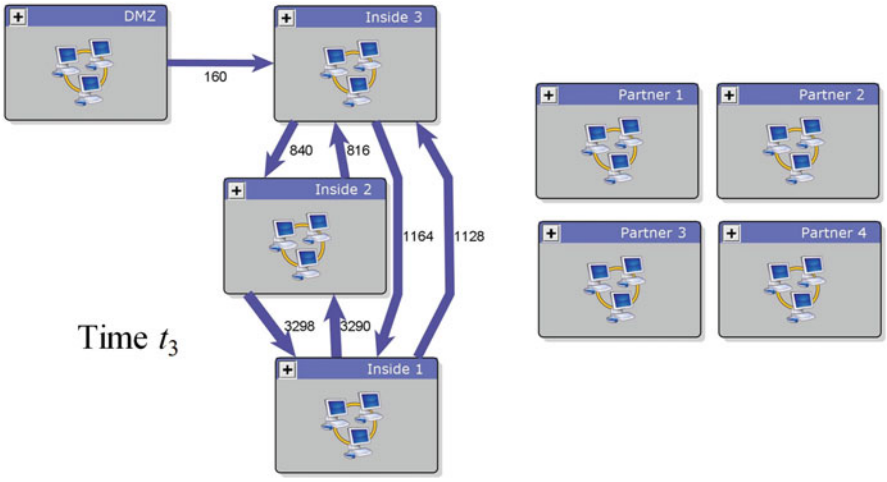

Fig. 29 Attack graph for restricting access to port 80 only from *Partner 4* to *DMZ*

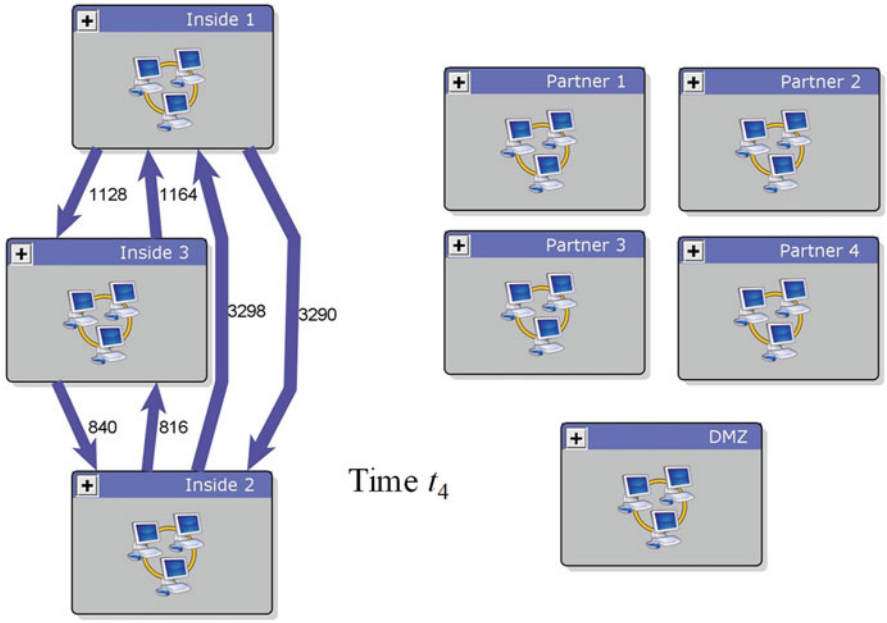

Fig. 30 Attack graph for restricting access to port 80 only from *DMZ* to *Inside 3*

In the next section, we compute metrics for each of these attack graph instances (Figs. 27, 28, 29, 30, and 31). Each attack graph represents the vulnerability exposures across the network at a given point in time, as steps were taken to incrementally reduce security risk (Fig. 33).

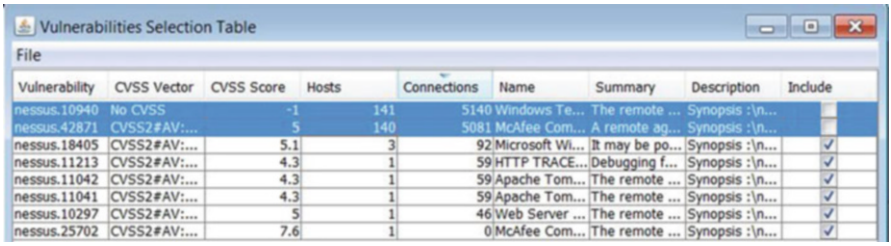

Fig. 31 Two vulnerabilities responsible for most risk exposures

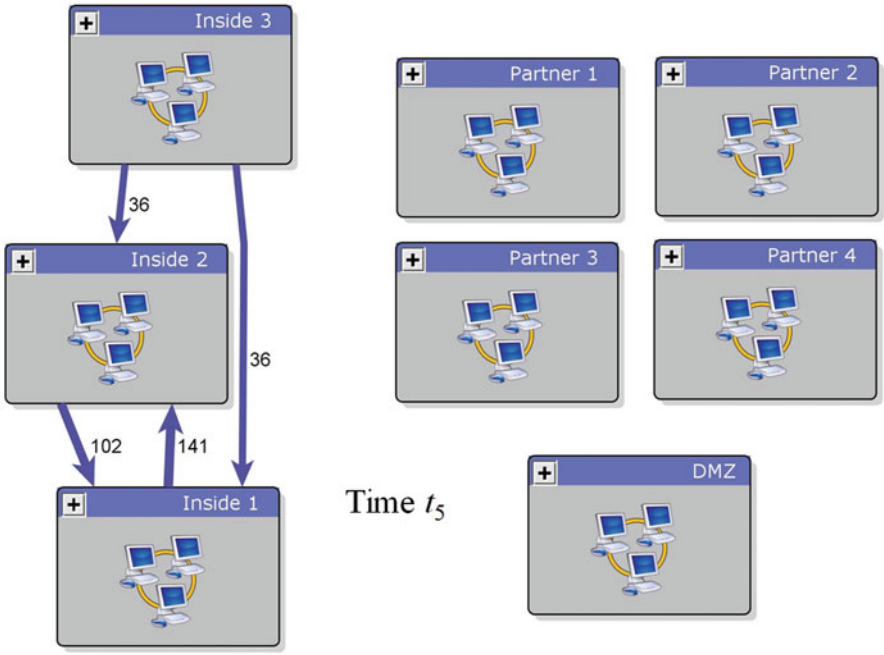

Fig. 32 Attack graph for the patching of two frequently exposed vulnerabilities

5.2 Security Risk Metrics

We now compute security risk metrics for the attack graphs in our case study (described in the previous section). By tracking these metrics over time, we assess the effectiveness of the network hardening measures taken at each step. For each attack graph representing the state of network security risk at time t_i, we compute the full suite of metrics (overall risk metric, four family-level metrics, and 11 individual metrics) for time t_i. We then plot these metrics values over time, at user-selected levels of detail. All metrics are calibrated from zero (least risk) to ten (most risk).

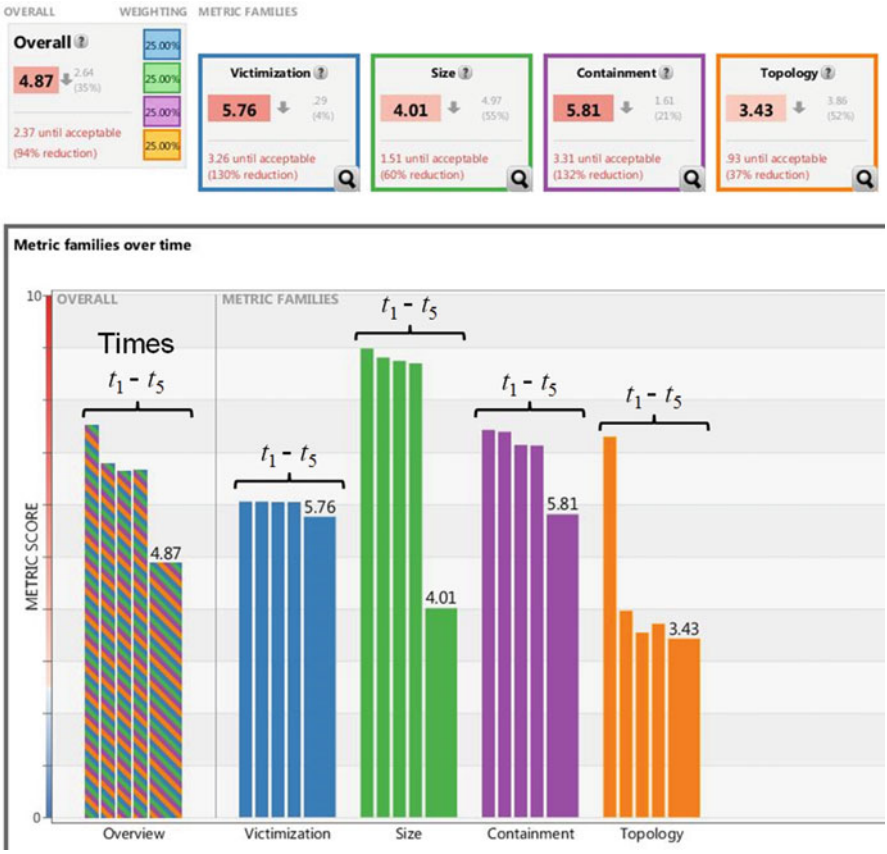

Fig. 33 Overall and family metrics for the case study (attack graphs at times t_1 through t_5)

In the next section (Sect. 5.2.1), we compute our metric for overall network risk for each instance in time (attack graph) in our case study. Section 5.2.2 then examines each of the family-level metrics in more detail.

5.2.1 Metric for Overall Network Risk

Figure 33 shows the initial metrics dashboard view for this case study. Because of the relatively small number of time values (five), the dashboard shows a bar chart rather than a line chart. The top row of the display is a quick view showing the current (most recent) values for the overall metric score and each of the four family-level scores. The left side of the main display shows scores for the overall network risk metric (for times t_1 through t_5). To the right are the scores (t_1 through t_5) for each of the metric families.

The overall risk score generally gets lower (at one point very slightly increasing), from an initial value of about 7.5 down to a final value of 4.87 (on the [0,10] scale). The family-level metrics follow this general trend, though with some differences.

The Victimization family metric is unchanged as a result of the firewall rule changes (times $t_1 - t_4$), decreasing only when vulnerability patches are applied at time t_5. This is consistent with the fact that the Victimization metrics depend on the state of endpoint hosts and their services/vulnerabilities, independent of attack reachability from other hosts. In that sense, the Victimization metrics are not actually based on attack graph analysis. Instead, they are summary statistics that one might find in more traditional vulnerability analysis (rather than TVA).

The metric for the Size family progressively decreases for each change in the network attack graph. The changes are relatively small for times $t_1 - t_4$, then there is a strong decrease for time t_5. The application of vulnerability patches at time t_5 causes the relatively large reduction in attack graph size, e.g., through the reduction of within-domain implicit attack vectors.

There is a similar pattern for the Containment metric, i.e., the number of vulnerabilities exposed across protection domains is significantly reduced (versus earlier network changes that do disconnect the attack graph, but only through relatively few across-domain vulnerabilities). On the other hand, the sharp drop in the Topology metric between t_1 and t_2 reflects the greater degree of topological changes (e.g., number of components increasing from one to four) between those times.

5.2.2 Family-Level Metrics

This section examines the metrics families for the five attack graphs in this case study. This includes the Victimization family (Sect. 5.2.2.1), the Size family (Sect. 5.2.2.2), the Containment family (Sect. 5.2.2.3), and the Topology family (Sect. 5.2.2.4). We show the individual metrics scores in each family, and how they combine into an overall metric for the family itself.

Victimization Family

Figure 34 shows the metrics within the Victimization family for the times $t_1 - t_5$. In this dashboard view, a user-defined threshold line of acceptable metric value (of 2.5 out of 10) is visible. The Existence metric is nearly constant just above the acceptable threshold for times $t_1 - t_4$, and then drops to nearly zero for time t_1. This reflects the number of vulnerable ports dropping to a relatively negligible number.

In Fig. 34, the Impact metric is low for times $t_1 - t_4$, then increases significantly for the last attack graph at t_5. This indicates that the patched vulnerabilities have relatively low impact (i.e., the remaining ones have higher impact). The Exploitability metric has the opposite trend; it is high for times $t_1 - t_4$, and then

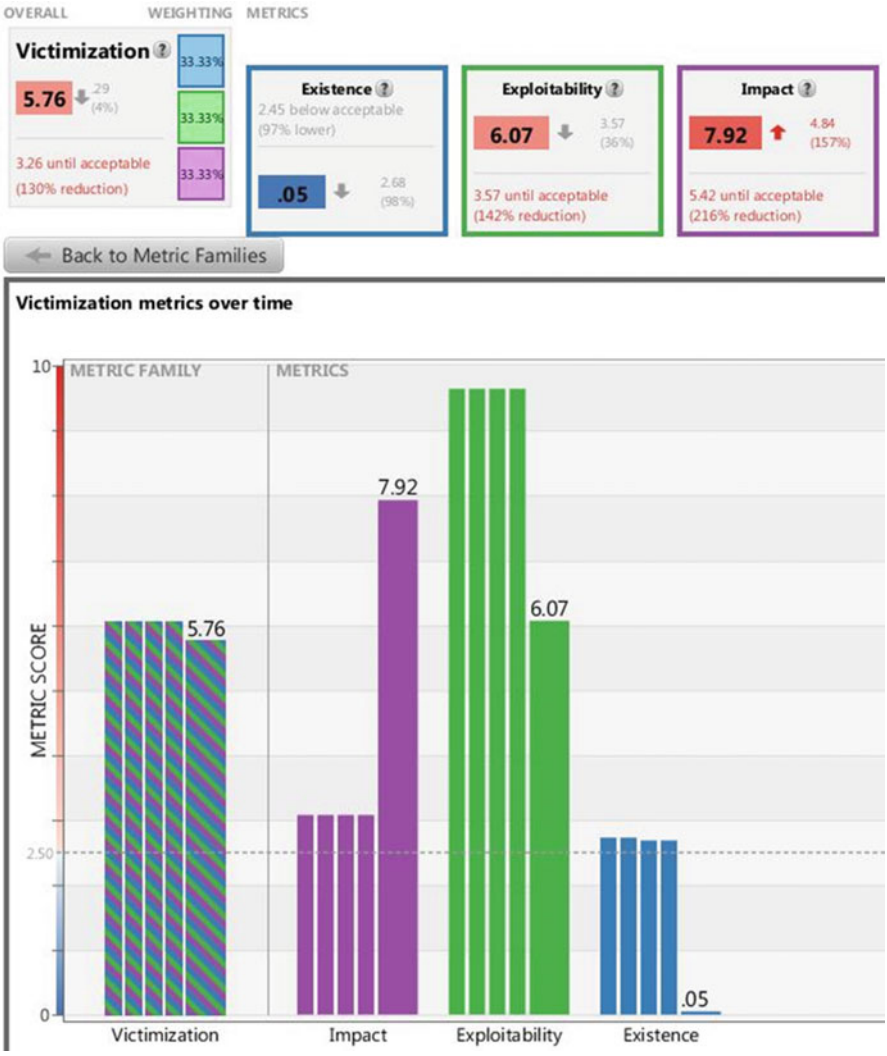

Fig. 34 Victimization metrics family

drops for t_5. This indicates that the patched vulnerabilities have relatively high exploitability (i.e., the remaining ones have lower exploitability).

The overall Victimization family metric changes little for these attack graphs. This is a result of the opposing trends for Impact versus Exploitability and Existence.

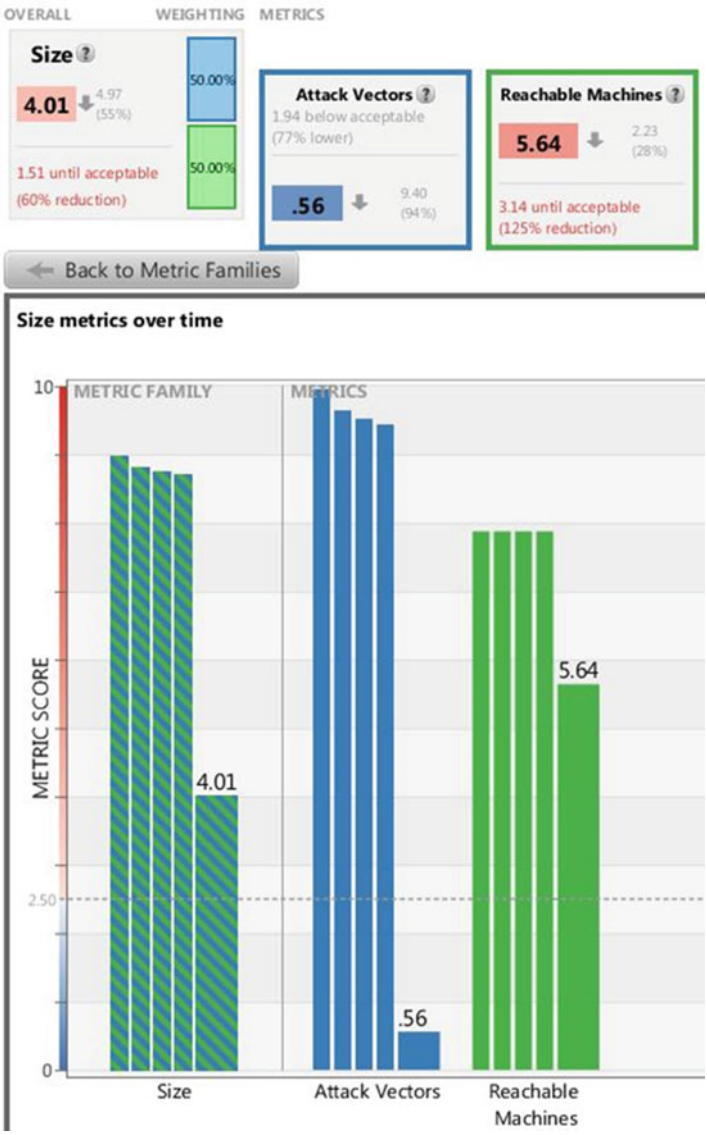

Fig. 35 Size metrics family

Size Family

Figure 35 shows the Size family for the times $t_1 - t_5$. The Attack Vectors metric decreases slightly for $t_1 - t_4$, and then sharply decreases to nearly zero for time t_5. This is consistent with the decrease from tens of thousands of across-domainattack

Fig. 36 Containment metrics family

vectors (Fig. 30) to a few hundred (Fig. 32) in the attack graphs. The number of attacker reachable machines is unchanged, until patches are applied that remove some of them.

Containment Family

Figure 36 shows the Containment family for the times $t_1 - t_5$. The Victims Across metric decreases slightly for the times $t_1 - t_4$, and then sharply decreases for time t_5.

This measures victim machines across domains, versus the Reachable Machines metric (Size family), which measures all reachable in the attack graph.

Similarly, the Vectors Across metric measures relative numbers of attack vectors across protection domains, versus the Attack Vectors (Size family), which measures all attack vectors (within and across domains). The Vulnerability Types Across measures distinct vulnerability types that are exposed across domains. In this case, it only changes for the last attack graph, since applied patches removed some vulnerability types.

Topology Family

Figure 37 shows the Topology family for the times $t_1 - t_5$. The Connectivity metric decreases from the highest possible risk (10) to nearly the threshold of acceptance (2.5). This reflects the subsequent decomposition of the attack graph into isolated components as hardening measures are applied.

The Cycles metric is relatively low, but remains unchanged except for the last attack graph at time t_5. This indicates that there are relatively few cycles in the attack graph, i.e., it has generally unidirectional flow.

The Depth metric is high for the baseline network at time t_1, then decreases to the threshold value for the first hardened attack graph at time t_2. This is because the direct access from *Partner 4* to *Inside 3* is removed, increasing the attack graph depth. In subsequent attack graphs (times), the Depth metric decreases as the attack graph has fewer steps.

6 Related Work

Cybersecurity metrics have been proposed based on a wide range of criteria, including intrusion detection, security policy, security incidents, game theory, dependability theory, and statistical methods [11–13]. There are many similarities between measuring cyber risk, cyber resilience [14–16], and cyber situational awareness [17]; particularly relevant current research at The MITRE Corporation seeks to measure the expected effectiveness of cyber resiliency.

Security metrics standardization efforts such as CVSS [9] and the NIST guidelines for security metrics [18] consider the relative severity of individual vulnerabilities in isolation, and do not consider the overall impact of combined vulnerabilities.

A number of proposed security metrics employ attack graph models, including those based on statistical properties of graph paths [19, 20], distances between attack graphs [21], percentage of compromised hosts [22], the weakest adversary required to compromise a network [23], attack success likelihood [24, 25], resilience to zero-day attacks [26], and scores along the dimensions of vulnerability, exploitability, and attackability [27]. Attack graph metrics have been applied for intrusion alert

Fig. 37 Topology metrics family

correlation [28] and prioritization [29]. Aspects of our attack graph metrics are previously described [30, 31].

7 Summary and Conclusions

This chapter describes a suite of metrics for measuring enterprise cybersecurity risk. These metrics measure risk based on a comprehensive network-wide model of multi-step attack vulnerability. Our metrics span different complementary dimensions of enterprise security, including elements of the CVSS standard. We provide rich interactive visualizations of multiple metrics, including timelines over multiple temporal scales, to understand how network security evolves over time.

We group our metrics into families, and combine individual scores into overall metric scores at the family level. We then combine family metric scores into an overall metric score for the network. We display these metrics (at the individual, family, and overall levels) in interactive visualizations, showing multiple metrics trends over time at user-selected temporal resolutions.

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. Experimental results suggest that our metrics are consistent with intuitive notions of attack risk across a network.

Acknowledgments The work of Steven Noel was funded in part by the MITRE Innovation Program (MIP) project *CyGraph: Graph-Based Analytics and Visualization for Cybersecurity* (project number EPF-14-00341), with George Roelke as MIP Cybersecurity Innovation Area Lead. The work of Sushil Jajodia was supported in part by the Army Research Office under grant numbers W911NF-13-1-0421 and W911NF-15-1-0576, by the Office of Naval Research under grant number N00014-15-1-2007, and by the National Science Foundation under grant number IIP-1266147.

References

1. S. Noel, E. Harley, K.H. Tam, M. Limiero, M. Share, CyGraph: graph-based analytics and visualization for cybersecurity, in *Cognitive Computing: Theory and Applications*, Handbook of Statistics, vol. 35, ed. by V. Raghavan, V. Gudivada, V. Govindaraju, C.R. Rao (Elsevier, New York, 2016)
2. S. Noel, E. Harley, K.H. Tam, G. Gyor, Big-data architecture for cyber attack graphs: representing security relationships in NoSQL Graph Databases, in *IEEE Symposium on Technologies for Homeland Security*, Boston, Massachusetts, April, 2015
3. Skybox Security, https://www.skyboxsecurity.com/
4. RedSeal Cybersecurity Analytics Platform, https://www.redseal.net/
5. M. Artz, *NetSPA: A Network Security Planning Architecture*, master's thesis, Massachusetts Institute of Technology (2002)
6. S. Jajodia, S. Noel, P. Kalapa, M. Albanese, J. Williams, Cauldron: mission-centric cyber situational awareness with defense in depth, in *30th Military Communications Conference (MILCOM)*, November 2011
7. X. Ou, W. Boyer, M. McQueen, A scalable approach to attack graph generation, in *13th ACM Conference on Computer and Communications Security*, New York, NY (2006)
8. S. Jajodia, S. Noel, Topological vulnerability analysis, in *Cyber Situational Awareness: Issues and Research, Advances in Information Security*, vol. 46, ed. by S. Jajodia, P. Liu, V. Swarup, C. Wang (Springer, Heidelberg, 2010)

9. NIST, NVD Common Vulnerability Scoring System (CVSS), http://nvd.nist.gov/cvss.cfm
10. P. Manadhata, *An Attack Surface Metric*, doctoral dissertation, Carnegie Mellon University, CMU-CS-08-152 (2008)
11. A. Jaquith, *Security Metrics: Replacing Fear, Uncertainty, and Doubt* (Addison-Wesley Professional, Reading, MA, 2007)
12. V. Verendel, Quantified security is a weak hypothesis: a critical survey of results and assumptions, in *ACM New Security Paradigms Workshop* (2009)
13. M. Pendleton, R. Garcia-Lebron, J.-H. Cho, S. Xu, A survey on systems security metrics. ACM Comput. Surv. **49**(4), 62 (2017)
14. D. Bodeau, R. Graubart, *Cyber Resilience Metrics: Key Observations*, The MITRE Corporation, https://www.mitre.org/sites/default/files/publications/pr-16-0779-cyber-resilience-metrics-key-observations.pdf (2016)
15. S. Musman, S. Agbolosu-Amison, *A Measurable Definition of Resiliency Using "Mission Risk" as a Metric*, The MITRE Corporation, https://www.mitre.org/sites/default/files/publications/resiliency-mission-risk-14-0500.pdf (2014)
16. D. Bodeau, R. Graubart, L. LaPadula, P. Kertzner, A. Rosenthal, J. Brennan, *Cyber Resiliency Metrics*, The MITRE Corporation, https://registerdev1.mitre.org/sr/12_2226.pdf (2012)
17. S. Noel, W. Heinbockel, An overview of MITRE cyber situational awareness solutions, in *NATO Cyber Defence Situational Awareness Solutions Conference*, Bucharest, Romania, August, 2015
18. M. Swanson, N. Bartol, J. Sabato, J. Hash, J. Graffo, *Security Metrics Guide for Information Technology Systems*, NIST Technical Report 800-55, July 2003
19. C. Phillips, L.P. Swiler, A graph-based system for network vulnerability analysis, in *ACM Workshop on New Security Paradigms*, New York, NY, USA, 1998
20. N. Idika, B. Bhargava, Extending attack graph-based security metrics and aggregating their application. IEEE Trans. Dependable Secure Comput. **9**(1), 75–85 (2012)
21. G. Bopche, B. Mehtre, Graph similarity metrics for assessing temporal changes in attack surface of dynamic networks. Comput. Secur. **64**, 16–43 (2017)
22. R. Lippmann, K. Ingols, C. Scott, K. Piwowarski, K. Kratkiewicz, M. Artz, R. Cunningham, Validating and restoring defense in depth using attack graphs, in *IEEE Conference on Military Communications (MILCOM)* (2006)
23. J. Pamula, S. Jajodia, P. Ammann, V. Swarup, A weakest-adversary security metric for network configuration security analysis, in *2nd ACM Workshop on Quality of Protection* (2006)
24. S. Noel, S. Jajodia, L. Wang, A. Singhal, Measuring security risk of networks using attack graphs. Int. J. Next-Gener. Comput. **1**, 135–147 (2010)
25. Z. Huang, *Human-Centric Training and Assessment for Cyber Situation Awareness*, doctoral dissertation, University of Delaware, ProQuest 10014764 (2015)
26. L. Wang, S. Jajodia, A. Singhal, P. Cheng, S. Noel, k-Zero day safety: a network security metric for measuring the risk of unknown vulnerabilities. IEEE Trans. Dependable Secure Comput. **11**, 30–44 (2013)
27. M. Tupper, A.N. Zincir-Heywood, VEA-bility security metric: a network security analysis tool, in *3rd International Conference on Availability, Reliability and Security* (2008)
28. S. Noel, E. Robertson, S. Jajodia, Correlating intrusion events and building attack scenarios through attack graph distances, in *20th Annual Computer Security Applications Conference (ACSAC)*, Tucson, Arizona, December 2004
29. S. Noel, S. Jajodia, Attack graphs for sensor placement, alert prioritization, and attack response, in *Cyberspace Research Workshop, Air Force Cyberspace Symposium*, Shreveport, Louisiana, November 2007
30. S. Noel, Metrics suite for network attack graphs, in *65th Meeting of IFIP Working Group 10.4 on Dependable Computing and Fault Tolerance*, Sorrento, Italy, January 2014
31. S. Noel, S. Jajodia, Metrics suite for network attack graph analytics, in *9th Annual Cyber and Information Security Research Conference*, Oak Ridge National Laboratory, Tennessee, April 2014