An Attack Volume Metric

Dr. Massimiliano Albanese

George Mason University

I. Iganibo, M. Albanese, M. Mosko, E. Bier, and A.E. Brito. **An Attack Volume Metric**. *Security and Privacy*, vol 6, no. 4, July 2023, DOI: https://doi.org/10.1002/spy2.298

GEORGE MASON UNIVERSITY

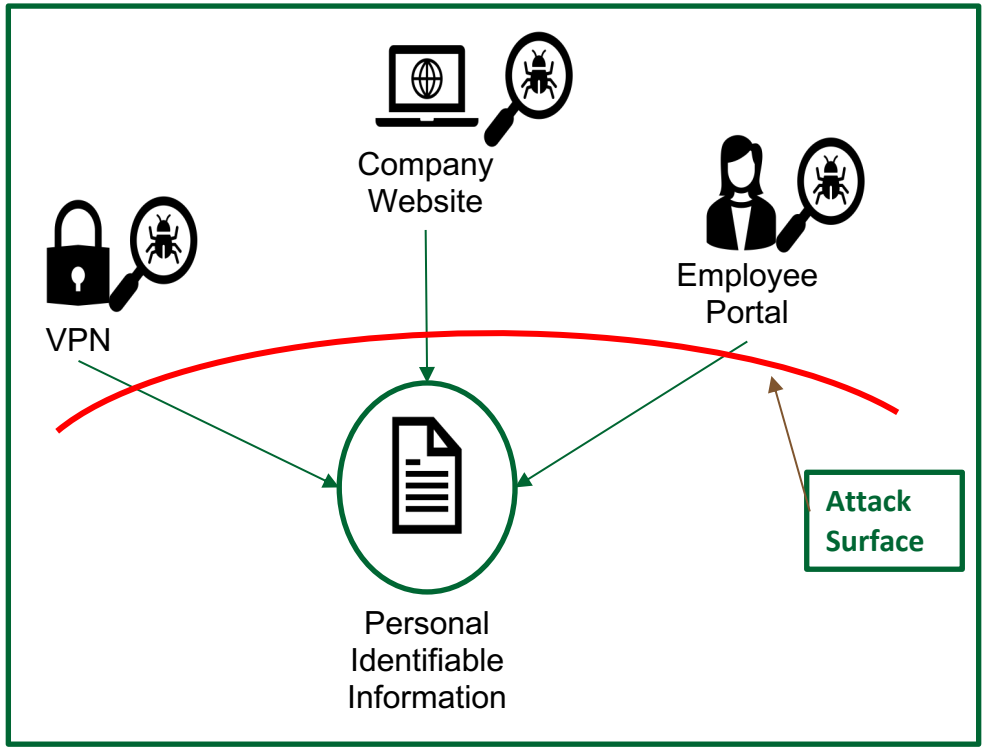
Outline

- Background and Motivation
- Vulnerability Metrics
- Beyond a System's Attack Surface: Attack Volume Metrics
- Experimental Evaluation
- Conclusions

MASON

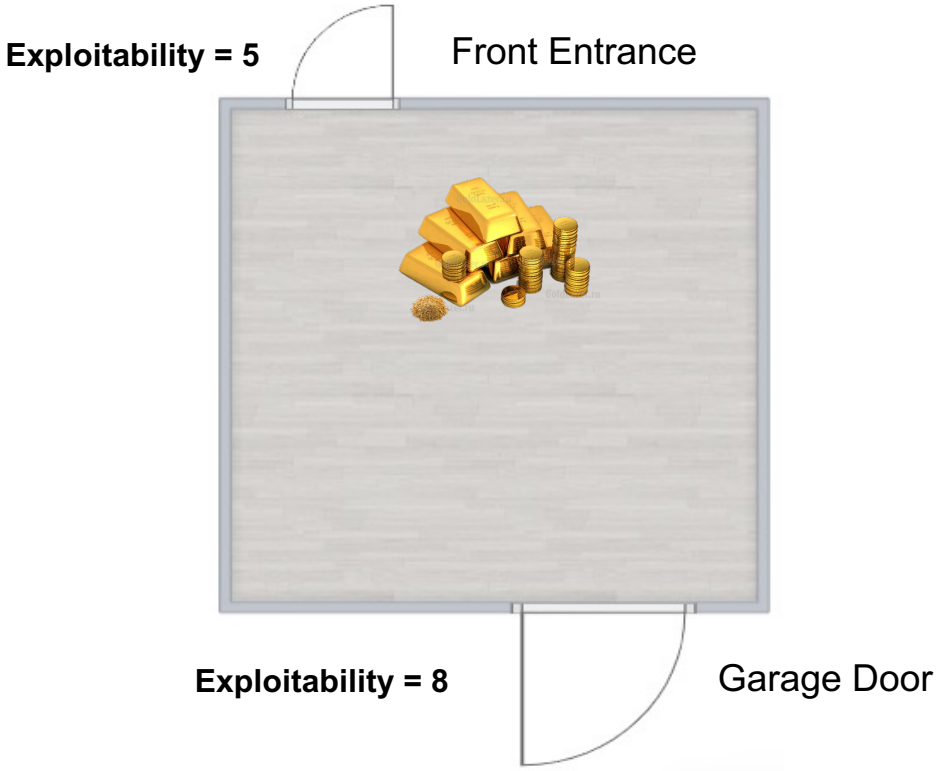
Attack Surface Model: Definition

- *"The **set of points** on the **boundary of a system**, a system element, or an environment **where an attacker can try to enter**, cause an effect on or extract data from that system."* (Source: NIST SP 800-160 Vol.2)
  - Knowledge of the Attack Surface allow us to identify entry points that enable cyber attacks
- Limitations of this model
  - Doesn't **measure** the effect of an exploit beyond the attack surface
  - Doesn't consider the **cascading effects** of an exploit



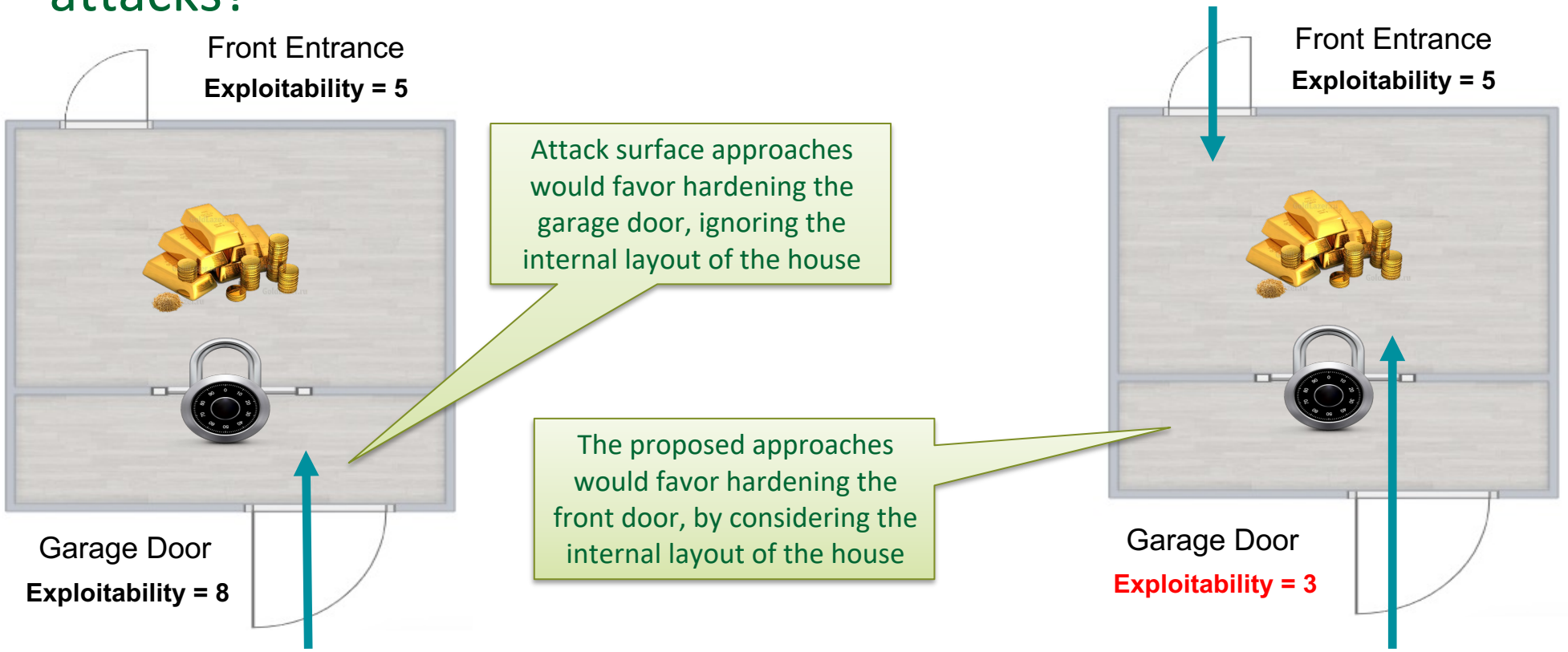
Not all Entry Points are Created Equal

The gold thief analogy: Which entry point will the thief exploit?



Not all Entry Points are Created Equal

- Can attack surface models accurately assess the impact of different attacks?

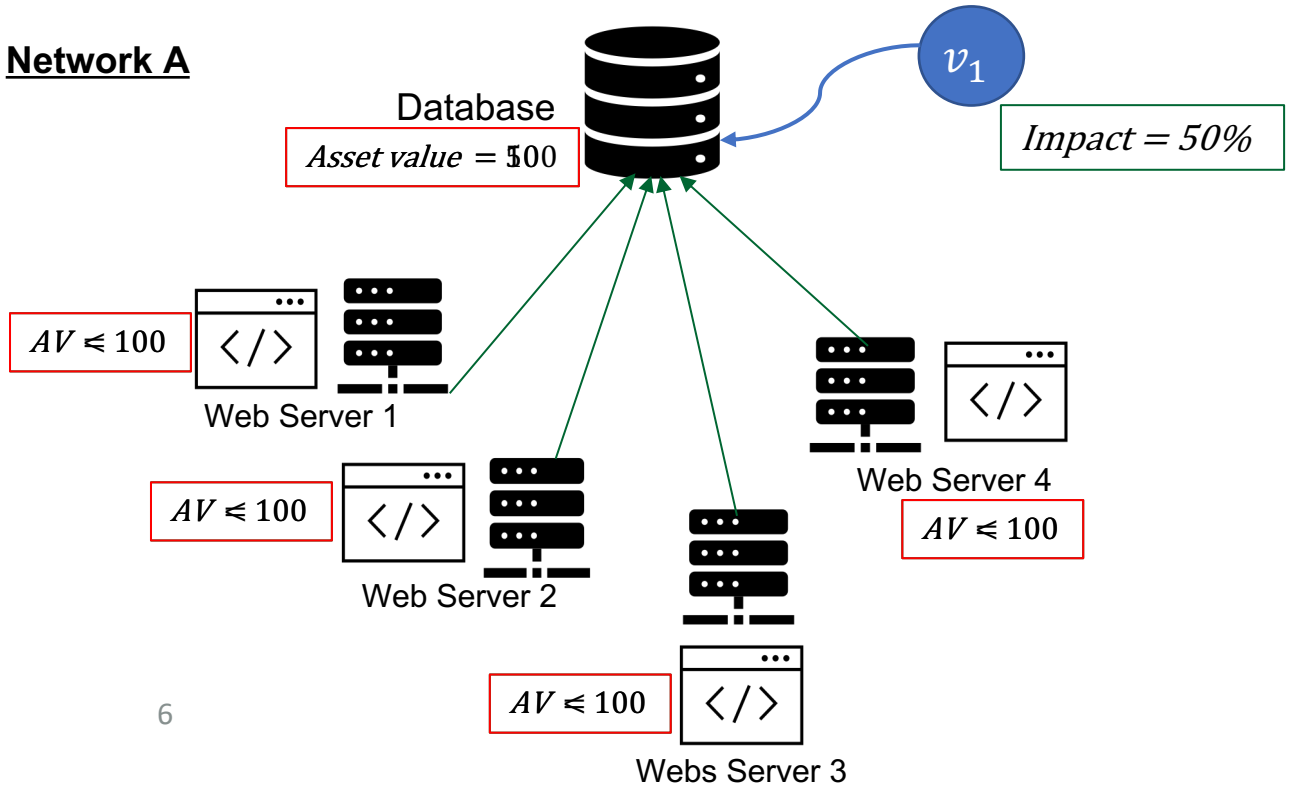


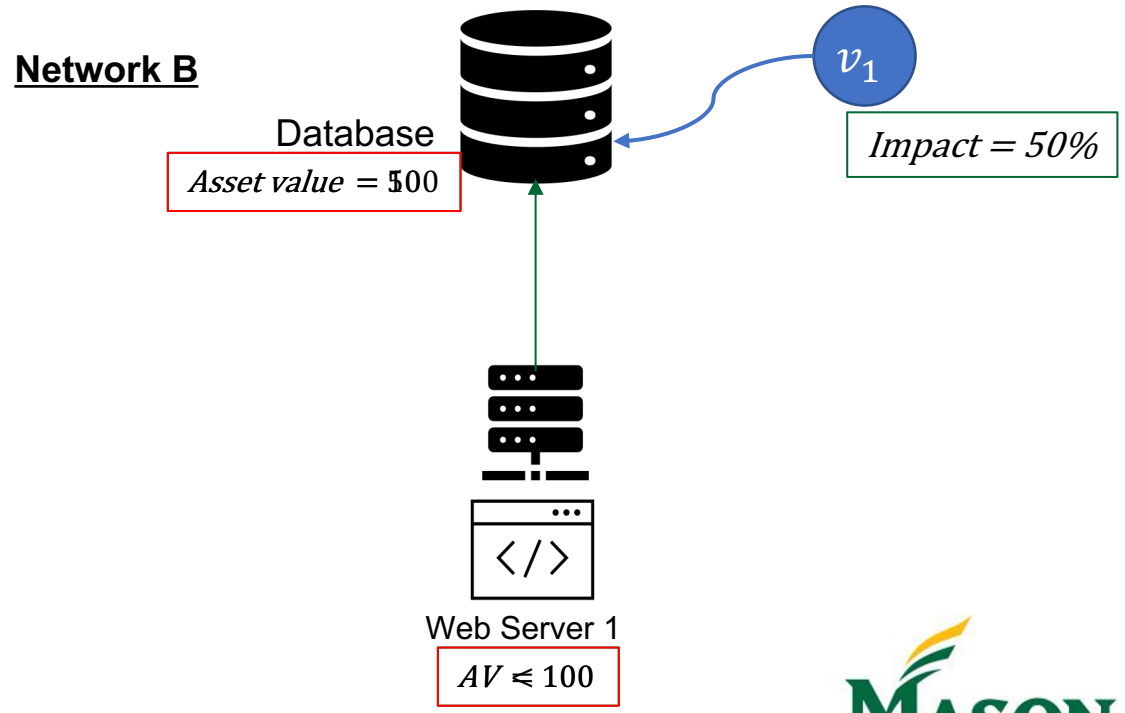
Limitation of the State of the Art: Example

Compare two networks of a cloud service provider:

- Are these attack surfaces equivalent?
- Can current metrics correctly assess the impact?

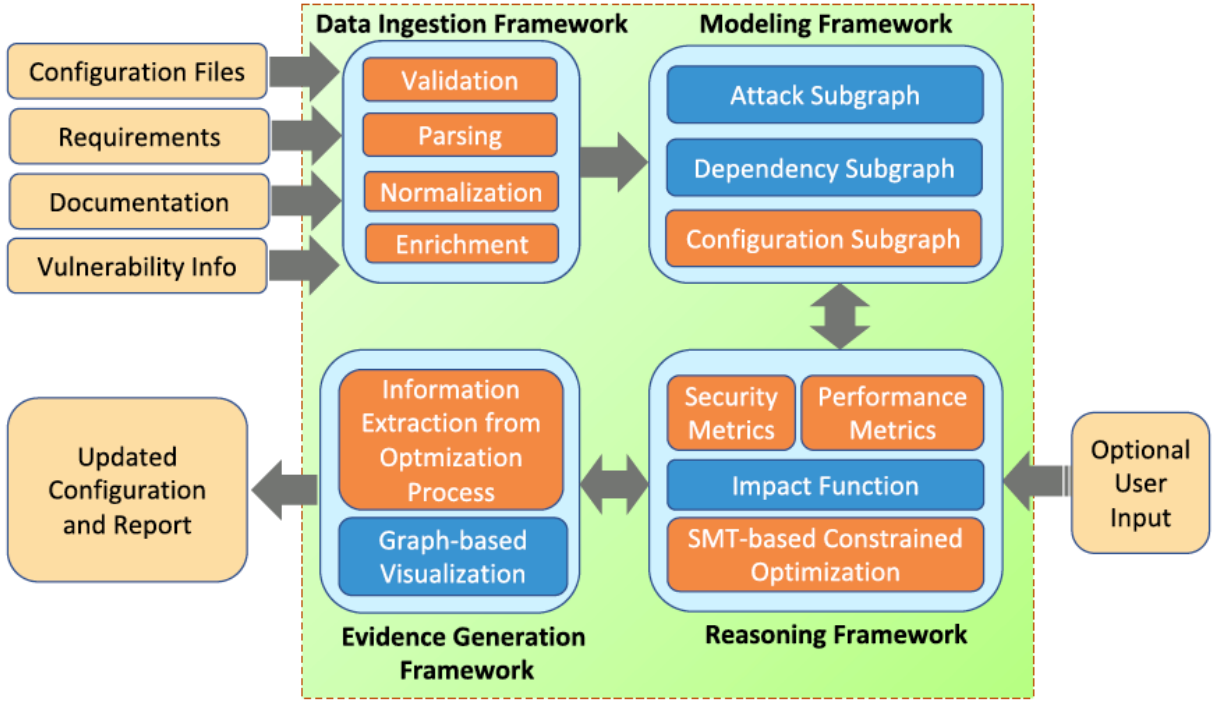
The same attack has a higher impact on this network





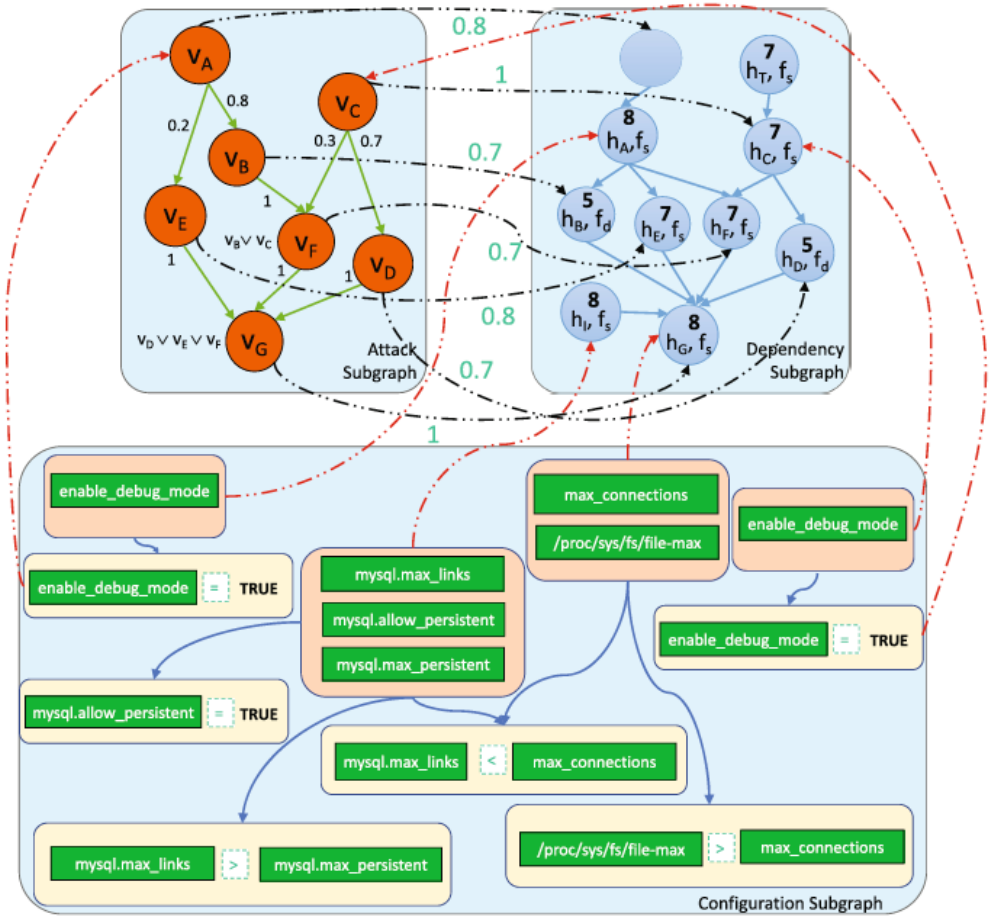
The SCIBORG Project

- The "Secure Configurations for the IoT Based on Optimization and Reasoning on Graphs" (**SCIBORG**) is a system to model, analyze, and optimize the configuration of complex systems
  1. **Ingests** system requirements, configuration files, software documentation, etc.
  2. Builds a queryable, **graph-based representation** of the relationships between vulnerabilities, configuration parameters, and system components
  3. Provides an API to perform a **quantitative analysis** of the security impact of config settings
  4. Automatically formulates a **constraint satisfaction problem** and uses a solver to find optimal parameter values
  5. Provides **human-readable evidence** for the optimality of the selected configuration



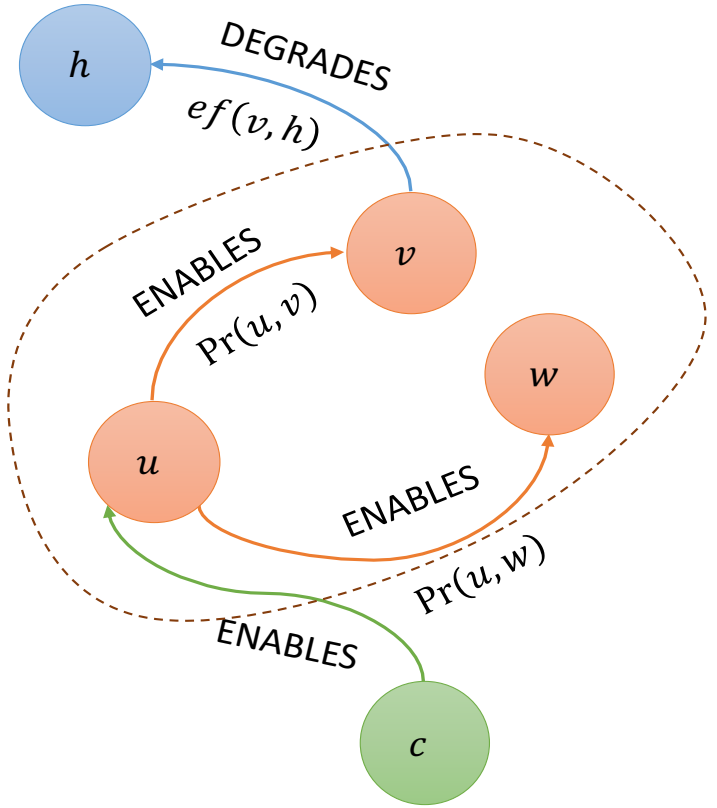
The SCIBORG Model

- **Attack Subgraph**
  - Models dependencies between vulnerabilities
- **Dependency Subgraph**
  - Models functional dependencies between system components
- **Configuration Subgraph**
  - Models relationships between configuration parameters and configuration constraints
- **Edges across Subgraphs**
  - Configuration Subgraph → Dependency Subgraph
  - Configuration Subgraph → Vulnerability Subgraph
  - Vulnerability Subgraph → Dependency Subgraph



Metrics

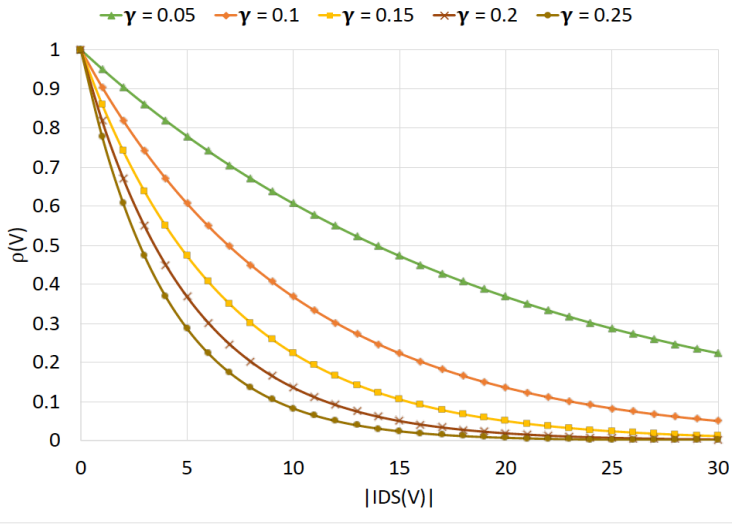
- Metrics are needed to evaluate the nodes and edges in the multi-graph
- **Exploitation Likelihood $\boldsymbol{\rho(v)}$**
  - Represents the conditional probability that a vulnerability v will be exploited, if all preconditions are met
- **Edge Probability $\boldsymbol{Pr(\mathrm{u}, v)}$**
  - Represents the relative probability that a vulnerability v will be exploited after exploiting u
- **Exposure Factor $\boldsymbol{\mathrm{ef}(\mathrm{v}, h)}$**
  - Represents the relative impact, on a scale from 0 to 1, to a component h due to the exploitation of vulnerability v



Exploitation Likelihood Metric

- **Exploitation Likelihood** labels **nodes in the vulnerability subgraph**
- The exploitation likelihood of a vulnerability v is defined as
  - $p(v) = \frac{\left(1-e^{-\alpha.\sqrt{t(v)}}\right)\cdot\left(1-e^{-\beta.Exploitability(v)}\right)}{e^{\gamma\cdot|IDS_K(V)|}}$

  where:
  - $\boldsymbol{t(v)}$ is the time since vulnerability v was discovered
    - More exploits and skills may be available for older vulnerabilities
  - $\boldsymbol{Exploitability(v)}$ is the CVSS $\boldsymbol{Explotaibility}$ score of v
    - Easily exploitable vulnerabilities are more likely to be exploited by the attacker
  - $\boldsymbol{IDS_K(V)}$ is the number of known IDS rules associated with v
    - Attackers may not choose vulnerabilities with higher chances of detection (i.e., those with more IDS rules)
  - $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and $\boldsymbol{\gamma}$ are tunable parameters to control the influence of the variables



Effect of $|IDS_k(v)|$ on likelihood

M. Albanese, I. Iganibo, O. Adebiyi. **A framework for designing vulnerability metrics**. Computers & Security, Volume 132, Elsevier, 2023. DOI: https://doi.org/10.1016/j.cose.2023.103382

Edge Probability

- At every step of an attack, adversaries can choose one of several vulnerabilities to exploit next to advance the attack
- All the variables that can influence the attacker's choice of vulnerabilities to exploit have been factored into each vulnerability's likelihood
  - Thus, the edge probability distribution can be computed by normalizing the likelihood of the enabled vulnerabilities
- Given an ENABLES edge (u, v) the probability of exploiting v after u is

$$\Pr(u, v) = \frac{\rho(v)}{\sum_{v^* \; s.t. \; (u,v^*) \in E} \rho(v^*)}$$

MASON

Exposure Factor Metric

- **Exposure Factor** labels the **edges from the nodes in the vulnerability graph to nodes in the dependency graph**
- The exposure factor of a component h to a vulnerability v quantifies the relative damage that exploitation of v would cause to h
- For a given DEGRADES edge (v, h), the $\boldsymbol{exposure\ factor}$ is defined as:
  - $ef(v,h) = \frac{1-e^{-\lambda \cdot impact\,(v)}}{e^{\delta \cdot |IDS_d(v)|}}$

  where:
  - $\mathbf{impact}\ (\boldsymbol{v})$ is the CVSS $\boldsymbol{Impact}$ score of v
  - $\boldsymbol{IDS_d(v)}$ is the set of deployed IDS rules associated with v
  - $\boldsymbol{\lambda}$ and $\boldsymbol{\delta}$ are tunable parameters

Known vs. Deployed IDS Rules

- The existence of IDS rules is one of the factors influencing the computation of the proposed metrics
  - In our implementation, we leverage rules from **Snort** and **Suricata**
- We only consider rules that are explicitly mapped to CVE entries and distinguish between **known** and **deployed** rules
  - **Known IDS rule**. Any IDS rule that is available to the community through publicly accessible repositories
  - **Deployed IDS rule**. Any IDS rule that is being actively used by a deployed IDS

Known IDS rules	Deployed IDS rules
Found in public repositories	Found in installed IDS
Known to the attacker	Not known to the attacker
Do not include custom rules	Include custom rules
Influence the *Exploitation Likelihood*	Influence the *Exposure Factor*

MASON

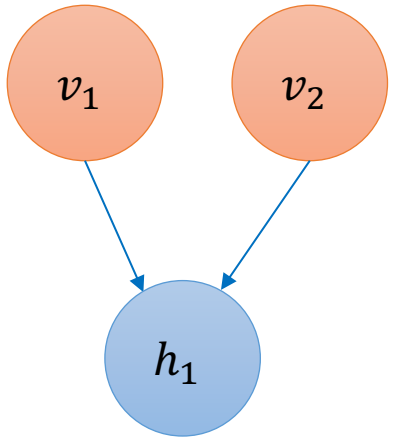
Beyond a System's Attack Surface

- Modeled **5 generations** of metrics
  - **Refined** existing attack surface metrics until all possible scenarios and properties of the system were considered
- Exposed vulnerabilities:
  - V^e: the set of vulnerabilities exposed on public-facing components
- $\boldsymbol{as_1} = |V^e|$
- $\boldsymbol{as_2} = \sum_{v \in V^e} \rho(v)$
- $\boldsymbol{as_3} = \sum_{v \in V^e} \sum_{h \in \{h \in H \,|degrades(v,h)\}} \rho(v) \cdot ef(v,h) \cdot u(h)$
- $\boldsymbol{as_4} = \sum_{v \in V} \sum_{h \in \{h \in H \,|degrades(v,h)\}} \rho^*(v) \cdot ef(v,h) \cdot u(h)$
- $\boldsymbol{avm} = \sum_{v \in V} \sum_{h \in \{h \in H \,|degrades(v,h)\}} \rho^*(v) \cdot ef(v,h) \cdot \left(u(h) + \sum_{h^* \in D(h)} u(h^*)\right)$

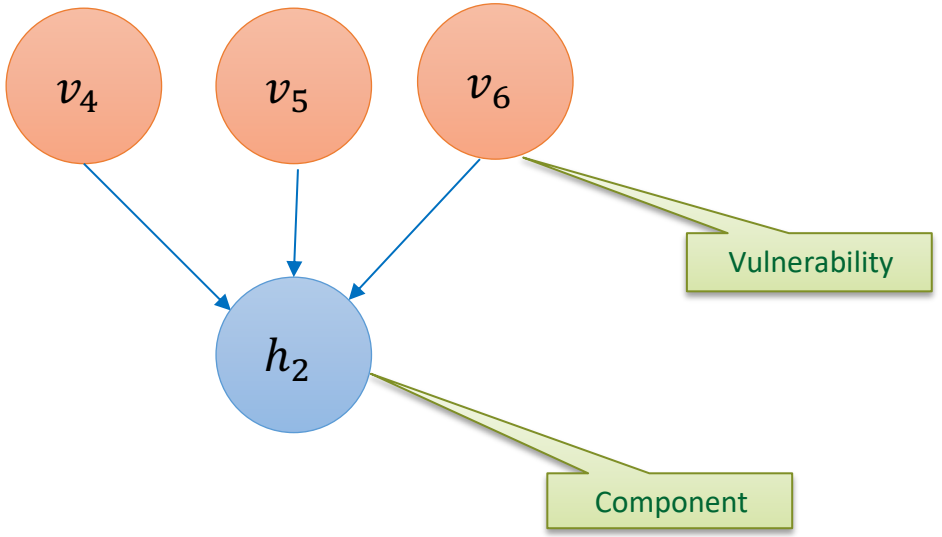
MASON

Generation 1 Metrics

- $\boldsymbol{as_1} = |V^e|$
- Simply counts exposed vulnerabilities
- **What if we have additional information about the vulnerabilities?**
  - Likelihood of vulnerability exploit



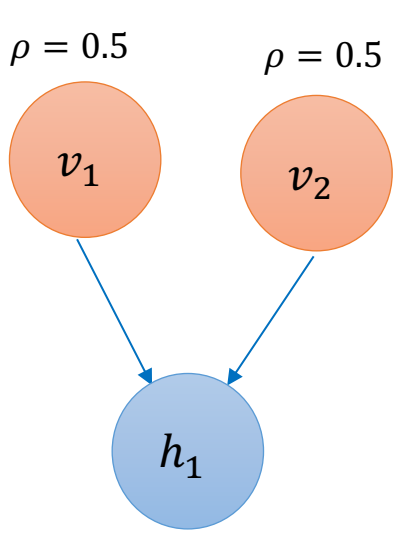
Scenario A



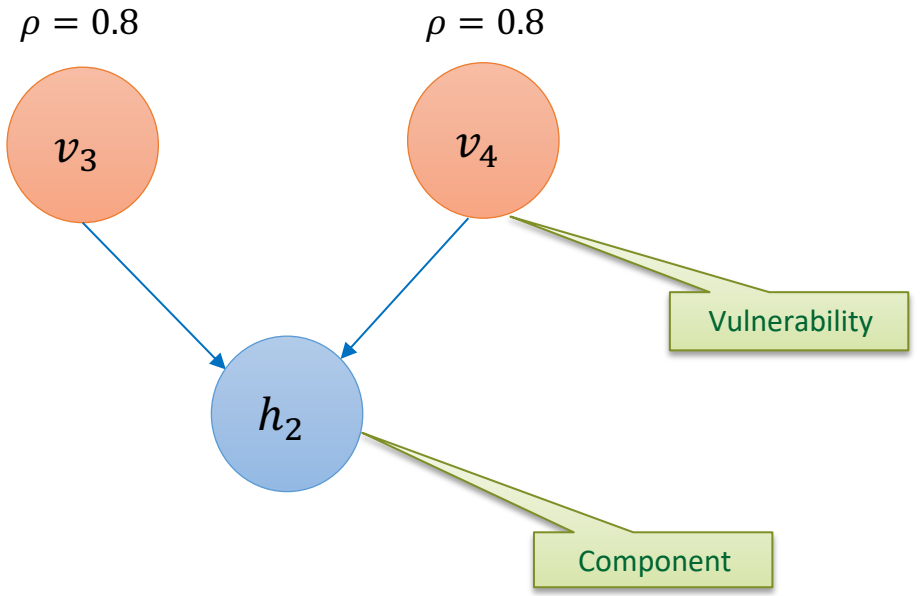
Scenario B

Generation 2 Metrics

- Assume we know the exploitation likelihood of each vulnerability
- Under this assumption, $\boldsymbol{as_1}$ is not sufficient anymore
- $\boldsymbol{as_2} = \sum_{v \in V^e} \rho(v)$
- **What if we have additional information on the vulnerabilities?**
  - Exposure factor



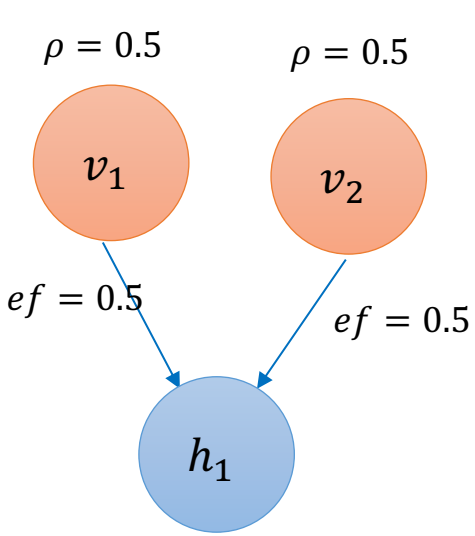
Scenario A



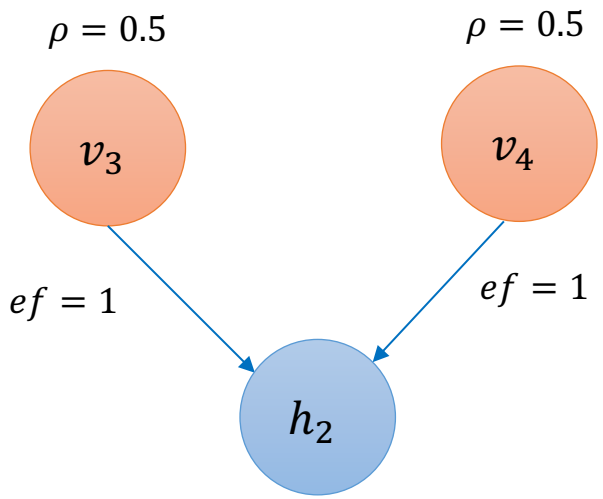
Scenario B

Generation 3 Metrics

- Assume we know the exposure factor of each component to vulnerabilities
- Under this assumption, $\boldsymbol{as_2}$ is not sufficient anymore
- $\boldsymbol{as_3} = \sum_{v \in V^e} \sum_{h \in \{h \in H \mid degrades(v,h)\}} \rho(v) \cdot ef(v,h) \cdot u(h)$
- **What if we have additional information about vulnerabilities that can be exploited after the exploit of the initial vulnerability?**
  - Multi-step attacks



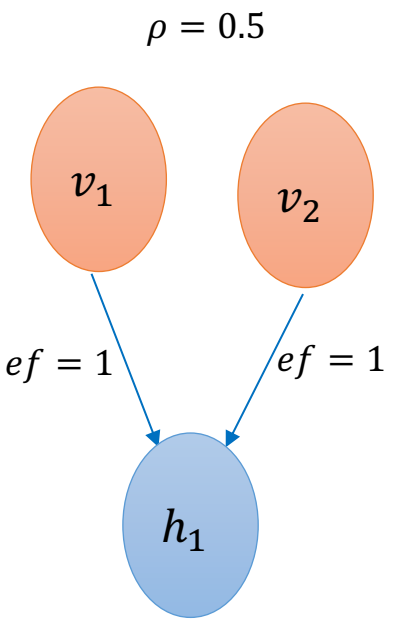
Scenario A



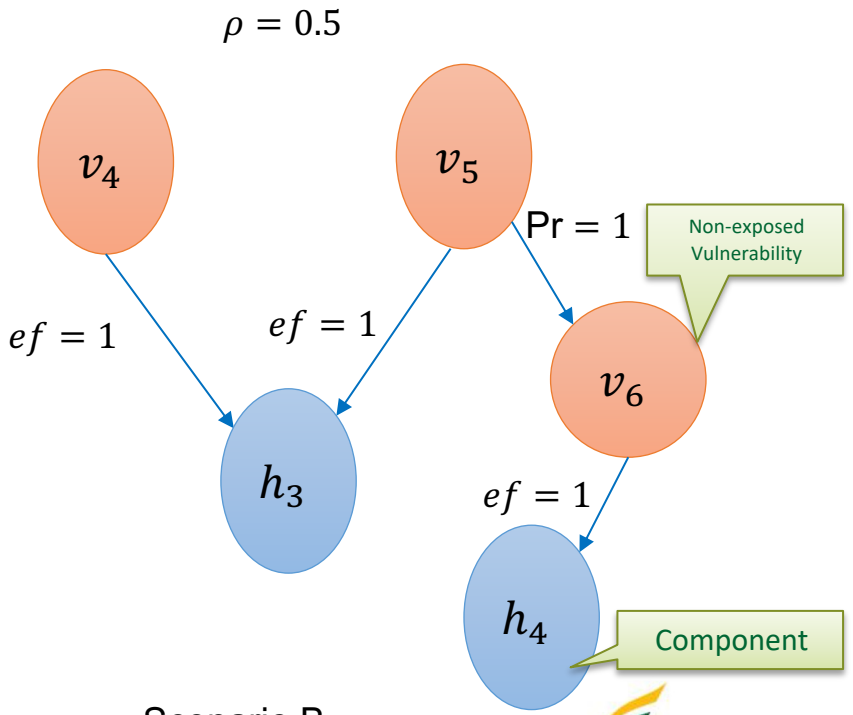
Scenario B

Generation 4 Metrics

- Assume we have information about non-exposed vulnerabilities
- Under this assumption $\boldsymbol{as_3}$, is not sufficient anymore
- Given a non-exposed vulnerability $v \in V \backslash V^e$, we consider its adjusted likelihood
  - $\rho^*(v) = \max_{u \in \{u \in V \mid enables(u,v)\}} \rho^*(u) \cdot \Pr(u,v)$
- $\boldsymbol{as_4} = \sum_{v \in V} \sum_{h \in \{h \in H \mid degrades(v,h)\}} \rho^*(v) \cdot ef(v,h) \cdot u(h)$
- **What if we have additional information about components that are dependent on the degraded component?**



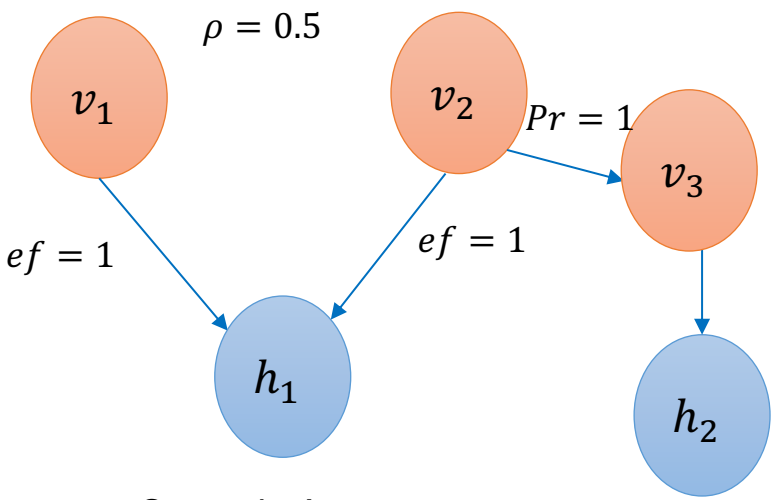
Scenario A

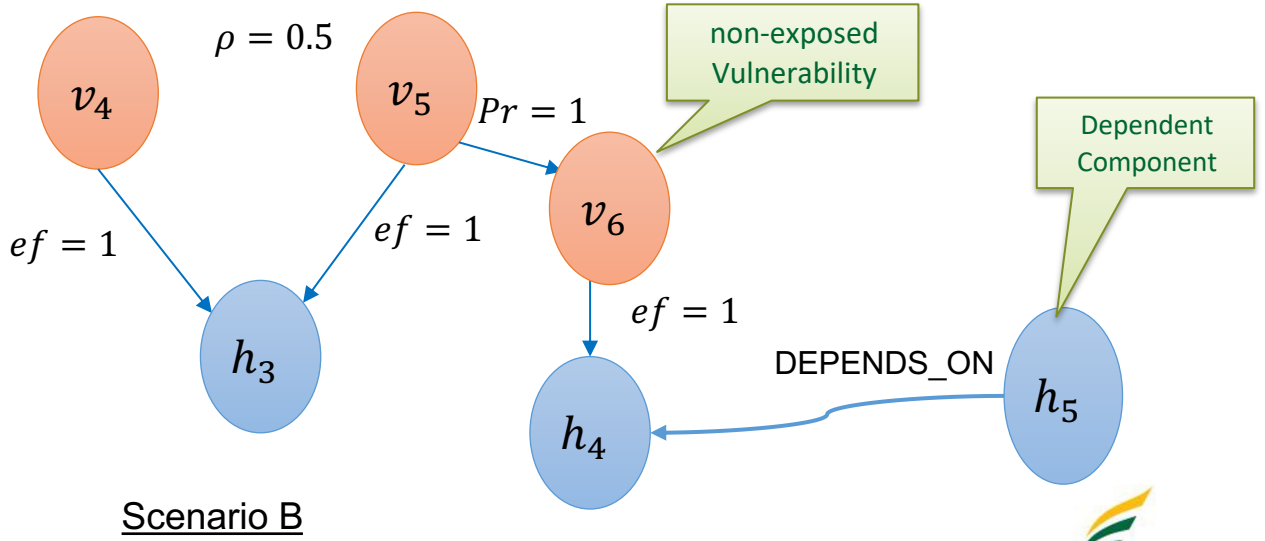


Scenario B

Generation 5

- Assume we know which components are dependent on compromised components
- Under this assumption, $\boldsymbol{as_4}$ is not sufficient anymore





Generation 5 Metrics: Attack Volume Metric

- An attack volume metrics can be defined as

$$avm = \sum_{v \in V} \sum_{h \in H s.t. degrades(v,h)} \rho^*(v) \cdot \left(ef(v,h) \cdot u(h) + \sum_{h^* \in D_\uparrow(h)} u(h^*) \cdot f_{h^*}\left(D_\downarrow(h^*)\right) \right)$$

- where $D_\uparrow(h)$ is the set of components that depend on h, whether directly or through a chain of dependencies, and $D_\downarrow(h^*)$ is the set of components that h^* directly depends on
- The metric computes the cumulative effect of exploiting all vulnerabilities in the system
  - For each exploit, it considers its **impact** on the vulnerable component and how such an impact **propagates** through the chains of dependencies

Experimental Evaluation

Types of Experiments

1. Evaluate the **practical applicability** of AVM to real-world scenarios
2. Evaluate the **effectiveness** of AVM
3. Evaluate the **scalability** of AVM

2 types of datasets

1. Real data – Real data consisting of several testbeds provided by DARPA to all performers in the Configuration Security (ConSec) program,
   - e.g., train control systems and satellite systems
2. Synthetic data – Built a tool to generate graphs of various sizes and complexity

MASON

Practical Applicability within SCIBORG

- To assess SCIBORG's ability to improve a system's configuration, we defined 4 scores, each calculating the Attack Volume for a given system configuration
  - **Worst Case Score (WCS):** measures the attack volume resulting from relaxing all the testbed constraints. This score defines an upper bound on the attack volume
  - **Current Configuration Score (CCS)**: measures the attack volume of the testbed's current configuration
  - **SCIBORG Analysis Score (SAC)**: measures the attack volume induced by the configuration recommended as a result of the SCIBORG analysis
  - **Operational Constraint Score (OCS)**: measures the attack volume induced by the elimination of all infeasible security constraints. This score defines a lower bound on the attack volume
- Expected result: $OCS \leq SAC \leq CCS \leq WCS$

MASON

Results

- The table below reports the four scores for different testbeds
- The scores show that the score pattern is consistently satisfied:
  - OCS ≤ SAC ≤ CCS ≤ WCS
  - AVM accurately measures the testbeds' exposure to cyber attacks

	SAFE TRAINS	VSAT-SPOKE	VSAT-RING	UNIT Tests
Operational Constraint Score (OCS)	0	40	3,220	0
SCIBORG Analysis Score (SAC)	0	40	3,220	40
Current Configuration Score (CCS)	7,960	1,280	3,760	120
Worse Case Score (WCS)	27,610	1,280	22,420	160

MASON

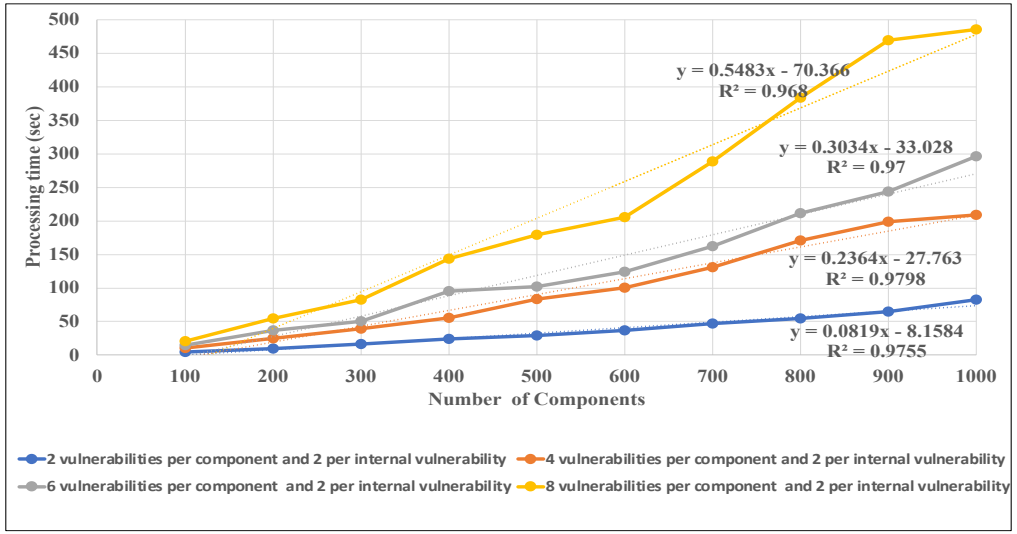
Effectiveness Evaluation

- The following 5 steps were repeated for the 5 scenarios described earlier
  1. Generated a set of n graphs using a baseline configuration (**Scenario A**)
  2. Modified the baseline configuration to generate graphs that metrics in the current generation can discriminate, but metrics in the previous generation cannot discriminate
  3. Generated the second set of n graphs using the modified configuration (**Scenario B**)
  4. Computed the average value of the attack volume for each set of graphs
  5. Verified if the average attack volume for the second set was larger than the first set

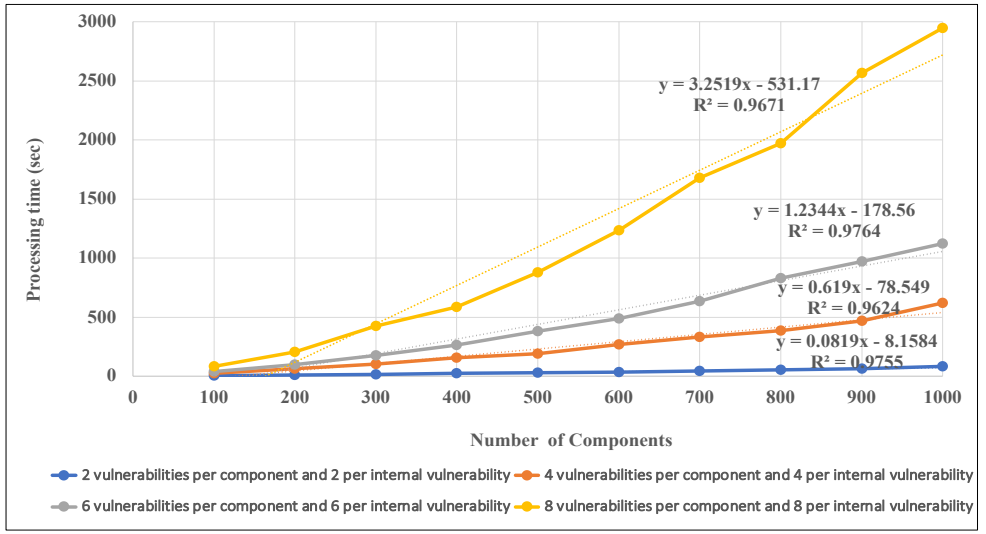
	Scenario A Average Attack Volume	Scenario B Average attack Volume
Generation 1	100	140
Generation 2	225	373
Generation 3	844	1,405
Generation 4	1,405	5,218
Generation 5	5,218	31,602

Result of the Scalability Evaluation

- **Experiment 1**. For a given value of the number of vulnerabilities per component, the computation time **grows linearly** with the number of components
- **Experiment 2**. For a given value of the number of vulnerabilities per component and for a given value of the number of enabling vulnerabilities per internal vulnerability, the computation time **grows linearly** with the number of components



Result of Experiment 1



Result of Experiment 2

Conclusions

- Established the **Attack Volume Metric** to score and compare complex system configurations w.r.t. to their overall vulnerability exposure
  - Results show that AVM can be applied to real-world scenarios
  - Results show that AVM is effective and scalable

- Future work
  - Dynamically updating the model to adapt to evolving vulnerability landscapes
  - Developing resilience against unknown vulnerabilities
    - Reasoning with uncertain or incomplete data

MASON

Questions?

Massimiliano Albanese, PhD
Associate Professor & Associate Chair for Research, Dept. of Information Sciences and Technology
Associate Director, Center for Secure Information Systems
George Mason University
Email: malbanes@gmu.edu
Web: http://csis.gmu.edu/albanese/
LinkedIn: http://www.linkedin.com/in/massimilianoalbanese