Expert Systems With Applications 193 (2022) 116467

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

# Prioritizing vulnerability patches in large networks ☆

Amir Olswang, Tom Gonda, Rami Puzis, Guy Shani *, Bracha Shapira, Noam Tractinsky

*Software and Information Systems Engineering, Ben Gurion University of the Negev, Beer Sheva, Israel*

ARTICLE INFO

*Keywords:*
Knowledge presentation
Decision support systems
Attack graph
Visualization
Security applications
Expert study

ABSTRACT

We consider here the question of prioritizing the patching of security vulnerabilities to prevent network attacks. Patching all vulnerable machines at once in large modern organizations is not feasible due to the large scale of their networks and the inability to halt operation during maintenance. This article explores two aspects of security maintenance: a method for prioritizing vulnerability patches, and visualization of the priorities to aid in decision making. State-of-the-art methods rank vulnerabilities by analyzing the connectivity graph or the logical attack graph and present the results in a table form, a view of the organizational network with highlighted failure points, or even the complete attack graph, in either case flooding the human operator with a lot of hardly comprehensible information. We suggest a Network Topology Vulnerability Score (NTVS) which shows preferable results by ranking vulnerabilities in a planning graph — an interim data structure used by planners when analyzing logical attack graphs. We also suggest a new abstracted presentation of the network in order to ease the comprehension of NTVS scores. The principal results obtained on two real networks show that patching vulnerabilities prioritized by NTVS leads to a faster decrease in the number of available attack paths toward the critical assets. A user study with a panel of security experts shows that the proposed visualization is considerably better than current commercial tools, helping experts to both prioritize vulnerability patches, and explain their decisions to higher management and to operation teams.



## 1. Introduction

Network security is more important than ever for organizations due to the increasing number of attacks that target enterprises, and the increasing regulations that require keeping customer data safe and private. The World Economic Forum in its 2018 report stated that cyber breaches recorded by businesses have almost doubled in five years (Forum, 2018). The National Institute of Standards and Technology (NIST) reports that the number of new vulnerabilities has increased from about 5632 in 2008 to about 16,514 in 2018 (NIST, 2019). Some attacks target critical systems or important data that is exposed directly to a public network, mostly the internet. There is substantial evidence (Shu et al., 2017) that advanced attacks penetrate the network through its weakest link and then utilize a chain of vulnerabilities to find an attack path inside the network, that will allow the attacker to compromise the desired critical assets or data.

Information Technology (IT) and security administrators work hard to maintain their organizational network and increase resilience to adversarial attacks. Vulnerability patching is one of the most important (Dempsey et al., 2012) yet most expensive security operations in terms of budget, man power, and service continuity. Indeed, all experts that we interviewed in this study indicated that the cost of remediating all vulnerabilities detected on the network by their vulnerability assessment tools is too high for their organizations. Given the large number of vulnerabilities and machines in enterprise networks administrators must carefully prioritize the vulnerability maintenance activities. They must consider operational and security consequences including downtime due to patching, the importance of the vulnerable asset to the enterprise, and the possible utility of the asset to the attacker as a stepping stone in a multi-stage attack.

Most security teams prioritize vulnerabilities based on their perceived risk and impact. The Common Vulnerability Scoring System (CVSS) (Mell et al., 2006) is formal framework for defining the severity of software vulnerabilities. CVSS is defined for a vulnerability regardless its circumstances within an organization. It does not capture the severity of vulnerabilities in context of sophisticated attacks, a vulnerability in a non critical system may be stepping stone on a path to a critical asset. An interesting example is the known target breach in which over 40 million credit card identities where stolen in 2013 (Shu et al., 2017). Attackers were able to gain access to a 3rd party HVAC system company. Through HVAC they allegedly penetrated the Target's

☆ Parts of this paper appeared in Gonda et al. (2017), Gonda et al. (2018).

* Corresponding author.
*E-mail addresses:* amir.olswang@forescout.com (A. Olswang), tomgond@post.bgu.ac.il (T. Gonda), puzis@bgu.ac.il (R. Puzis), shanigu@bgu.ac.il (G. Shani), bshapira@bgu.ac.il (B. Shapira), noamt@bgu.ac.il (N. Tractinsky).

https://doi.org/10.1016/j.eswa.2021.116467
Received 10 January 2021; Received in revised form 17 December 2021; Accepted 24 December 2021
Available online 12 January 2022

network. Within the Target's network the attackers were able to exploit a vulnerability in the point-of-sale system, which allowed them to steal credit card data.

Logical attack graphs (LAG) (Ou et al., 2005) is a modeling technique for analysis of sophisticated multi-stage attacks (Albanese et al., 2012). We elaborate on LAG structure in Section 3.1. LAG allows the inference of attack plans, sequences of actions (e.g. exploits) that allow attackers to achieve their goals, such as access to specific sensitive information (Hoffmann, 2015). Previous research suggested patching the vulnerabilities exploited in the shortest attack plans (Bhattacharya & Ghosh, 2008) or used various centrality measures, computed on the LAG or on the network connectivity graph, for prioritizing the vulnerabilities to be patched first (Hong & Kim, 2013; Sawilla & Ou, 2008). Others propose combinatorial optimization for eliminating all possible attack plans with minimal number of vulnerability patches (Albanese et al., 2012).

However, scoring the vulnerabilities is not sufficient to build a well motivated security maintenance plan. Security administrators must understand the severity of the top ranked vulnerabilities in the most intuitive way. Previous work suggested visualizations of the entire attack graphs at various levels of granularity (Barrère & Lupu, 2017; Homer et al., 2008; Jajodia & Noel, 2010; Williams et al., 2008). Manual analysis of attack paths in large networks requires substantial effort. In addition, visualizations of entire attack graphs are hard to comprehend and communicate between the security teams, operational teams that actually execute the security maintenance plans (Shostack, 2003), and the higher management.

In this paper we propose a decision support framework consisting of main contributions: the **Network Topology Vulnerability Score (NTVS)** and the **Vulnerability Focused Attack Path Visualization**.

The first contribution, NTVS, ranks vulnerabilities by computing their Page Rank within the (relaxed) planning graph — a data structure often used in the classical planning community, to compute forward search heuristics (Hoffmann, 2003). NTVS represents the advantage an attacker gains by exploiting specific vulnerability while targeting a critical asset. We demonstrate over a range of benchmarks including scans of a real organization network, that centrality measures computed over the planning graph are superior to centrality measures computed over the LAG and the connectivity graph for prioritizing vulnerabilities.

The second contribution, visualization, simplifies the presentation of the computed ranks. Human experts must understand why a certain vulnerability is more important than the other when they devise the final maintenance plan. Our presentation scheme (see Fig. 1) allows the expert to rapidly understand the position and role of a vulnerable machine in the organizational network.

We evaluate the proposed solution using: (1) simulations performed with attack graphs of real organizations, highlighting the superiority of the planning graph analysis when prioritizing vulnerabilities; and (2) a panel of ten hands-on experienced security experts that face the necessity of prioritizing maintenance tasks on a daily basis. Our results clearly indicate that the experts recognized the value of the NTVS score and its visualizations, and preferred the proposed user interfaces over the current commercial solutions.

We conduct offline experiments over real world networks. We experiment with both real and artificial networks, demonstrating that results over the artificial networks do not apply to the real world networks. The proposed visualization and its applicability to real-world decision making received strong endorsement from security experts who evaluated it.

Our goal here is to develop methods that rank vulnerabilities such that an administrator would need to patch less vulnerabilities before making the important assets of the organization safe.

As such, the NTVS score allows IT to rapidly evaluate the importance of patching a vulnerability in the context of an attack on a network.

## 2. Related work

In this section we discuss related work analysis and visualization of attack graphs. We also discuss the commercial baseline to which our improved presentation is compared.

### 2.1. Scoring vulnerability patches

#### 2.1.1. Device/vulnerability level scoring

The Common Vulnerability Scoring System (CVSS) score (Mell et al., 2006) is a popular tool for capturing the severity of vulnerabilities. CVSS is an open standard for assessing the risk of a vulnerability. It is comprised of three sub scores — Base, Temporal and Environmental. Base represents the ease of exploiting a vulnerability, its scope, and its impact on an asset. Temporal is a score for metrics that evolve over time, such as the state of exploit techniques, the existence of patches, and more. The environmental component allows the analyst to customize the CVSS score depending on the importance of the affected IT asset to the organization. This component captures only the importance of the vulnerable asset, ignoring other assets that are in an attack path going through this vulnerability.

Most commercial applications use the base score, while some update it with temporal intelligence. Commercial applications typically do not implement the full environmental component, as it requires detailed customization in each organization.

Mell et al. (2006) investigated the impact of using a more complete CVSS score as opposed to using only the base score, as many commercial solutions do today. They suggested the use of publicly available information that can contribute to the temporal and environmental CVSS score. They showed that by using this available information they can reduce vulnerability management cost by 19% and allow teams to better focus on the riskier issues. This work is definitely a basis to promote the use a more complete CVSS score.

*The gap.* So far CVSS represents the risk of the specific vulnerability on the specific device, it does not embed the risk level of attack graphs passing through it and the targets they can reach.

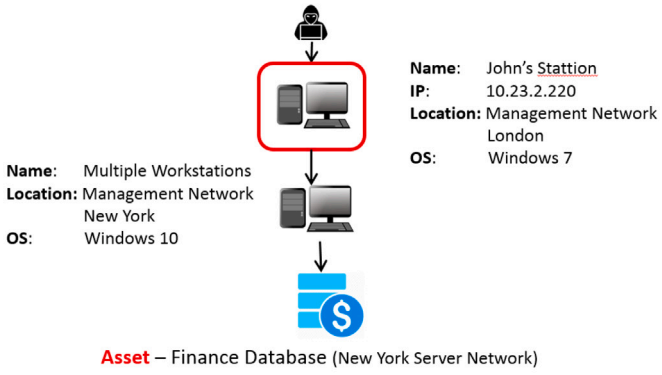
#### 2.1.2. Attack graph level scoring

Previous work was done to define metrics that will embed the risk from attack graphs. Frigault and Wang (2008) presented a Bayesian network approach to measure the overall security of the network. Using probability scores and the dependency between the different steps in the graph, they compute the overall probability of the attacker to reach his target given a specific attack graph.

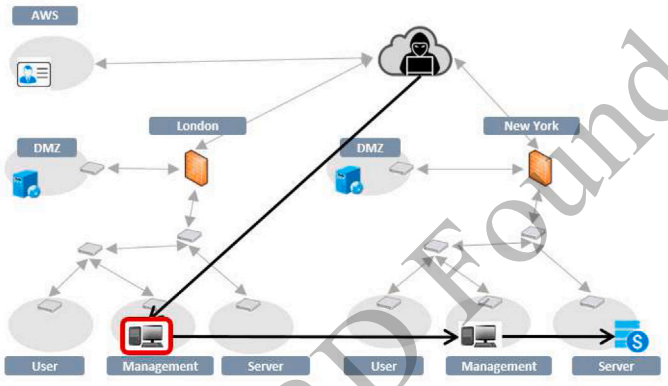
Idika and Bhargava (2010) studied the shortcoming of computing the risk from an attack graph based on the individual metrics: Shortest Path metric, the Number of Paths metric, and the Mean of Path Lengths metric. They suggested an algorithm that compares attack graphs found in the network and is able to rank their relative security risk.

Noel and Jajodia (2014) suggested a suite of attack graph security metrics that would summarize overall network security risk. They defined four families of metrics:

- Victimization: the relative number of victims in all the attack graphs, and the average exploitability and impact value from CVSS metric of the network vulnerabilities that are exploited.
- Size: the size of the attack graph. The premise is that the larger the graph, the more ways the network can be compromised.
- Containment: organizations divide their networks into different domains or subnets (segments). This family represents the risk from attacks crossing these segments.
- Topology: this family takes into account the connectivity allowed on the network between devices. The cycles this graph creates and the depth in the amount of steps an attacker needs to perform from any starting point to its target affect the difficulty of an attack.




(a) Logical view: Details on the assets of the path are displayed next to each asset. The focus of the visualization is an easy to understand display of the attack path, and the device's role in it.




(b) Topological view: The assets on the path are represented in their topological context on the map. The focus of the visualization is to display the attack path and the device in the context of the network topology, providing deeper information.

**Fig. 1.** Two suggested views of a possible attack. An attacker exploits John's station as a first step of an attack path to the finance server.



Each family is summarized and an overall security metric that can be monitored over time to understand the value of different configuration changes, and software patches on the network security.

*The gap.* All of the above metrics focus on calculating the security risk level of an entire network or an attack graph. Our visualization requires a metric that can be computed per vulnerability while taking into account the entire network. Such contextual device specific metric can be used to prioritize the vulnerability patches. In our metric we focus on the number (or portion) of attack paths passing through the vulnerability, and the number of steps each path has. The main motivation that drives our study is that a vulnerability that is exploited in multiple short paths to a critical target should get a higher priority than a vulnerability that is exploited by fewer long attack paths toward the organization's critical assets.

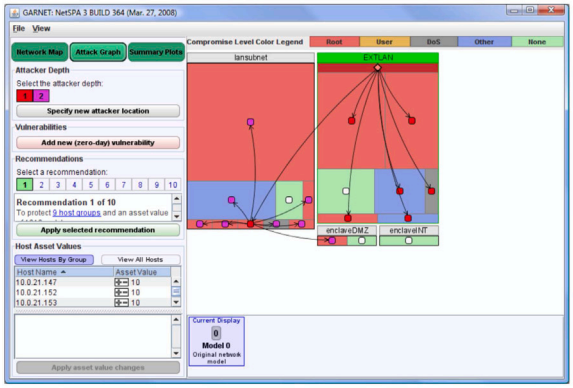
### 2.2. Attack graph visualization

An attack graph (Ou et al., 2006) is a data structure that represents all the steps an attacker would take from his original state until the desired target is compromised. Typically graphs represent the chain of states (nodes) and actions between states (edges) an attacker needs to perform trying to reach the target asset. An attack path is a higher level representation of the graph, that focuses on the sequence of devices the attacker needs to exploit to reach his target, consolidating the different steps inside a device. The following are related work done to use attack path representation to create an application that would help experts prioritize vulnerabilities on their network.

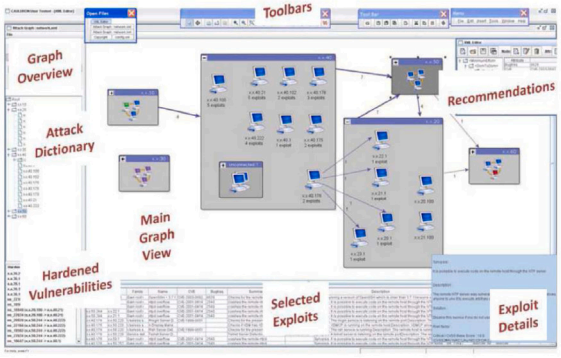
Williams et al. (2008) created the GARNET visualization tool, that allows users to view attack paths in the network. The visualization focuses on displaying the different sub networks of the organization as seen in Fig. 2a, and how attack paths begin in one subnet and may spread to other subnets. In addition, they also provide metrics that determine the security state of a network, according to the number of hosts that can be compromised. They performed a heuristic evaluation of their tool with 5 domain experts and used their usability comments to improve the tool.

Jajodia et al. (2011) created the topological vulnerability analysis (TVA) approach. They presented a scalable system that creates a multi-level abstraction that combines a huge number of different attack graphs into a few high-level graphs, allowing for easier understanding and usage. They suggest a visualization that displays these high-level graphs as seen in Fig. 2b.

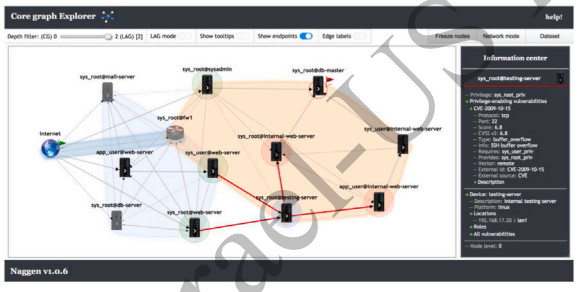
Barrère and Lupu (2017) presented the Naggen tool (network attack graph generation). Their tool, seen in Fig. 2c simplified the attack graph view by representing what they call core graphs. In the core graph a single edge between two nodes shows that there is at least one route in the attack graph between these two nodes. In many cases there are many routes which are collapsed to a single edge representation. In this way, Naggen allows a simpler high-level representation of the potential attack graph. They created a user interface that allows one to view the high-level core routes, and drill down into the different routes that represent each core edge.



(a) GARNET(Williams et al., 2008): Network subnets arranged in grid layout with weighted sizing. Arrows indicate attack paths between different networks.




(b) CAULDRON(Jajodia et al., 2011): An implementation of the TVA (Jajodia & Noel, 2010) system, displays all possible paths through the network. Hosts that share the same vulnerabilities are grouped to reduce the number of paths displayed.




(c) Naggen(Barrère & Lupu, 2017): displays core graph attacks on the network. All attack paths, and every host on the path are displayed.

**Fig. 2.** Previously suggested user interfaces and visualizations of attack graph.

*The gap.* All the above visualizations focus on the attack graphs as the starting point for the security admin. We believe that in mid to large organizations this approach is sub-optimal. Such organization's security teams have to handle dozens to thousands of vulnerabilities on their network. They may not be able to devote the time and skill required to compare attack graphs visually and derive conclusions. We believe these teams need a simple interface that does a pre-processing of the attack path information and presents a clear prioritized list of issues to resolve while on the same time providing the required justification of the priorities. Table 1 summarizes the differences between the above related work and our solution.

### 2.3. Commercial applications

Tenable.io,[1] Qualys Vulnerability Management[2] and Nexpose[3] are the three most popular enterprise level vulnerability management systems (Gartner, 2019). All three solutions enhance the CVSS score with some internal research and create a severity score. All of them present the data in different report dimensions (by device, by vulnerability, by severity and more). None of them displays the attack graph or use its analysis to help prioritize vulnerabilities. Tenable seen in Fig. 3 is the commercial version of the most popular open source vulnerability scanner — Nessus.

## 3. Background

We now review relevant background, starting with attack graphs and their use in ranking vulnerability patches. We then discuss graph centrality measures, and the planning graph data structure. In the area of visualizations, we discuss solutions that provide a wider context to an attack. We also discuss the use of prototypes in early stage research.

### 3.1. Logical attack graphs

Logical attack graphs (LAGs) represent the possible actions and outcomes of actions applied by an attacker trying to gain a goal asset in a system (Ou et al., 2006). An example of an attack graph can be seen in Fig. 4.

The graph contains 3 types of nodes: *Primitive fact nodes* represent facts about the system. For example, they can represent network connectivity, firewall rules, user accounts on various computer and more. In the example graph (Fig. 4) primitive fact nodes are represented by rectangular nodes.

*Derivation nodes* (or action nodes) represent an action the attacker can take in order to gain a new capability in the system. Action nodes are represented in Fig. 4 by ovals.

*Derived fact nodes* (or privilege nodes) represent a capability an attacker gains after performing an action (derivation phase). For example, a node stating that the attacker can execute arbitrary code on a specific machine with administrator privileges. Derived fact nodes are represented by diamonds in Fig. 4.

Edges in the LAG from a fact node to an action node represent the dependency of the action on the facts, and edges from an action to fact represent the derivation of that fact following the action.

**Definition 3.1.** Logical attack graph. Formally, a logical attack graph is a tuple: $G = (N_p, N_a, N_f, E, L, g)$ Where $N_p$, $N_a$ and $N_f$ are three sets of disjoint nodes in the graph, $E$ is a set of directed edges in the graph where

$$E \subseteq (N_a \times N_p) \cup ((N_p \cup N_f) \times (N_a)) \quad (1)$$

$L$ is a mapping from a node to its label, and $g \in N_p$ is the attacker's goal (multiple goals can be transformed into a single goal using an action with preconditions as the multiple goals). $N_p$, $N_a$ and $N_f$ are the sets of privilege nodes, action nodes and primitive fact nodes, respectively.

[1] https://www.tenable.com/products/tenable-io.
[2] www.qualys.com/apps/vulnerability-management.
[3] https://www.rapid7.com/products/nexpose.

**Table 1**
Comparison of our approach to other related work.

| Visualization | Garnet (Williams et al., 2008) | Cauldron (Jajodia et al., 2011) | Naggen (Barrère & Lupu, 2017) | Our work |
|---|---|---|---|---|
| Network display | High level blocks of the different networks | Groups of devices with the same vulnerability/exposure | All devices that participate in any graph are presented | "Network Topology" display or "Logical View" Display |
| Node | Devices on an attack path | Groups of devices that have the same vulnerability or exposure | Devices on an attack path | Devices on an attack path |
| Edges | A step in an attack path between two nodes | | | |
| Support Prioritization | Users can choose to focus on the edges that allow the attacker to jump between networks (blocks) | Displaying hosts or groups on the highest amount of paths to the most important assets. | Displaying the device on the highest number of paths to the most important assets. | The prioritization is calculated for the user and is presented in a sorted list, using the combination of CVSS and NTVS |
| Prioritizing vulnerability patches in large networks | The visualization of blocks scales well. Information overload in displaying dozens of edges crossing between the blocks. | Grouping may not be sufficient in networks with thousands of devices, leading to a cluttered display. Difficult to understand the important vulnerabilities. | The visualization supports prioritization only in a small networks. In networks with hundreds of devices the information overload is significant. | The prioritization is pre-computed and presented in a sorted list. |

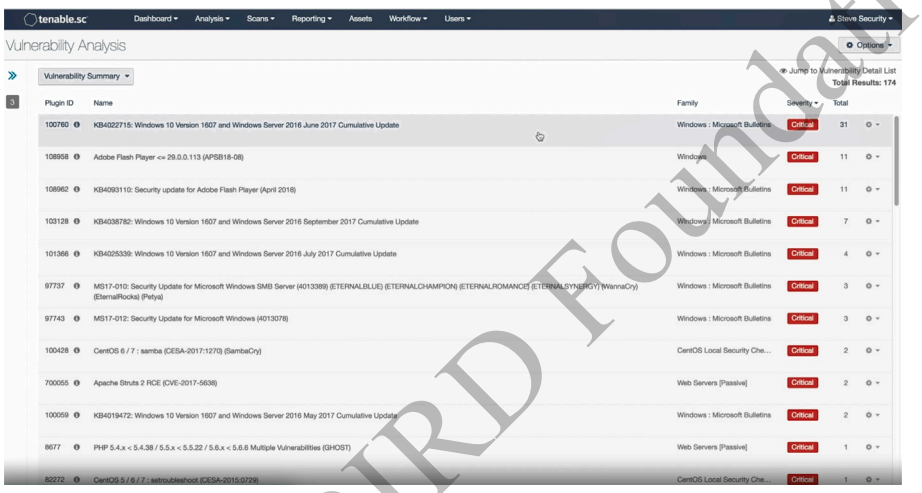



**Fig. 3.** Tenable.sc (Tenable, 2020): This view shows a list of vulnerabilities sorted by their severity. Rapid7 and qualys have similar displays.

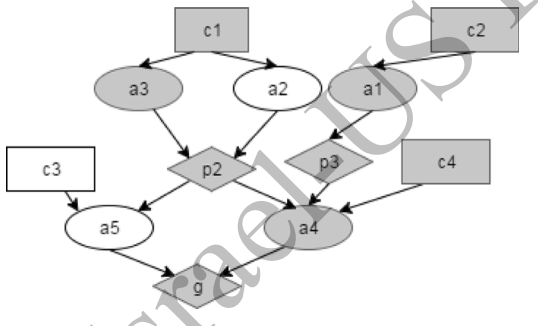



**Fig. 4.** Example of an attack graph with an attack plan (grayed nodes).

The edges in an LAG are directed. There are two types of edges in attack graph: $(a, p)$ an edge from an action node to a derived fact node, stating that by applying $a$ an attacker can gain privilege $p$. $(p, a)$ is an edge from a fact (either primitive or derived) node to an action node, stating that $p$ is a precondition to action $a$. For example, in order to apply exploit $e$ on machine $m_2$ from machine $m_1$, there must be a connection from $m_1$ to $m_2$ (represented by a primitive fact node $p$), and the user must have already gained access to code execution on $m_1$ (represented by a derived fact node $d$). Hence, there will be edges from $p$ to $e$ and from $d$ to $e$. In addition, if using exploits $e$ results in obtaining code execution privileges on $m_2$, represented by a derived fact node $c$, then there will be an edge from $e$ to $c$.

The labeling function maps a fact node to the fact it represents, and an action node to a rule that defines the derivation of new facts. Formally, for every action node $a$, let $C$ be $a$'s child node and $P$ be the set of $a$'s parent nodes, then $(\wedge L(P) \Rightarrow L(C))$ is an instantiation of interaction rule L(a). LAGs are a special case of And/Or Graphs (De Mello & Sanderson, 1990) where each action can instantiate only one fact (or derived fact). We will use this notation from Gefen and Brafman (2012)

- $pre(a) = \{v \in N_p \cup N_f : (v, a) \in E\}$
- $add(a) = \{v \in N_p : (a, v) \in E\}$
- $ach(v) = \{a \in N_a : v \in add(a)\}$

where $pre(a)$ is the set of facts which are preconditions to the action $a$. $add(a)$ is the set of facts gained by applying the action $a$ (in LAGs this set contains only one node). $ach(v)$ is set of actions which can achieve derived fact node $v$.

An attack plan $G'$ is a subgraph of $G$. The attack plan must hold the following:

- $g \in G'$
- $\forall a \in N_a : pre_G(a) \subseteq G'$
- $\forall v \in N_p : \exists a \in ach_G(v)$ s.t. $a \in G' \wedge |ach_{G'}(v)| = 1$

That is, an attack plan is a sub-graph of G′ that contains the goal node of graph G. Each action $a$ in G′ is fulfilled by all of the preconditions of $a$ in G. Each fact is achieved by exactly one action. Attack plan represents a scenario in which an attacker infiltrates the organization and achieves his goals.

Attack graphs have been used to depict possible ways for an attacker to compromise a computer network. Initially, attack graphs were used to better visualize the paths an attacker can take in the network. DARPA created attack graphs manually as part of a red-team analysis (Swiler et al., 2001), but once attack graphs could be generated automatically, they were also used to improve the security of the networks.

A common use of attack graphs is to find a set of vulnerabilities to patch which will prevent the attacker from reaching the goal. Work in this area uses various methods such as minimum-cost SAT solving (Huang et al., 2011) and specialized methods (Albanese et al., 2012). In practice, even after finding such a minimal set of vulnerabilities, the cost of patching them may still be prohibitively high given limited IT resources. Our goal in this project is to identify methods to rapidly reduce the number of possible attack plans, when it is not possible to completely prevent the attacker from reaching the goal.

Many researches also assume some cost metric on actions in the attack graph (Obes et al., 2013), corresponding to the time it takes to launch an exploit, or to the risk of detection. It is also common to measure the probability of success when performing an exploit action (Wang et al., 2008). Researchers then try to find attack plans which minimize/maximize the suggested metric, assuming that a rational attacker will first try to launch attacks that minimize cost (Sommestad & Sandström, 2015).

For example, researchers have suggested to use classical planning, MDPs, and POMDPs to create good or optimal attack plans (Hoffmann, 2015; Shmaryahu, 2016). Assuming that the attacker optimizes the attack plans, in this paper, we focus on shortest attacks only, in an attempt to cut as many of them as possible and eventually increase the cost of an attack.

### 3.2. Graph centrality measures

Centrality measures attempt to estimate the importance of a node within a graph (Freeman, 1977). This is useful in many domains, such as in finding prominent members in social networks, identifying bottlenecks routes in traffic networks, and many more. Previous research on ranking vulnerability patches has used several centrality measures in order to identify which vulnerabilities should be patched (Hong & Kim, 2013).

The most obvious centrality measure is degree centrality, $C_D(v) = degree(v)$, mainly because it is easy to compute. Although it is easy to compute, degree centrality often poorly represents the true importance of a node in a graph.

Betweenness centrality captures a more delicate aspect of the importance of the node in a given graph. This measure represents for each node, the number of shortest paths between any two nodes that passes through that node:

$$C_B(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}} \tag{2}$$

where $\sigma_{st}$ is the number of shortest paths between nodes $s$ and $t$, and $\sigma_{st}(v)$ is the number of shortest paths between nodes $s$ and $t$ that pass through node $v$. Betweenness was previously used to find important nodes in attack graphs (Hong & Kim, 2013).

Below, we also use a modification of betweenness, in which we consider only shortest paths between a set of source nodes (the initial facts nodes in the LAG, or the first layer of the planning graph) to the goal node (Brandes, 2008). We denote this betweenness *s-t* betweenness, and the original betweenness is denoted *all-pairs* betweenness.

Another commonly used graph centrality measure is Closeness Centrality. This measure captures how close is a certain node to the rest of the nodes in the graph.

$$C_C(v) = \frac{1}{\sum_u d(u, v)} \tag{3}$$

where $d(u, v)$ is the shortest distance between $u$ and $v$. In this centrality method, nodes on the fringe of the graph have lower scores than nodes in the center of the graph.

Google's PageRank has also been used to rank important nodes in a graph. PageRank is measuring the likelihood for a web-surfer to be at page $i$ (Page et al., 1999), for estimating the importance of web pages. The metric is given by:

$$C_{PR} = \frac{1-d}{N} + d \sum_{j \in In(j)} \frac{\pi_j}{|Out(j)|} \tag{4}$$

where $N$ is the number of nodes in the graph, $Out(j)$ are the outgoing neighbors of $j$, $In(j)$ are the ingoing neighbors of $j$, and $\pi_j$ is the probability that the web-surfer will be at nodes $j$. $0 < d < 1$ is a damping factor, representing how likely a web surfer will get bored and move to another web page which is not directly linked to the current node.

Similar to current study, some researchers propose to use network centrality measures to find the vulnerabilities to patch first. Some of these measures are applied on a two-layered attack graph model (Hong & Kim, 2013; Hong et al., 2014; Sawilla & Ou, 2008). The nodes in the first layer, the *connectivity graph*, represent the hosts in the system. A directed edge between two nodes $(a, b)$ means that when controlling node $a$, an attacker is able to subvert the node $b$ using some exploit. The second layer is an AND–OR tree containing all the ways to compromise a machine from an arbitrary other machine. The authors compute different network centrality measures on the connectivity graph to find the computer to patch. The authors also proposed using a measure called Attacker victim centrality (AVC) (Hong & Kim, 2013), based on betweenness centrality, considering only host pairs near the attacker and the target.

### 3.3. Planning graphs

A planning graph (Blum & Furst, 1997) is a data structure originating in the automated planning community. It is a directed, layered graph with two types of nodes and two types of edges. The layers alternate between fact layers, containing only fact nodes, and action layers containing action nodes. Edges from fact to action nodes denote action preconditions, while edges from action to fact nodes denote action outcomes.

In classical planning problems, already obtained facts can be removed by other actions. Planning graphs hence include additional information, such as which facts cannot be achieved at the same time (mutexes) (Bryce & Kambhampati, 2007). However, in penetration testing, once a fact, such as access to a given machine, is obtained, it is never lost, resulting in simpler graphs that are easier to represent and reason about. We also note that the number of nodes in such *delete-free* planning graphs is $O(n^2)$ in the worst case, where $n$ is the number of nodes in the respective LAG. A low polynomial computational effort executed daily seems reasonable in networks containing thousands of machines.

The planning graph is constructed incrementally. The first layer of the relaxed planning graph is a fact layer, and contains one node fact that is true initially. The next layer is an action layer, containing all actions that can be executed using the facts at the previous layer. That is, all actions whose preconditions appear in the previous layer. The third layer contains all the effects of the actions at the second layer.

When no new facts appear in a fact layer, the graph expansion can be stopped. In our case, as we care only about shortest attack plans, we can stop once the goal fact appears in the graph.

In addition, we add for each fact $p$ a special *no-op* action, that takes $p$ as precondition, and generates $p$ as output. Hence, each fact layer is a superset of the preceding fact layer. Once no new facts have been obtained in a fact layer, we can stop the expansion of the planning graph.

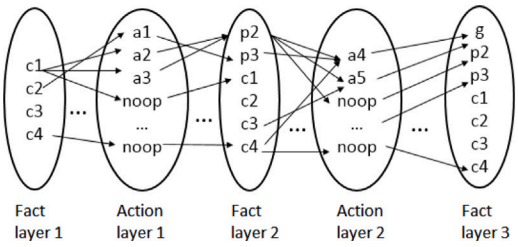



**Fig. 5.** Planning graph of graph G from Fig. 4.

Facts often appear in multiple layers in the planning graph — once a fact has appeared at layer $i$, it will appear in all fact layers $j > i$, through the no-op actions. We denote each fact by its layer, that is, for fact $p$ at layer $i$, we write $p_i$. As in each fact layer at least one new fact must be added, the number of fact layers cannot exceed the number of facts. Typically, several new facts are added at each layer, and the depth of the planning graph is in many practical problems much lower than the number of facts. Symmetrically, the depth cannot exceed the number of actions Fig. 5 shows a planning graph for the graph $G$ presented in Fig. 4. We omit some of the edges between the facts and *no-op* actions for ease of presentation.

In this paper, we suggest analyzing the planning graph, replacing the standard centrality measures applied on the computer connectivity graph or on the attack graph with a similar analysis of the planning graph.

## 4. The vulnerability prioritization framework

Our approach consists of a new score and a new visualization that combine to present a more comprehensive and clearer view of network vulnerabilities. First in Section 4.1 we discuss the attack plan enumeration method which is an important building block in the proposed scoring system. Next we explain the proposed NTVS score in Section 4.2 and then in Section 4.3 we discuss a visualization of CVSS and NTVS, alongside attack path representation to facilitate the decision making.

### 4.1. Enumerating all attack plans

In this paper we focus on the task of ranking vulnerability patches in order to rapidly reduce the number of shortest attack plans in an any-time approach where patches are applied one at a time. In order to evaluate the patching strategies we first identify all the available shortest attack plans.

We now explain how one can use the planning graph in order to enumerate all possible shortest attack plans. This process is exponential, and cannot be applied to larger planning graphs. It is useful, however, in order to evaluate the performance of the various centrality measures.

We analyze the planning graph, rather than the LAG, because LAGs contain cycles, which are avoided in the planning graph due to *no-op* actions and fact duplication. We use a BFS-style algorithm, moving backward from the goal node $g_n$ at the last fact layer $n$, described in Algorithm 1.

When traversing backwards, we maintain a set of plans. For each plan there is a set of unsatisfied facts, initialized with the goal. To expand a plan backwards from layer $i$, for each unsatisfied fact $p_{i+2}$, we identify an action $a$ (possibly a no-op) that has $p$ in its effects. We remove $p_i$ from the list of unsatisfied facts, and for each fact $q$ in the preconditions of $a$ we add $q_i$ to the set of unsatisfied facts. If $a$ provides an additional unsatisfied fact $r_i$, it is also removed from the list of unsatisfied facts. That is, we will not search for another action $a'$ to satisfy $r_i$.

There can be many potential actions that satisfy a needed fact $p$, each corresponding to a different plan. Thus, for each action $a$ that satisfy $p$ we create a copy of the plan and add $a$ to the copy. Thus, the expanded plan is split into multiple similar plans, with a single different action.

**Algorithm 1:** Enumerating All Attack Plans

```
EnumeratePlans(PG, t) :
    Input: Planning graph PG, target node t
    Output: Set of all the attack plans in the graph
    i ← lastLayer(PG)
    P_i ← {⟨⟩} // Solution plans
    pre(⟨⟩) ← t //precondition of the empty plan
    while i > 0 do
        P_{i-2} ← ∅
        foreach plan p = ⟨a_1, ..., a_k⟩ ∈ P_i do
            unsat(p) ← unsatisfied f ∈ pre(p)
            foreach minimal action set A at layer i − 1 s.t.
             unsat(p) ⊆ eff(A) do
                p' ← A · p
                pre(p') ← pre(p) ∪ pre(A)
                add p' to P_{i-2}
        i ← i − 2
    return P_0
```

More precisely, let $P_i$ be the set of unsatisfied facts of the expanded plan at layer $i$, and $\mathcal{A}^P_{i-1} = \{a : \exists p \in P_i, p \in effects(a)\}$ be the set of actions at layer $i-1$ that satisfy at least one fact in $P_i$. We create a copy of the plan for each minimal subset $A^P_{i-1} \subseteq \mathcal{A}^P_{i-1}$ such that $P_i = \bigcup_{a \in A^P_{i-1}} effects(a)$, and add $A^P_{i-1}$ to the copy.

Once we have reached the initial layer we have enumerated all possible shortest plans, but possibly also some longer plans. Let $\Pi$ be the set of all such plans. $\Pi$ may contain some redundancies, due to the use of no-ops. More specifically, given $P_i = \{p_i, q_i\}$, and two actions, $a_p$ and $a_q$, that produce $p, q$, respectively, we may have 4 different alternatives — $\langle a_p, a_q\rangle$, $\langle a_p, noop_q\rangle$, $\langle noop_p, a_q\rangle$, and $\langle noop_p, noop_q\rangle$ for expanding the plan backwards. Then, at layer $i-2$, we can choose $a_p$ where $noop_p$ was selected and $a_q$ where $noop_q$ was selected. Ignoring the no-ops, which are not real actions to be executed, we obtain 4 identical plans. To remove such duplicates, once we have obtained the set of all plans, we remove no-ops from all plans, and then remove duplicate plans, ignoring the action order within a plan. We now remove all plans which are longer than the shortest plan.

Using the above planning graph construction and plan enumeration method only yields plans with a finite number of actions (which is the number of action layers in the planning graph). In order to allow plans with various lengths, additional edges should be added to the planning graph between the final fact layer and the final actions layer. In this paper we focus only on shortest plans, and ignore longer plans.

This process is obviously $np$ hard, but in the real world graph that we have obtained, it runs sufficiently fast to be useful. We use the resulting set of shortest plans to evaluate the usefulness of the various centrality metrics. However, one may use a set of shortest plans also to, rank vulnerabilities according to the number of shortest plans. The more attack plans require a vulnerability or an action the more important it is. To do so we can count for each action $a$ the number of plans in which $a$ participates:

$$C_\Pi(a) = |\{\pi : \pi \in \Pi, a \in \pi\}| \tag{5}$$

In the future, we will explore sampling techniques, originating from research in AND–OR graphs, to rapidly provide a sample of shortest plans.

### 4.2. Network Topology Vulnerability score

To focus on the risk that can be deduced from attack graphs, we introduce a new metric called NTVS (Network Topology Vulnerability

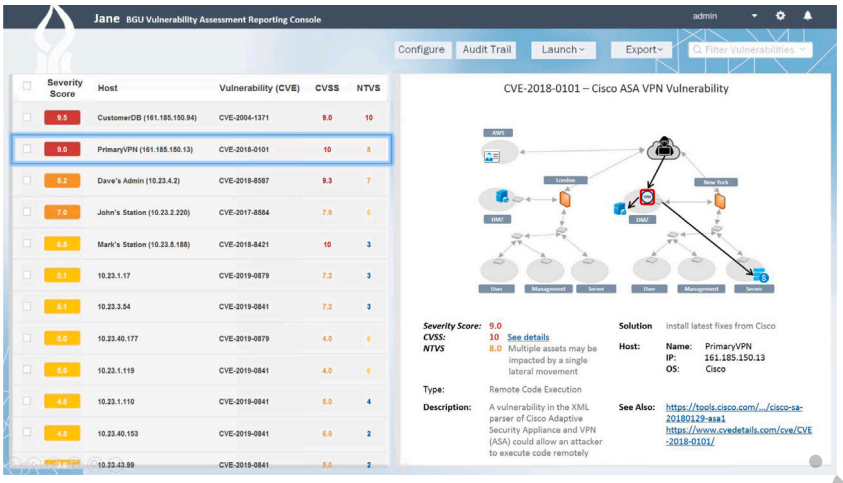



**Fig. 6.** Jane: our suggested visualization with a high granularity network display. On the left, Jane lists the vulnerabilities found on the organization network, sorted by decreasing severity. We also display the CVSS and NTVS that are used to compute the severity. The user can choose a vulnerability on the left and see its details on the right detailed pane. In the detail pane, the user sees details about the vulnerable device and its vulnerability. In addition, the detail pane includes a visualization of the attack path, based on the topological map of the organization's network.

score). The main intuition behind NTVS is to expose the potential of using any vulnerable device as a step in a path toward a critical asset of the organization. NTVS is based on the number of attacks that pass through a specific vulnerability on a specific device.

The metric has values in the range $[0, 10]$, where 0 denotes a vulnerability that does not help the attacker to subvert a critical asset. An example of a 0 NTVS score is a denial of service attack on a non-critical asset in an isolated lab environment. While the attacked machine is no longer operational due to the attack, it is not a critical assess and cannot not facilitate attack propagation.

An NTVS value of 10 denotes a direct attack on a critical asset. For example, an attacker directly exploiting the denial of service vulnerability on a critical web service provided by the organization to its users.

In other cases an asset acquired by the attacker may facilitate attack propagation through by enabling discovery of additional assets, lateral movement, privilege escalation, allowing to evade defenses etc. The actual NTVS score of a vulnerability is determined by the number of substantially different attack paths from the device to the critical assets, and by the number of steps on the shortest path found. To compute the metric we first rank the vulnerabilities following Gonda et al. (2018). Given the ranking, we assign a value of 10 to vulnerabilities that are on critical targets and have a direct attack path from the attacker location (usually the Internet). We assign a value of 0 to all the vulnerabilities that are not on any path to a target. For the rest of the vulnerabilities we assign a value in the range of $[1, 9]$ by normalizing the ranking ordered list $\frac{(R-r)}{(R-1)} \cdot 8 + 1$ where $R$ is the number of vulnerabilities in the ranking list and $r$ is the item's position in the list.

The suggested NTVS metric allows focusing the security experts on a specific device and its vulnerabilities, while taking into account the additional risk that the vulnerability on the specific device poses, when it is exploited, for reaching a critical target.

### 4.3. Vulnerability focused attack path visualization

We now describe the second main contribution of this paper — the novel visualization designed to present to the security expert the information summarized in the NTVS score in a clear and compact manner.

Similar to commercial applications, our suggested visualization lists the vulnerabilities by decreasing order of importance, to allow experts to immediately focus on the vulnerabilities that are deemed most crucial to eliminate. While other applications typically list vulnerabilities by decreasing CVSS score, we rank vulnerabilities by decreasing the average of CVSS and NTVS, and displaying the two scores side by side. We use a color code from red (most critical) through orange (important) to blue (low risk).

While encapsulating the vulnerability information in a single score is important for rapid prioritization of vulnerability patches, it is also important to present additional information that can help to convince the experts that an action needs to be taken. This information should be presented in a useful manner to allow the expert to rapidly understand the information. We hence suggest a visualization designed to help the security expert understand the values provided by the NTVS score. The visualization displays the position of a given device within the network, with relation to the entry point of an attack and the critical assets in the network.

In case the vulnerability allows the attacker to get control over the vulnerable device it displays the possible attack paths from the vulnerable device to any critical asset. We display this information without changing the orientation of the security expert from his view of devices and vulnerabilities. We created two flavors for these visualizations: topological view and logical view.

#### 4.3.1. Topological view

This view, seen in the detailed pane of the Jane application (Fig. 6) shows all subnets on the network topology graph. When a user selects a specific vulnerability (on the left pane), the system highlights the relevant device. In addition, the system highlights possible attack paths that pass through the selected device. This allows the user to rapidly see whether compromising this device may lead to an attack on a critical asset.

For example, in Fig. 6, the user has chosen the Cisco's VPN vulnerability (CVE-2018-0101) on the left list. In the detailed pane on the right, the application displays information about the vulnerable device, the vulnerability details and the NTVS score and its topological visualization. The visualization displays the two attack paths that exploit the VPN in the DMZ network on its way to attack two different critical assets (the corporate web site which is on the same DMZ network and a finance servehr hosted on the Server network). All assets are displayed on their topological location on the map, as in the Cauldron Tool (Fig. 2b). The network is split into protection domains. Protection domains are sets of hosts with unconstrained access to one another. The connectivity between the protection domains is based on the network

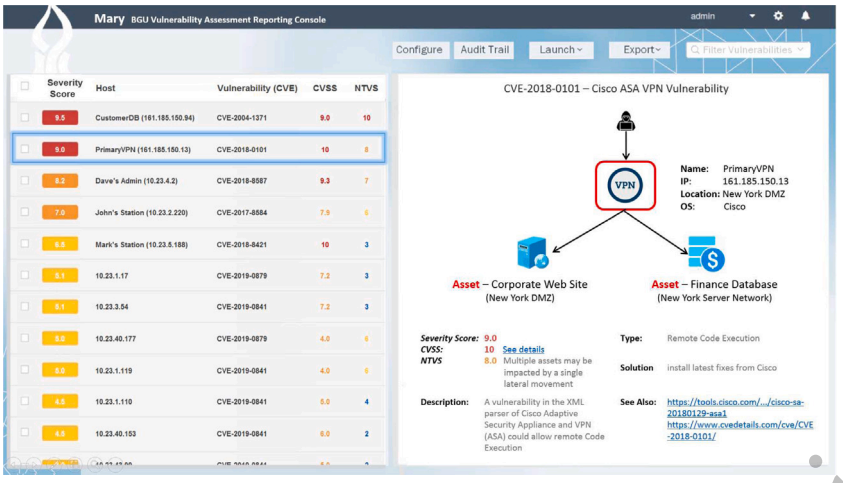



**Fig. 7.** Mary: our suggested visualization with a low granularity network display. Mary's interface is almost identical to Jane's (Fig. 6). The one difference is the visualization of the attack path in the detail pane, which is presented without the network topology context.

routing analyzed from the different routers switches and firewalls, as done by many commercial tools such as Skybox.[4]

#### 4.3.2. Logical view

This view, seen in the detailed pane of the Mary application (Fig. 7), shows a summarized attack path visualization from the attacker through the vulnerable device and all other steps needed until the attack reaches the organization critical assets. For example, in Fig. 7, with the same vulnerability chosen above, the visualization shows only the assets involved in the attack path without the context of the topological map.

In case of many possible attack paths we can use tools such as the TVA model (Jajodia & Noel, 2010) to abstract and combine the multiple paths into a few high-level paths according to the protection domains. In the case of a large and complex network, the topological view may require an additional abstraction beyond the protection domains to present a high-level view of the network without losing the orientation it provides. This could be achieved with network modeling as suggested by Skybox (Cohen et al., 2012).

## 5. Network data acquisition

All evaluation in previous research in pentesting via attack graphs is limited, as far as we know, to artificial simulated networks. It is obviously desirable to evaluate new approaches over real networks. We hence created realistic models using data obtained from scanning the computer networks of two organizations, each containing a few subnets (Gonda et al., 2017). Using the machine configurations and existing exploits discovered during the scan, we can create real world models that allow an empirical evaluation of our approach and comparison to state-of-the-art. We now provide some explanations of the model and the networks, unfortunately omitting many details due to confidentiality restrictions.

To collect the information required to build the attack graphs, we began with scanning the various subnets using the Nessus scanner (Beale et al., 2004) in each organization. Nessus starts its scan from some source host. It identifies all machines reachable from the source host (where the scan is running) including desktops, gateways, switches, and more, possibly through several switches and routers. We hence executed several such scans, each from a different subnet within the organization, as well as one scan from outside the organization network.

The resulting scans contain the set of machines that are visible from a representative host in each subnet. Naturally, machines inside a subnet are all visible to each other. We assume that all machines within a subnet can directly access the machines that are visible from the representative host (the one with the Nessus scanner) in that subnet. This assumption may not be true when firewall rules allow or deny communication between specific IP addresses. In general, only a part of the machines outside the subnet are visible from within the subnet, due to, e.g., firewall restrictions. We model the accessibility of machines identified through the scans as direct edges in the connectivity graph. That is, machine $m_1$ is connected to machine $m_2$, if $m_2$ was revealed by a scan performed from the subnet containing $m_1$.

In addition, Nessus reveals for each identified host its operating system. Both scanned organizations contained hosts running Windows and Linux (with a few versions of each operating system). Nessus also identifies software with potential vulnerabilities that run on the scanned machines. The attack graph we created contains approximately 50 types of such software, including well known applications such as *openssh*, *tomcat*, *pcanywhere*, ftp services, and many more.

Nessus identifies vulnerabilities of varying importance. For the purpose of this study we ignored all the lesser vulnerabilities, which do not allow an attacker control of the system. For example, the vulnerability identified as CVE-2014-6271 (dubbed ShellShock) allows unauthenticated remote attackers to execute arbitrary code on a vulnerable server by sending a specially crafted packet. On the other hand, the vulnerability identified as CVE-2014-4238 allows remote authenticated users to affect availability of the system (Denial of service). We identified about 60 types of important vulnerabilities that exist in the scanned networks. We ignored all hosts that do not run any software for which an important vulnerability exists.

Next we marked a few random hosts in the innermost subnet, as the target hosts. The assumed attack goal is to gain control over at least one of these target hosts (see Fig. 9).

Sadly, we are currently unable to publish the network data. Understandably, modern organizations are concerned about revealing information concerning their network configuration, which might be useful for malicious hackers. It is not surprising, thus, that there is currently no publicly available network data containing all the required information, including network connectivity, machine configurations and relevant software. We are negotiating with the network administrator the publication of some of the data that we have collected, and hope that it would be useful for other researchers in the future.

[4] https://www.skyboxsecurity.com/.

**Table 2**
Network statistics.

| Domain | Shortest plans | | | | $\|V\|$ | $\|E\|$ | max $d(v)$ |
|---|---|---|---|---|---|---|---|
| | Count | Hosts | Nodes | | | | |
| Org1 | 1032 | 5 | 29 | CG | 24 | 236 | 21 |
| | | | | LAG | 1013 | 1604 | 102 |
| | | | | PG | 8439 | 9030 | 74 |
| Org2 | 960 | 5 | 29 | CG | 95 | 7222 | 92 |
| | | | | LAG | 17 523 | 25 940 | 105 |
| | | | | PG | 158 322 | 166 740 | 106 |
| Local +20 | 48 | 4 | 23 | CG | 25 | 74 | 20 |
| | | | | LAG | 394 | 560 | 22 |
| | | | | PG | 2754 | 2920 | 21 |

**Table 3**
Spearman correlation between a centrality measure and the number of plans containing a vulnerability. Computation time is in seconds.

| Graph | Centrality | Org1 | | Org2 | | Local+20 | |
|---|---|---|---|---|---|---|---|
| | | $\rho$ | Time | $\rho$ | Time | $\rho$ | Time |
| PG | All s-t betweenness | 0.94 | 3.61 | 0.93 | 4310.0 | 1 | 0.44 |
| PG | PageRank | 0.51 | 0.87 | −0.027* | 13.1 | 0.17 | 0.21 |
| PG | Closeness | 0.44 | 0.66 | 0.63 | 34.1 | 0.26* | 0.11 |
| CG | AVC | 0.29 | 0.05 | 0.96 | 2.01 | 0.67 | 0.03 |
| CG | All pairs betweenness | 0.28 | 0.03 | 0.96 | 2.12 | 0.39 | 0.01 |
| LAG | Closeness | 0.27 | 0.47 | 0.024* | 200.4 | 0.14* | 0.03 |
| | Random | −0.10* | 0.06 | 0.006* | 1.61 | −0.17* | 0.02 |

*Denotes $p$-value $> 0.05$.

## 6. Evaluation

We now present empirical evaluation of the two main contributions of this work — the computation of the number of attack plans that pass through a given vulnerabilities, and the presentation of this information to decision makers.

### 6.1. Comparing graph centrality measures

We begin with comparing the utility of various graph centrality measures in prioritizing the vulnerabilities to be patched. We focus on patching vulnerabilities to eliminate minimal attack plans.

That is, we use the various metrics computed over the different graphs to rank the vulnerabilities to be patched. We check which ranking method identifies vulnerabilities that are exploited by more attack plans and induces an increase in the cost of the minimal attack plan using fewer patches.

#### 6.1.1. Domains

Attack graphs available for research, such as data published in Durkota et al. (2015) and Hoffmann (2015), contain artificially created examples or simulated computer networks. We also explored a number of simulated attack graphs, but all of them are either too symmetric or too shallow (e.g. a single vulnerability per machine) to represent a real organization. As an example, we use the LocalPlus-20 dataset (Durkota et al., 2015) to demonstrate that conclusions derived from it are very different from the conclusions derived from an attack graph of a real organization.

To collect data we scanned the networks of two real organization as described in Section 5. The details of the networks are presented in Table 2. The hosts had a total of 144 important vulnerabilities which an attacker could leverage.

#### 6.1.2. Methods

Our goal is to identify centrality measures which most accurately estimate the number of attack plans that include exploitation of the various vulnerabilities. Given an attack graph we apply the centrality measures described in Section 3.2 on three representations of the network information: the connectivity graph, the logical attack graph, and the planning graph.

**Table 4**
The number of shortest attack plans after patching the most important vulnerabilities.

| Graph used | Metric | # of patches applied | | | | | |
|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 3 | 5 | 10 | 19 |
| Planning graph | Closeness | 1032 | 920 | 696 | 472 | 192 | **0** |
| | Betweenness | 1032 | 960 | 816 | 528 | 168 | **0** |
| | PageRank | 1032 | **903** | **645** | **387** | **0** | **0** |
| | AVC | 1032 | 1032 | 1032 | 1032 | 1032 | 744 |
| Logical attack graph | Closeness | 1032 | 1032 | 936 | 702 | 468 | 208 |
| | Betweenness | 1032 | 1032 | 1032 | 1032 | 1032 | 744 |
| | PageRank | 1032 | 1032 | 1032 | 1032 | 1032 | 744 |
| | AVC | 1032 | 1032 | 1032 | 1032 | 1032 | 744 |
| Connectivity graph | Closeness | 1032 | 920 | 696 | 472 | 360 | 360 |
| | Betweenness | 1032 | 920 | 696 | 472 | 72 | 72 |
| | PageRank | 1032 | 920 | 696 | 472 | 72 | 72 |
| | AVC | 1032 | 920 | 696 | 472 | 72 | 72 |
| Random | | 1032 | 1032 | 888 | 792 | 720 | 640 |

In the network connectivity graph (denoted CG below) nodes are computers, and vulnerabilities are not explicitly represented. Hence, for the connectivity graph we choose a computer based on the centrality measure, and patch all its vulnerabilities before moving to the next computer. In the logical attack graph (denoted LAG) and the planning graph (denoted PG), where vulnerabilities are explicitly represented as nodes, we directly rank the vulnerabilities to be patched using the centrality measures. Hence, the patching strategies often interleave vulnerability patches over different computers.

For each centrality method we begin with the original graph (LAG or planning graph), and compute the metric for all vulnerability nodes in the graph. We then rank all the nodes according to the centrality method by decreasing value.

All metrics are computed over both the LAG and the planning graph, except for the plan count, which is computed only over the planning graph. A vulnerability appears only once in the LAG, but numerous times on different layers of the planning graph. We thus rank the vulnerabilities in the planning graph by aggregating the centralities of all replicas of the vulnerabilities.

We compute the ground truth plan count by enumerating all attack plans as described in Section 4.1. As we have explained above, this process is computationally intensive for large networks, but for the real networks explored in this study we managed to enumerate the set of all shortest plans, and we use this set to evaluate how many plans are removed with each vulnerability that is patched.

Next we select a few centrality measures to demonstrate a patching strategy in both the network of a real organization and in the LocalPlus-20 network. Selecting the optimal subset of $k$ vulnerabilities to patch in order to block the maximal number of attack plans is a hard problem. This can be shown by a reduction from the minimal-set cover problem.

We apply a greedy heuristic by selecting the most central node. The vulnerability corresponding to this node is removed, and we recompute the respective attack graph (the connectivity graph, the LAG or the planning graph) without the removed vulnerability node. A host is removed from the connectivity graph when it no longer can be controlled by the attacker. The process terminates when all attack plans are blocked.

In addition, we enumerate the set of attack plans over the original planning graph using Algorithm 1. We identify the subset of plans with the minimal cost (shortest plans). Whenever we remove a vulnerability following the above procedure, we also remove all the shortest plans that use this vulnerability.

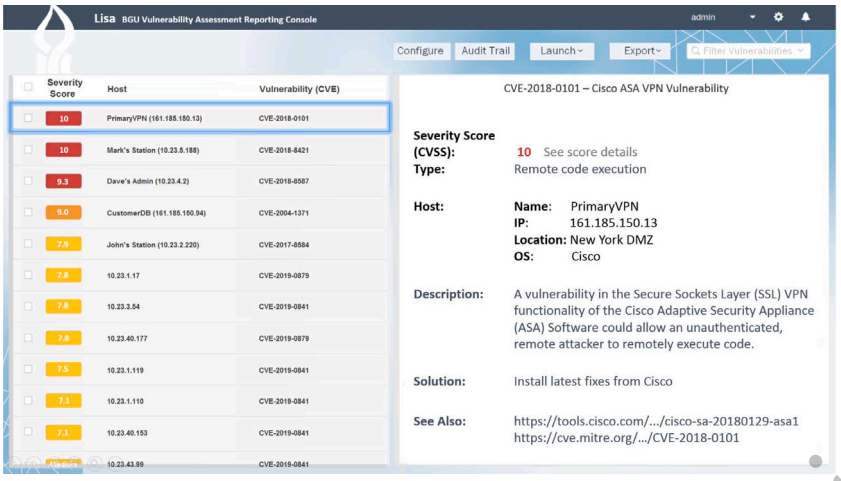



**Fig. 8.** Lisa: The commercial baseline user interface similar to the one provided by current commercial applications. The Commercial baseline does not use the NTVS metric. As such, the interface is very similar to Jane and Mary, with two main differences. First, the list on the left does not have the NTVS metric and is sorted by CVSS severity only. Second, in the detail pane on the right it provides only textual information.

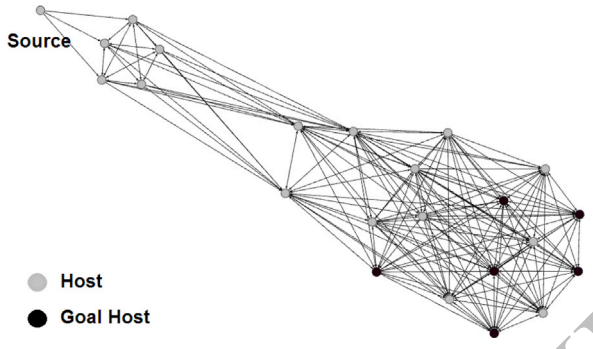



**Fig. 9.** Connectivity graph of a network. Each node is a host computer, a directed edge between two hosts $(u, v)$ means host $u$ can initiate a connection to node $v$.

### 6.1.3. Results

The use of graph centrality measures is based on the assumption that there is a strong correlation between the centrality score and the number of attack plans in which a vulnerability is used. We begin with evaluating this assumption. We hence computed the Spearman correlation between each centrality measure and number of shortest plans containing the vulnerability.

The results of the correlation analysis are presented in Table 3. We choose Spearman correlation for this analysis because (1) the scales of the different centrality measures vary and (2) the measures are used to choose the vulnerabilities to patch first and thus, ranking is more important than the actual values. The entries are sorted by the correlation significance and measures that perform worse than (or as good as) random were omitted for better clarity. All-sources single target Betweenness centrality computed on the planning graph has the strongest correlation with the number of attack plans (0.92) followed by Closeness and PageRank. State-of-the-art technique that achieves the closest performance is the AVC computed on the Connectivity Graph with $p = 0.025$. We can clearly see from Table 3 that the planning graph provides more useful information for estimating the number of plans in which vulnerabilities appear.

Next, we simulated the process of applying patches to vulnerabilities using the ranking of each centrality measure, estimating the reduction in the number of shortest attack plans following the policy dictated by the centrality measure (Table 4).

Fig. 10 presents the reduction in the number of attack plans as a function of the number of applied patches (removal of vulnerabilities) on the various domains. On all domains, the state-of-the-art AVC metric over the CG provides good results. It is interesting to see, though, that most metrics computed over the planning graph preform badly over the simulated Local+20 network, while providing the best results over the two real networks. On Org1 the best metric is PageRank over the planning graph, while on Org2 the best metric is Betweenness over the planning graph.

Over the 3 domains, no centrality measure applied on LAG produced competitive results. This is surprising given the popularity of LAGs in attack analysis research.

Org2 seems to be the most difficult network. On this domain most metrics failed to decrease the number of plans faster than a random selection. Looking at Table 2 we can see that this is also the network with the most complicated graphs in all representations, even though it has a similar number of shortest plans to Org1.

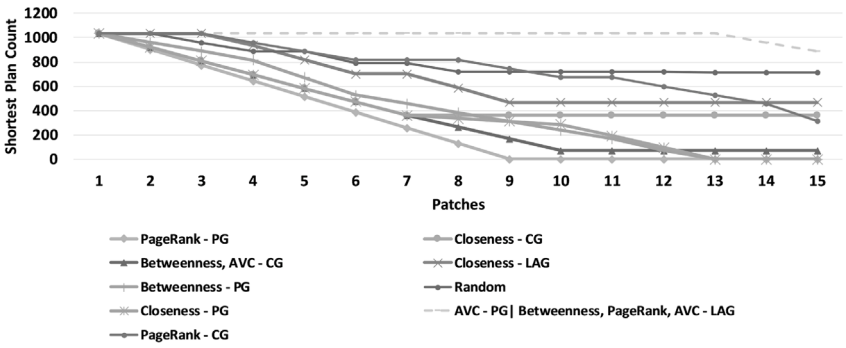
The artificial Local+20 network produces the lowest number of plans, as well as the simplest graphs. This further shows that artificial networks provide poor simulation of real world networks. The difference between the performance of the metrics over the real network and the simulated benchmark clearly present the urgent need for experiments with real world data. Simulated networks in this case may not model properly the real world, and benchmarks based on them may be misleading.

## 6.2. Expert study

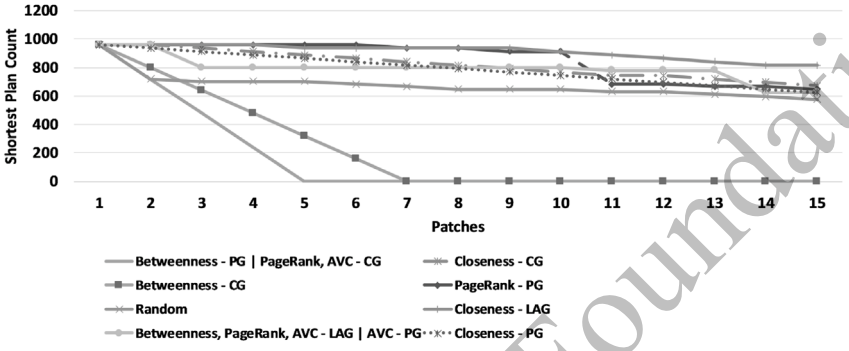
As we have seen above, the centrality methods allow us to rank vulnerabilities by their importance in the context of possible attacks. We now evaluate our suggested visualization techniques for presenting this information to decision makers.

To evaluate our proposed visualization, we performed an expert study with ten network security experts. The evaluation included three parts. In the first part we interviewed the experts regarding their background and the process they usually employ to detect network vulnerabilities in their organizations. Then, the experts performed prioritization tasks using the three user interfaces described above. Finally, the experts provided feedback (both structured and non-structured) on our system. Overall, the evaluation session lasted about 70 min. The evaluation study received an Institutional Review Board (IRB) approval.

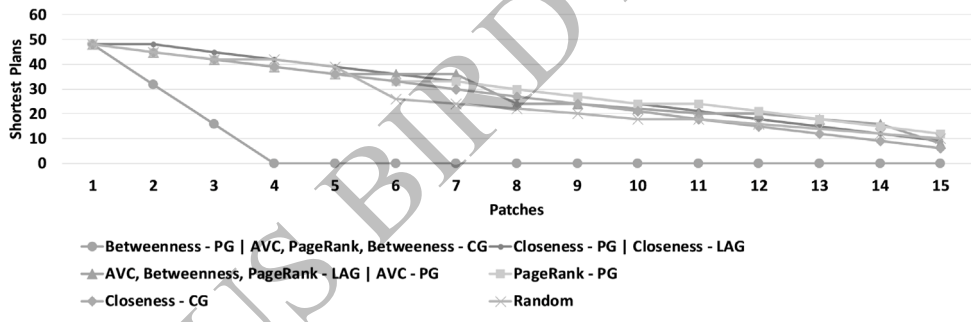
Naturally, conducting such thorough evaluation with experts is costly, and it is difficult to collect a sufficient amount of subjects for a significant statistical analysis. Hence, the results below should be seen mostly as a qualitative, rather than a quantitative study. Sample size for expert evaluation of visualizations is usually smaller than the sample




(a) Org1




(b) Org2




(c) Local+20

**Fig. 10.** Number of shortest attack plans (y axis) available after applying $k$ patches (x axis) to the most central nodes, according to the different centrality methods. Dashed lines represent centrality measure computed on the LAG, and solid lines represent measures computed over the planning graph.

size of non-expert evaluations for several reasons. First, in any domain, there is a limited number of available experts. This in itself limits the number of available experts for the evaluation. Elmqvist and Yi (2015). Second, it has been demonstrated repeatedly that considerably fewer experts are required to evaluate any type of interface relative to the required number of non-experts (e.g., Tory and Moller (2005)). Finally, expert evaluations usually focus on informal, qualitative aspects of the visualization, such as verifying if the visualization is useful for the domain, whether it presents the relevant content, and fits the experts' workflow (Kriglstein & Pohl, 2015). Such studies usually involve in-depth study of a few participants (Carpendale, 2008), as in our paper. Indeed, in a survey of 113 papers that included evaluations of visualizations, Isenberg et al. (2013) found the median number of evaluators to be 9 participants, with studies focusing on expert reviews usually employing only one or two experts.

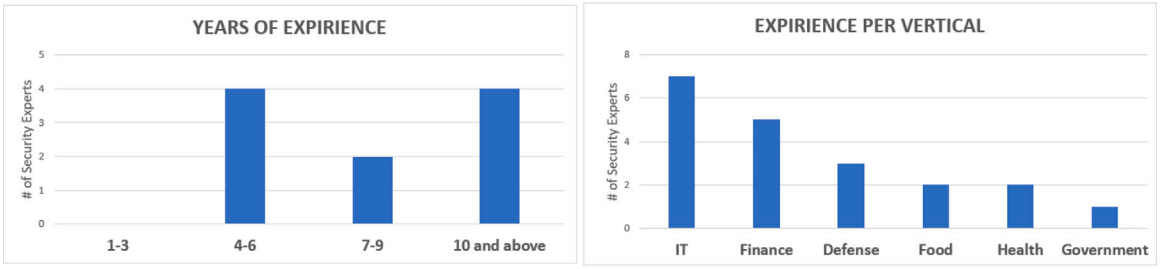
### 6.2.1. Method

We begin by describing the method we conducted in our expert study.

*Participants.* Ten network security experts from the United States and Israel participated in the evaluation. All are senior security experts that have substantial years of experience in vulnerability management in mid to large organizations (Fig. 11a). Fig. 11b shows the verticals of the organizations that the experts worked in throughout their careers. As can be seen, the experts have experience in diverse verticals, ranging from food companies to defense force organizations.

*Context and scenario.* For the prioritization tasks, we created an environment based on a fictitious corporate, which we called ACME Bank. We defined the ACME's network topology, its critical assets, network segmentation between the networks and finally a list of its vulnerabilities. We created an orientation presentation to allow the expert to learn the environment.

The network is based on our experience with real networks of organizations, and includes two locations, London and New York. Each location is divided into four main networks: a DMZ — the network that is exposed to external users or customers, Users — the network for internal employees, Management — the network that hosts the security




(a) Experts network security experience

(b) Experts vertical experience

**Fig. 11.** Experts background.

and network teams, and Server — the network that hosts the corporate internal servers. In addition, there are virtual instances hosted in an AWS service.

Each of the networks has about 100 devices in each of its locations, except for the user network that has 2000 devices in each location. There is segmentation enforcement in the network to ensure the users cannot access any of the other internal networks. The security and network team can access all networks from their management network. We defined 3 critical assets in the bank and assume the same criticality for all of them. The assets are the customer database in the AWS, the Web Servers in both DMZs (London and New York) and the Finance Server in New York's server network.

We created a list of up to date vulnerabilities that match the type of each device and its role. In the details pane, we present up to date information about the vulnerabilities. For example, a VPN in the DMZ is vulnerable to CVE-2018-0101, which is a remote code execution vulnerability on a Cisco ASA VPN Service. Another example is of the vulnerabilities in an IT admin machine we used, CVE-2018-8587, which is a vulnerability in outlook exploited by sending a malicious file.

*Alternative user interfaces.* We created three prototype user interfaces that present the vulnerabilities information to the expert. These three alternative interfaces, which we name Lisa, Mary, and Jane, are shown in Figs. 6, 7, and 8. Jane and Mary are the two novel interfaces, which use the NTVS score, and display network information, while Lisa is our representation of the commercially baseline user interfaces, designed to mimic popular commercial applications. All interfaces display information over the same network and vulnerabilities.

In all three interfaces, the vulnerabilities are listed on the left in a similar manner, sorted by decreasing severity. Jane and Mary both sort the vulnerabilities in the same manner, by the combination of NTVS and CVSS, and display both scores. Lisa, the baseline, ranks the vulnerabilities by CVSS score only, presenting to the user only CVSS information.

Following conventions in commercial applications, the right pane shows the details of the selected device vulnerability from the left list. The details pane shows standard information about the device and its vulnerability, such as its IP, OS, and the location of the device. The details pane also presents the CVSS score, and the vulnerability type description, as well as additional resource links about the specific vulnerability.

In addition to the information about the device and its vulnerability, both Jane and Mary present a visualization of the NTVS in the detail pane on the right. Jane displays a detailed topological visualization (Fig. 1b) of the network. This display represents the graph on the network topology map of the customer, allowing the experts to see all the subnets of the organization, their connections, and the position of the selected device within the network.

Mary, on the other hand, displays a logical representation (Fig. 1a) that focuses on the attack path. This high level view shows only the attack entry point, the assets that the attack targets, and the position of the selected device in the context of the attack.

#### 6.2.2. Procedure

We conducted a personal video conference interview with each of the experts. Before the interview, the experts were asked for permission to record the session to facilitate future analysis of their responses. They all agreed. The interview began with inquiries over the expert background. Then we presented to the expert the ACME bank scenario. After understanding the scenario, each expert worked with the three alternative user interfaces to prioritize vulnerability patches. The user interfaces were presented in a mixed order to mitigate potential order effects. Finally, we asked the expert a set of summarizing questions. The entire interview session lasted about 70 min on average. To preserve anonymity, personal details are not presented in Table 5 except for general, study-related information.

*Background interview.* The interview started with background questions regarding the expert's experience in network security in general, and specifically with vulnerability management. We also asked the expert to elaborate about the internal process they have used for handling vulnerabilities on the network. We focused on the process of prioritizing and remediating of vulnerabilities in the organizations they have worked for.

*Scenario presentation.* We presented the ACME network details and asked the experts to assume the role of Security Admin at the Bank. The experts then were asked to learn and understand the ACME network.

*Vulnerability prioritization using the alternative interfaces.* We presented the user interface prototypes one by one. The participants received the interfaces in different orders and in a balanced manner. For each interface prototype we requested the experts to prioritize the vulnerabilities according to their personal analysis.

Once the experts completed their analyses and presented their decision in each interface, we asked them to rate that alternative on 3 different aspects, using a scale of 1 to 5.

- How easy was it to prioritize the vulnerabilities, given the information presented by the application?
- How confident do you feel about your decision?
- How helpful would the application be in case you would need to persuade others in your decision?

*Expert feedback.* To conclude the study, we asked the experts to compare the alternative user interfaces. They ranked each alternative UI once based on its potential contribution and then based on their overall preference.

Finally, we asked some open questions on the value that the experts see in the various visualizations, and about the information needed to prioritize vulnerabilities effectively.

#### 6.2.3. Results

In their background interview (Table 5) experts provided an average rating of 3.3 to their current ease of prioritizing vulnerability's. All ratings are between 1–5, where lower ratings always denote worse,

**Table 5**
Background interview.

| Expert | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Experience (years) | 5.5 | 15 | 15 | 6 | 9 | 4 | 6 | 7 | 16 | 16 |
| Location | Israel | US | Israel | US | US | US | US | Israel | Israel | Israel |
| Organization's size | SML | MED | SML | LG | LG | MED | MED | LG | LG | LG |
| Experts experience with different verticals | | | | | | | | | | |
| IT | + | | + | | + | | + | + | + | + |
| Finance | | + | + | | | + | | | + | + |
| Government & defense forces | + | + | | | | | | | | + |
| Healthcare | | | | + | | | | + | | |
| Food | | | | + | | | + | | | |
| Experts experience with commercial Tools | | | | | | | | | | |
| Tenable | | + | + | | + | + | + | + | + | + |
| Rapid7 | + | | | + | | | + | | + | |
| Qualys | | | | | + | | | | | |
| Others | | | + | | | | | + | | + |
| Prioritization ease | 3 | 3 | 5 | 5 | 3 | 3 | 3 | 3 | 1 | 4 |
| certainty | 4 | 3 | 5 | 5 | 3 | 4 | 3 | 4 | 2.5 | 5 |

and 5 denotes the most easy, certain, explainable and most contributing solutions. Experts with experience in large organizations reported that their current prioritization task is less easy in general and they feel less certain about their decision then experts with experience in small organizations (2.8 vs. 4, and 3.6 vs. 4.5, respectively). This is aligned with our premise that a large amount of vulnerable devices, and higher network complexity, are key obstacles in the prioritization decision making.

Table 6 shows the grades and rankings given by the experts to the alternative applications for the different tasks. We now review these results in detail.

Fig. 12 shows the grades given by the experts to the alternative user interfaces immediately after their experience with the interface. As can be seen, in all aspects, both of our suggested interfaces have a higher score than the baseline Lisa interface ($p < 0.02$ using a two tailed paired t-test). That is, the experts found Jane and Mary to be easier to use then Lisa (4.8,4.55 vs. 2.95 respectively), felt more certain (4.5,4.4 vs. 3.1 respectively) in their decisions, and found the to be more explainable (4.1,3.95 vs. 2.5 respectively) than the Lisa baseline that mimics commercial applications.

Full results can be seen in Table 6. Interestingly, experts that viewed Lisa first, rated it much higher (4.17 on average for ease) than those who viewed Lisa second or third (2.43 on average for ease). When asked after the experiment, the experts said that Lisa is a fair representation of their current solutions. As such, it is reasonable to assume that users who viewed Lisa first felt comfortable with it due to their previous experience. Experts who viewed Lisa later implicitly compared Lisa to the richer interface that they viewed earlier. This farther strengthens our conclusions about the superiority of Jane and Mary over Lisa.

The differences between Jane and Mary are not significant.

After completing the required tasks on Jane, Mary, and Lisa, we asked the experts to grade the overall contribution each prototype may have to their current daily work. The experts provided an average grade of 4.4 for Jane, and 3.85 for Mary, while grading the effect of Lisa as 2.0 only ($p < 0.001$ using a t-test for the differences between Mary and Jane to Lisa, differences between Mary and Jane are not significant).

In the comparative questions following the experiment, experts said that the logical view in Mary can be understood much faster, and said its visualization is very easy to explain to non-technical peers and managers. They were concerned that the topology view of Jane would make it too complex to explain to managers. They also expressed concerns that the topological view will become exceedingly complex in large networks.

On the other hand, experts said that Jane is a slightly better tool to improve their confidence in their priority decision. They mentioned that the topological view gives much better context to the information,

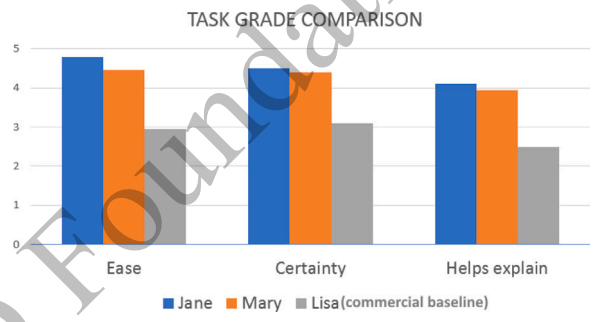



**Fig. 12.** Average grades given immediately after experiencing each alternative interface. Scale of 1 to 5 (5 is best).

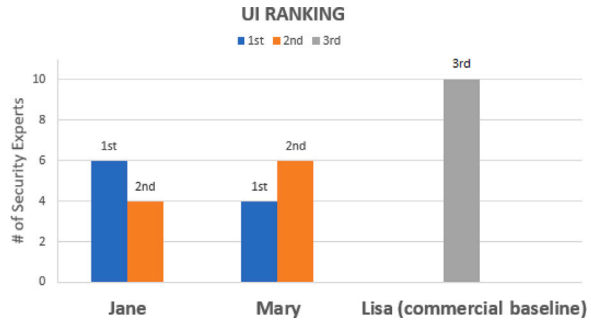



**Fig. 13.** Overall ranking of the interfaces by experts. Lower is better.

and this was the reason most of them ranked Jane as their first choice (7 for Jane vs. 3 for Mary and none for Lisa) (see Fig. 13).

Clearly, the topological and the logical representations have different strengths, and complement one another, with Jane being better for the security expert decision making, and Mary better for explaining the decisions to management. As such, a tool can offer both views, allowing users to switch between them.

For the most part, experts from different countries differed little in terms of their approach and ranking. The one aspect on which they did differ was the current complexity of the prioritization task. Israeli experts attested to slightly more complexity in prioritizing relative to US experts. We feel that the general uniformity is because most of the experts we interviewed worked at global companies which spread the security culture across their different locations.

Experts in large organizations preferred the topological view to help them achieve certainty on their decision. This could be because

**Table 6**
Detailed results of the expert study.

| Expert | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | Average | Stddev |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Order of presentation | Jane Mary Lisa | Jane Mary Lisa | Lisa Jane Mary | Mary Lisa Jane | Mary Lisa Jane | Lisa Jane Mary | Jane Lisa Mary | Mary Jane Lisa | Lisa Mary Jane | Mary Jane Lisa | | |
| Jane - easy | 5 | 5 | 5 | 4 | 5 | 5 | 4 | 5 | 5 | 5 | 4.80 | 0.44 |
| Jane- certain | 4 | 5 | 5 | 4.5 | 5 | 4 | 3 | 4.5 | 5 | 5 | 4.50 | 0.68 |
| Jane - explainable | 4 | 3 | 3 | 4 | 5 | 4 | 4 | 5 | 5 | 4 | 4.10 | 0.78 |
| Mary - easy | 5 | 5 | 5 | 5 | 5 | 5 | 4.5 | 4 | 5 | 2 | 4.55 | 0.96 |
| Mary- certain | 4 | 4 | 5 | 3.5 | 5 | 5 | 4 | 4 | 4.5 | 5 | 4.40 | 0.57 |
| Mary - explainable | 4 | 3 | 4 | 4 | 4 | 4 | 5 | 4.5 | 5 | 2 | 3.95 | 0.90 |
| Lisa - easy | 3 | 1 | 5 | 3 | 5 | 4 | 2 | 2 | 3.5 | 1 | 2.95 | 1.46 |
| Lisa- certain | 5 | 1 | 3 | 3 | 5 | 4 | 2 | 1 | 3 | 4 | 3.10 | 1.45 |
| Lisa - explainable | 3 | 2 | 3 | 3 | 3 | 4 | 2 | 3 | 1 | 1 | 2.50 | 0.97 |
| Jane - contribution | 5 | 5 | 4 | 3 | 5 | 4 | 3.5 | 4.5 | 5 | 5 | 4.40 | 0.74 |
| Mary - contribution | 4 | 3 | 5 | 4 | 4 | 3 | 4.5 | 4 | 4 | 3 | 3.85 | 0.67 |
| Lisa - contribution | 3 | 1 | 3 | 2 | 3 | 2 | 2 | 2 | 1 | 1 | 2.00 | 0.82 |
| Topo Vis - easy | 5 | 5 | 4 | 5 | 5 | 5 | 4 | 5 | 4.5 | 5 | 4.75 | 0.42 |
| Topo Vis - certain | 4 | 4 | 2 | 4 | 5 | 5 | 4 | 5 | 5 | 5 | 4.30 | 0.95 |
| Logical Vis - easy | 4 | 3 | 5 | 5 | 4 | 4 | 5 | 4 | 5 | 3 | 4.20 | 0.79 |
| Logical Vis - certain | 3 | 3 | 2 | 3.5 | 4 | 4 | 5 | 4 | 4 | 3 | 3.55 | 0.83 |
| Jane - rank | 1st | 1st | 2nd | 2nd | 1st | 1st | 2nd | 1st | 1st | 1st | | |
| Mary Rank | 2nd | 2nd | 1st | 1st | 2nd | 2nd | 1st | 2nd | 2nd | 2nd | | |
| Lisa - Rank | 3rd | 3rd | 3rd | 3rd | 3rd | 3rd | 3rd | 3rd | 3rd | 3rd | | |

in a smaller network an expert may be familiar with the relative positions of devices within the network. In larger networks, however, an expert cannot recall the locations of devices according to their labels (presented in the logical view), and the topological view provides this missing information.

In further post-study open discussions we learned that almost all of the organizations our experts have worked at (9 out of 10 of current organizations they work for) deploy a prioritization process that relies mostly on the CVSS score with some differences between different areas of their network. For example a production network that holds sensitive data or supports customers will get a priority over the rest of the networks. One organization has a per case analysis process that adds an analysis of the specific impact of the service that is vulnerable.

All of the mid to large size organizations have a planned periodic schedule for performing their vulnerability remediations. Some experts reported that if the internal process required to remediate a vulnerability and it could not be done in time for the periodic schedule, they are required to perform a thorough analysis and mitigation plan on why this is not done. In most cases the analysis is both about the risk from the vulnerability and the different mitigation plans, why they cannot be performed on time, and when a solution can be implemented. We believe this further supports the value of our suggestion. Experts mentioned that our suggested interfaces provide clear support of the risk analysis expected from them.

An interesting observation from the expert study is that experts from different countries and different verticals expressed similar approach to issues of prioritization of network vulnerabilities and provided very similar rankings of the suggested solutions relative to standard solutions. This could indicate that, at least in this aspect of network security, processes across global companies have become similar. If that is the case, our proposed solution may prove helpful in improving standards across the field and not just in isolated cases.

In our interviews, many of the experts indicated that the prioritization of remediation requires balancing of the risk and the effort of safely patching the vulnerability without damaging critical network operations. They mentioned that adding a metric that would indicate the ease of implementing the required remediation in their environment would help them improve their prioritization process. We believe this could be added to our model. We can collect public data that will allow an initial value for this metric, such as whether there already exists an available published patch.

## 7. Conclusion and future work

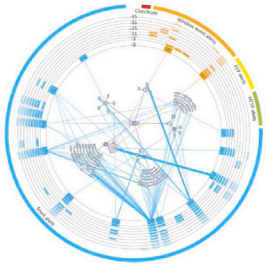
*Summary of the results.* In this paper we investigate techniques to support security experts while deciding which vulnerabilities to patch first. Assuming that it is not feasible to eliminate all attack paths toward the critical assets we strive to (1) identify vulnerability patches that eliminate as many shortest attack paths as possible, (2) efficiently communicate this information to the security experts focusing on mid to large network environments, and (3) provide the experts with sufficient information to both motivate patching strategy to their management and communicate with the operational teams.

We experiment with real world attack graphs, obtained by scanning the computer networks of two organizations, and with a standard simulated LAG benchmark. It is interesting to see that the results over the simulated network are very different from the results over the real network, emphasizing the need for additional real world networks for experiments. Our results show that the planning graph, a data structure from automated planning, facilitates better vulnerability ranking using centrality measures than the traditional LAG or connectivity graphs.

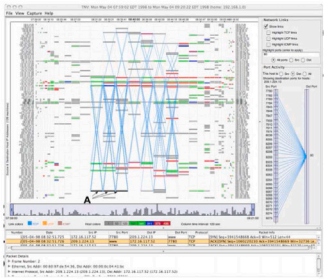
We suggest two possible methods for displaying the network and attack information to experts — a topological view showing the entire network structure, and a logical view focusing on the role of the vulnerable device in possible attacks. A study conducted with 10 security experts shows that all experts preferred our suggested interfaces over a baseline mimicking current commercial solutions that they are familiar with.

*Limitations.* The main practical limitation of our approach is that it is designed to evaluate multi-hop attacks, where the attacker moves from one machine to another, until it reaches a particular desired asset. Our method cannot currently handle other types of attacks, such as denial of service (DOS) (Arora et al., 2021). Extending our methods to such attacks may require developing different types of graphical representations.

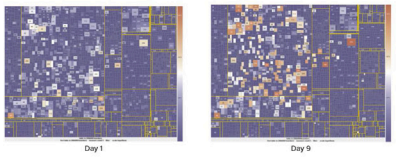
The reported expert study was not designed to disentangle the effects of the NTVS metric and the related visualization on the satisfaction of the experts. We see this distinction as secondary to the main thrust of the paper, since both elements are important and complementary. The feedback received from the experts confirms this conjecture , although in a real application, experts may choose to deploy only one of the proposed elements.



(a) Vizalert(Livnat et al., 2005): Visualization paradigm for network intrusion detection. Showing different types of attacks (color coded) targeting hosts on the network map. The events are shown over time, using the multiple rings, were the internal rings are the most recent and the outer rings show previous intervals.



(b) TNV(Goodall et al., 2005): Displaying IP to IP traffic in a 90 minute time period. The view focuses (using most of the view) on a 15 minute interval using its fisheye design. The rest of the 60 minute duration are displayed in narrow columns before and after the focused area.




(c) Treemap visualization (Mansmann et al., 2007) displaying the spread of botnet computers from the perspective of a large service provider.The left image shows the state of malware in the beginning of the attack and the right image shows the state after 9 days.

(d) CyVi prototype (Muhati et al., 2020) displaying malicious connections (in read) vs legitimate connections (in green) on geographically distributed network.

**Fig. A.14.** Network security visualizations providing a wide context of an attack. Clearly, all Visualizations present a considerable amount of data, which may cause information overload.

*Future work.* There are additional challenges that need to be addressed in followup studies. First, we intend disentangle the effects of the NTVS and visualization on the usability of vulnerability ranking in a future study, including the possibility of improving our current visualizations (Jane and Mary) and allowing security experts to toggle between different modes of network visualization.

Second, in our suggested visualization, we show the possible topological paths from the vulnerable devices to the critical systems. In cases of mobile devices including laptops, phones, tablets, and mobile IoTs, further research is required to help experts prioritize maintenance of mobile devices efficiently. While additional research is needed here on attack graph generation, the visualization that we suggest should also be adjusted to display mobile devices.

Finally, displaying all possible attack paths to critical systems in a large environment is a difficult challenge. We surveyed related work that tackled this problem, yet we did not find a clear usability testing showing a beneficial solution in large scale networks. An hierarchical view may be needed to make our interfaces useful in very large networks.

In a broader sense from a different angle, the proposed NTVS and visualization techniques can be used to prioritize maintenance in complex industrial control systems for the purpose of failure prevention and security alike. Consider faults in a production floor machinery detected, for example, through the analysis of continuous mechanical vibration (Hu et al., 2018). Similar to vulnerabilities detected using Nessus, these artifacts may be arranged in a complex hybrid bond graph structure (Xiao et al., 2020) and represented using a set of logical rules similar to LAG. In such representation a production plan is reminiscent to an attack plan and NTVS may be used to select failures that intervene with many production plans. The proposed visualization can be used by the operators to select such faulty systems to be replaced or updated.

## CRediT authorship contribution statement

**Amir Olswang:** Conceptualization, Software, Investigation, Writing – original draft. **Tom Gonda:** Conceptualization, Software, Formal analysis, Data curation, Investigation, Writing – original draft. **Rami Puzis:** Conceptualization, Supervision, Writing – review & editing. **Guy Shani:** Conceptualization, Supervision, Writing – review & editing, Project administration. **Bracha Shapira:** Conceptualization, Supervision, Writing – review & editing. **Noam Tractinsky:** Conceptualization, Supervision, Writing – review & editing.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

This work was partially supported by the CBG Cyber Security Center at Ben Gurion University of the Negev, Israel, and by ISF, Israel grant 1210/18.

## Funding

*ISF – Israel Science Foundation*
*The BGU Cyber Security Research Center*

*Ethics statement*

IRB approval was obtained (required for studies and series of 3 or more cases)

## Appendix. Additional types of network security visualization

Extensive research was dedicated to using visualization to help security experts get a quick understanding of an attack without losing the wider context of the network. This is done to support a rapid analysis that will allow focusing on the most critical risk and take the appropriate actions (Fig. A.14). An interesting example of helping security admins focus while analyzing network activity is a visualization created by Goodall et al. (2005). The researchers used a fisheye pattern in their TNV tool shown in Fig. A.14b to allow users to focus on IP to IP traffic in a specific time frame without losing the context of what happened before and after (the previous and following time frames appear as narrow columns on the right and left of the center view).

Another example that uses a high level abstraction is the work done by Mansmann et al. (2007) in which they utilize a tree map visualization to show malware propagation in a huge network see Fig. A.14c

An interesting use of the network topological map to help users get better and faster orientation was done by Livnat et al. (2005) who created the VisAlert tool. This tool seen in Fig. A.14a visualizes attacks on the network by severity type and target. In the center of their view they have a topological map surrounded by multiple rings that represent the attack type and their severity. Arrows are displayed from a specific attack on the ring to the attacked target in the network allowing easy context of the risk from the attack.

A recent work by Muhati et al. (2020) visualized malicious connections (attacks) on geographic distributed network seen in Fig. A.14d. The view allows the expert to see the locations of attacked assets in an intuitive method. In addition their suggested tool allows to visualize the effects of defense strategies that block malicious connections.

Perhaps the most obvious observation from the user interfaces in Fig. A.14 is the information overload — the amount of displayed information is considerable, requiring experts to invest much cognitive effort to understand the information and draw conclusions.

## References


Albanese, M., Jajodia, S., & Noel, S. (2012). Time-efficient and cost-effective network hardening using attack graphs. In *Dependable systems and networks (DSN), 2012 42nd annual IEEE/IFIP international conference on* (pp. 1–12). IEEE.

Arora, A., Yadav, S. K., & Sharma, K. (2021). Denial-of-service (dos) attack and botnet: Network analysis, research tactics, and mitigation. In *Research anthology on combating denial-of-service attacks* (pp. 49–73). IGI Global.

Barrère, M., & Lupu, E. C. (2017). Naggen: A network attack graph generation tool—IEEE CNS 17 poster. In *2017 IEEE conference on communications and network security (CNS)* (pp. 378–379). IEEE.

Beale, J., Deraison, R., Meer, H., Temmingh, R., & Walt, C. V. D. (2004). *Nessus network auditing*. Syngress Publishing.

Bhattacharya, S., & Ghosh, S. (2008). An attack graph based risk management approach of an enterprise lan. *Journal of Information Assurance and Security*, *3*.

Blum, A. L., & Furst, M. L. (1997). Fast planning through planning graph analysis. *Artificial Intelligence*, *90*(1), 281–300.

Brandes, U. (2008). On variants of shortest-path betweenness centrality and their generic computation. *Social Networks*, *30*(2), 136–145.

Bryce, D., & Kambhampati, S. (2007). A tutorial on planning graph based reachability heuristics. *AI Magazine*, *28*(1), 47–83.

Carpendale, S. (2008). Evaluating information visualizations. In *Information visualization* (pp. 19–45). Springer.

Cohen, G., Meiseles, M., & Reshef, E. (2012). System and method for risk detection and analysis in a computer network. US Patent 8, 099, 760.

De Mello, L. H., & Sanderson, A. C. (1990). AND/OR graph representation of assembly plans. *IEEE Transactions on Robotics and Automation*, *6*(2), 188–199.

Dempsey, K., Chawla, N. S., Johnson, A., Johnston, R., Jones, A. C., Orebaugh, A., Scholl, M., & Stine, K. (2012). *Information security continuous monitoring (ISCM) for federal information systems and organizations: National institute of standards and technology special publication 800-137*. CreateSpace Independent Publishing Platform.

Durkota, K., Lisỳ, V., Bošanskỳ, B., & Kiekintveld, C. (2015). Optimal network security hardening using attack graph games. In *Proceedings of IJCAI* (pp. 7–14).

Elmqvist, N., & Yi, J. S. (2015). Patterns for visualization evaluation. *Information Visualization*, *14*(3), 250–269.

Forum, W. E. (2018). *The global risks report 2018 13th edition*. World Economic Forum.

Freeman, L. C. (1977). A set of measures of centrality based on betweenness. *Sociometry*, 35–41.

Frigault, M., & Wang, L. (2008). Measuring network security using bayesian network-based attack graphs. In *2008 32nd annual IEEE international computer software and applications conference* (pp. 698–703). IEEE.

Gartner (2019). Gartner customer for vulnerability assessment solutions. https://www.gartner.com/reviews/market/vulnerability-assessment. Accessed: 2019-09-28.

Gefen, A., & Brafman, R. I. (2012). Pruning methods for optimal delete-free planning. In *ICAPS*.

Gonda, T., Pascal, T., Puzis, R., Shani, G., & Shapira, B. (2018). Analysis of attack graph representations for ranking vulnerability fixes. In *GCAI* (pp. 215–228).

Gonda, T., Shani, G., Puzis, R., & Shapira, B. (2017). Ranking vulnerability fixes using planning graph analysis. In *IWAISe: First International Workshop on Artificial Intelligence in Security*.

Goodall, J., Lutters, W., Rheingans, P., & Komlodi, A. (2005). Preserving the big picture: Visual network traffic analysis with tnv. In *Proc. IEEE workshop visualization for computer security (VizSEC 05)* (pp. 47–54).

Hoffmann, J. (2003). The metric-FF planning system: Translating"ignoring delete lists"to numeric state variables. *Journal of Artificial Intelligence Research*, *20*, 291–341.

Hoffmann, J. (2015). Simulated penetration testing: From" Dijkstra" to" Turing Test++". In *ICAPS* (pp. 364–372).

Homer, J., Varikuti, A., Ou, X., & McQueen, M. A. (2008). Improving attack graph visualization through data reduction and attack grouping. In *International workshop on visualization for computer security* (pp. 68–79). Springer.

Hong, J. B., & Kim, D. S. (2013). Scalable security analysis in hierarchical attack representation model using centrality measures. In *Dependable systems and networks workshop (DSN-W), 2013 43rd annual IEEE/IFIP conference on* (pp. 1–8). IEEE.

Hong, J. B., Kim, D. S., & Haqiq, A. (2014). What vulnerability do we need to patch first? In *Dependable systems and networks (DSN), 2014 44th annual IEEE/IFIP international conference on* (pp. 684–689). IEEE.

Hu, Q., Qin, A., Zhang, Q., He, J., & Sun, G. (2018). Fault diagnosis based on weighted extreme learning machine with wavelet packet decomposition and KPCA. *IEEE Sensors Journal*, *18*(20), 8472–8483.

Huang, H., Zhang, S., Ou, X., Prakash, A., & Sakallah, K. (2011). Distilling critical attack graph surface iteratively through minimum-cost sat solving. In *Proceedings of the 27th annual computer security applications conference* (pp. 31–40). ACM.

Idika, N., & Bhargava, B. (2010). Extending attack graph-based security metrics and aggregating their application. *IEEE Transactions on Dependable and Secure Computing*, *9*(1), 75–85.

Isenberg, T., Isenberg, P., Chen, J., Sedlmair, M., & Möller, T. (2013). A systematic review on the practice of evaluating visualization. *IEEE Transactions on Visualization and Computer Graphics*, *19*(12), 2818–2827.

Jajodia, S., & Noel, S. (2010). Topological vulnerability analysis. In *Cyber situational awareness* (pp. 139–154). Springer.

Jajodia, S., Noel, S., Kalapa, P., Albanese, M., & Williams, J. (2011). Cauldron mission-centric cyber situational awareness with defense in depth. In *2011-MILCOM 2011 military communications conference* (pp. 1339–1344). IEEE.

Kriglstein, S., & Pohl, M. (2015). Choosing the right sample? Experiences of selecting participants for visualization evaluation.

Livnat, Y., Agutter, J., Moon, S., Erbacher, R., & Foresti, S. (2005). A visualization paradigm for network intrusion detection. In *Proc. sixth ann. IEEE SMC information assurance workshop (IAW 05)* (pp. 92–99).

Mansmann, F., Keim, D., North, S., Rexroad, B., & Sheleheda, D. (2007). Visual analysis of network traffic for resource planning, interactive monitoring, and interpretation of security threats. *IEEE Transactions on Visualization and Computer Graphics*, *13*(6), 1105–1112.

Mell, P., Scarfone, K., & Romanosky, S. (2006). Common vulnerability scoring system. *IEEE Security & Privacy*, *4*(6), 85–89.

Muhati, E., Rawat, D. B., Garuba, M., & Njilla, L. (2020). CyVi: Visualization of cyber-attack and defense effects in geographically referenced networks. In *2020 IEEE 17th annual consumer communications & networking conference (CCNC)* (pp. 1–4). IEEE.

NIST (2019). *National Institute of Standards and Technology (NIST): Vulnerabilities*. National Institute of Standards and Technology (NIST).

Noel, S., & Jajodia, S. (2014). Metrics suite for network attack graph analytics. In *Proceedings of the 9th annual cyber and information security research conference* (pp. 5–8). ACM.

Obes, J. L., Sarraute, C., & Richarte, G. (2013). Attack planning in the real world. arXiv preprint arXiv:1306.4044.

Ou, X., Boyer, W. F., & McQueen, M. A. (2006). A scalable approach to attack graph generation. In *Proceedings of the 13th ACM conference on computer and communications security* (pp. 336–345). ACM.

Ou, X., Govindavajhala, S., & Appel, A. W. (2005). MulVAL: A logic-based network security analyzer. In *USENIX security symposium, Vol. 8* (pp. 113–128). Baltimore, MD.

Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). *The PageRank citation ranking: Bringing order to the web*: *Technical Report*, Stanford InfoLab.

Sawilla, R. E., & Ou, X. (2008). Identifying critical attack assets in dependency attack graphs. In *European symposium on research in computer security* (pp. 18–34). Springer.

Shmaryahu, D. (2016). Constructing plan trees for simulated penetration testing. In *The 26th international conference on automated planning and scheduling* (p. 121).

Shostack, A. (2003). Quantifying patch management. *Secure Business Quarterly*, *3*(2), 1–4.

Shu, X., Tian, K., Ciambrone, A., & Yao, D. (2017). Breaking the target: An analysis of target data breach and lessons learned. arXiv preprint arXiv:1701.04940.

Sommestad, T., & Sandström, F. (2015). An empirical test of the accuracy of an attack graph analysis tool. *Information & Computer Security*, *23*(5), 516–531.

Swiler, L. P., Phillips, C., Ellis, D., & Chakerian, S. (2001). Computer-attack graph generation tool. In *DARPA information survivability conference & exposition II, 2001. DISCEX'01. Proceedings, Vol. 2* (pp. 307–321). IEEE.

Tenable (2020). Tenable.sc vulnerability scanner. https://www.tenable.com/products/tenable-sc. Accessed: 2020-07-30.

Tory, M., & Moller, T. (2005). Evaluating visualizations: do expert reviews work? *IEEE Computer Graphics and Applications*, *25*(5), 8–11.

Wang, L., Islam, T., Long, T., Singhal, A., & Jajodia, S. (2008). An attack graph-based probabilistic security metric. In *Data and applications security XXII* (pp. 283–296). Springer.

Williams, L., Lippmann, R., & Ingols, K. (2008). GARNET: A graphical attack graph and reachability network evaluation tool. In *International workshop on visualization for computer security* (pp. 44–59). Springer.

Xiao, C., Yu, M., Zhang, B., Wang, H., & Jiang, C. (2020). Discrete component prognosis for hybrid systems under intermittent faults. *IEEE Transactions on Automation Science and Engineering*.