Cyber Test and Evaluation Metrics

**James Riordan, David Bigelow,
Joel Schander, and William Streilein**

16 Feb 2016

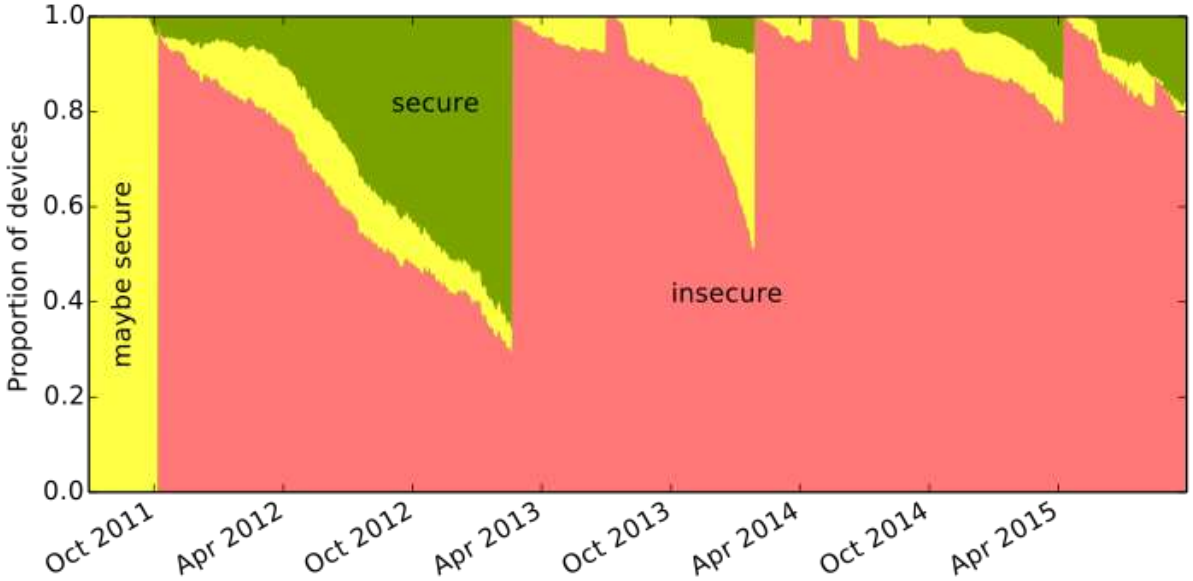
LINCOLN LABORATORY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

This material is based upon work supported under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the U.S. Air Force.

Goals and Observations

- **Objective: Develop improved cyber security metrics and assessment methods to be employed in Developmental Test as to ensure success of cyber capabilities in Operational Test toward providing robustness and resiliency against hostile activity**
- **Fundamental differences between traditional physical systems and cyber systems than necessitate adaptation of test and evaluation criteria**
- **Early and clear statement of these criteria is essential for the operational testing and evaluation processes to ensure that requirement and design decisions of SUTs enable passing operational tests**

Time Scale Differences Between Cyber and Traditional Testing



- **Survivability of cyber systems varies rapidly in time**
  - **Vulnerabilities in software are discovered**
  - **Patches are developed, released, tested, and applied**
- **Continuing survivability involves future events**
  - **Assessing it cannot be purely based on currently known vulnerabilities**
  - **Cyber events occur faster than the time scale of testing**

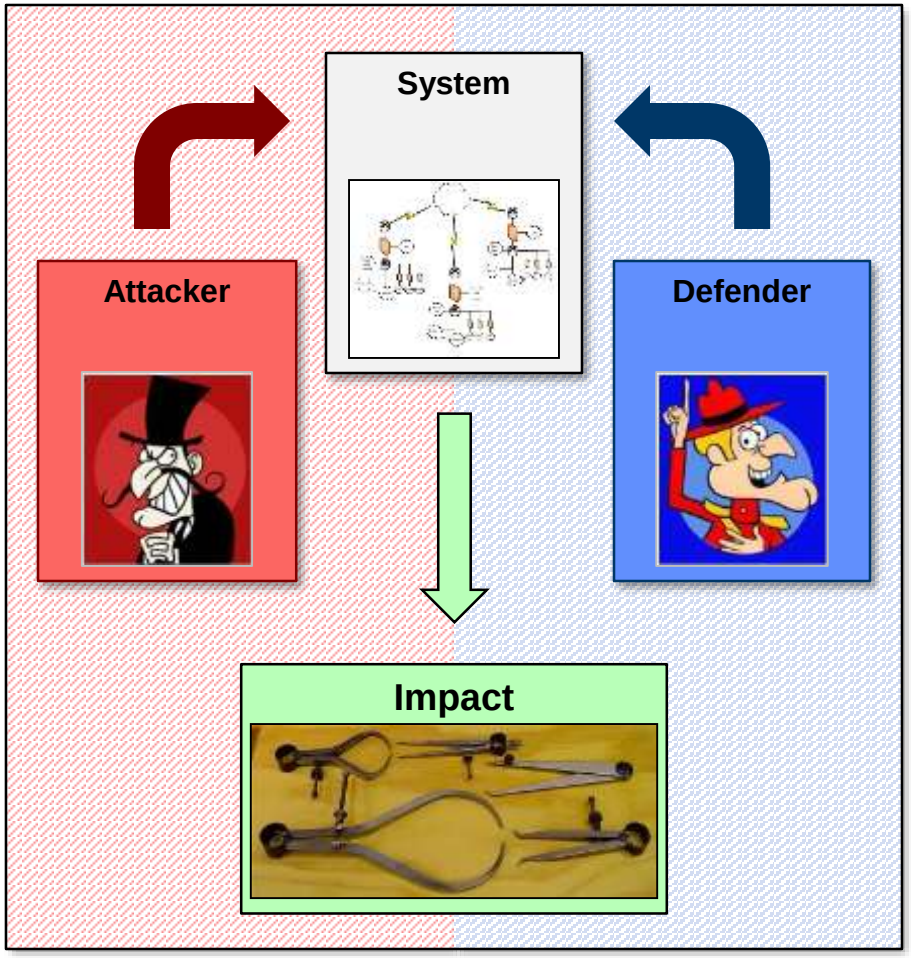
Image taken and analysis from http://androidvulnerabilities.org/

Experience Differences Between Cyber and Traditional Testing

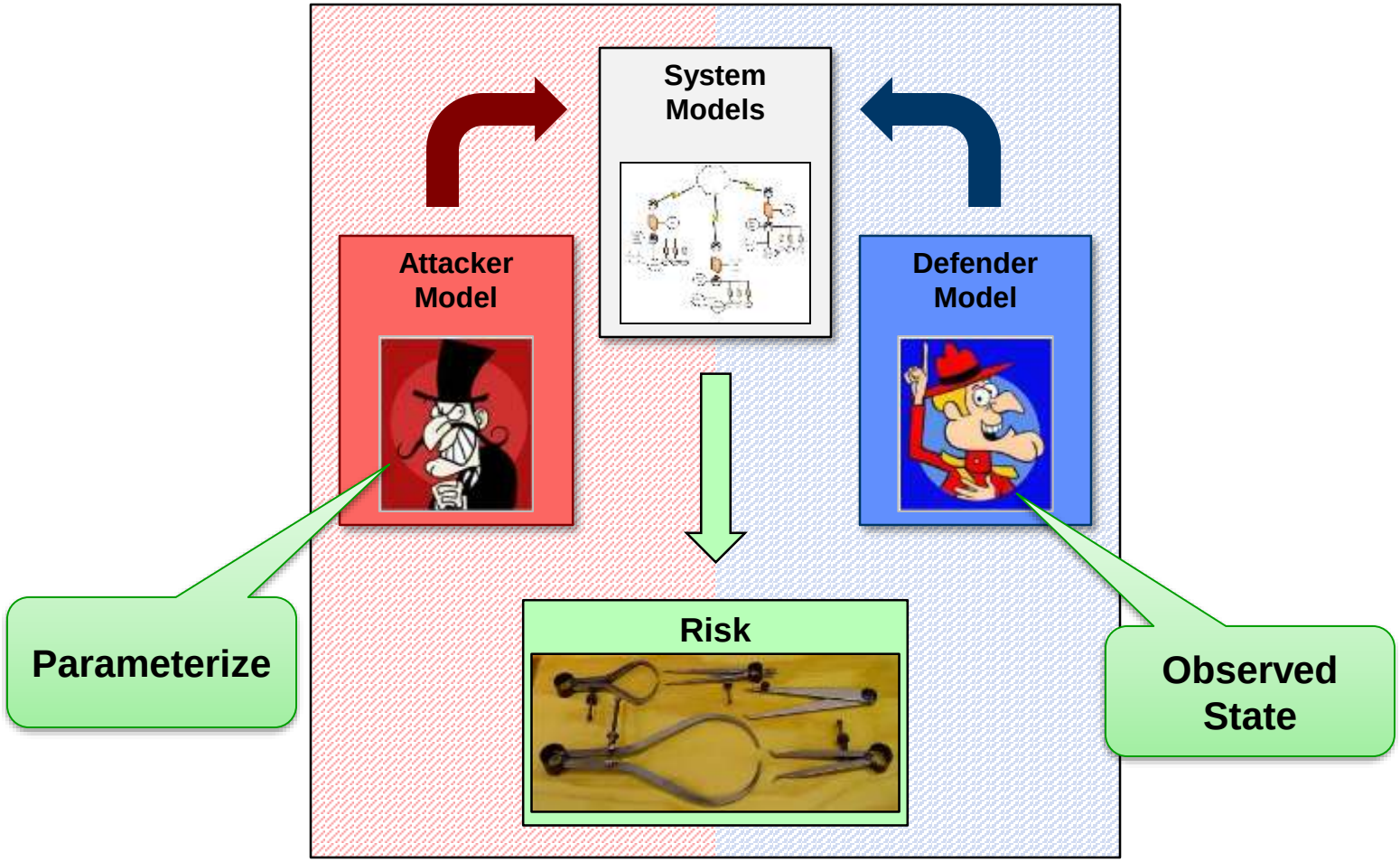
- **Does the system provide robustness and resiliency against hostile activity?**

- **This is not currently possible to answer in an objective, consistent, and repeatable fashion. We lack formal, computable definitions:**
  - **Robustness**
  - **Resiliency**

Security By Its Nature Concerns Attacker / Defender Interaction



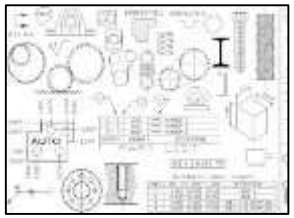
Derive a Parameterized Metric from Attack Model / Evaluate Using Observed Defender Data

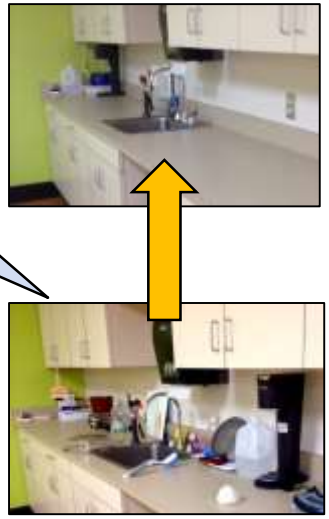


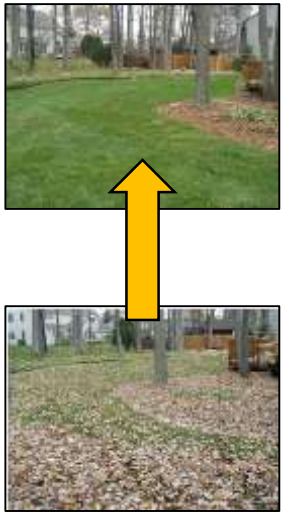
Current Cyber Security Metrics Concern Compliance within Enterprise Networks

Most current cyber security metrics are intended to drive the improvement of enterprise environments.

This is not what is needed in testing.

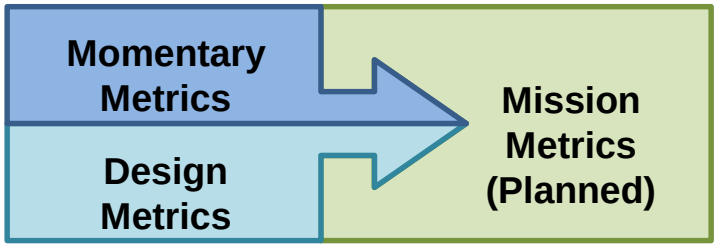






Requirements | **Design** | **Development** | **Developmental Testing** | **Operational Testing** | **Operational Environment**

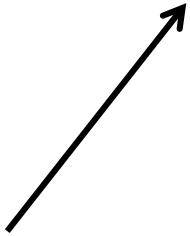
Momentary and Design Metrics



- **The Momentary Metric is applied at the time of testing addresses vulnerabilities present on SUT at the time of testing**
  - **Needs to address:**
    - **The full collection of known vulnerabilities on the SUT**
    - **The severity of the vulnerabilities**
    - **The exploitability of the vulnerability within test context**
- **The Design Metric is applied at the time of testing to assess the continued survivability after testing has completed**
  - **The historical record will be used to generate vulnerability discovery and patch rates of software on the SUT**
- **The Mission Metrics will assess likely mission impact stemming from security posture (computation via probabilistic quantitative analysis)**
  - **Requires a collection of representative missions**

Momentary Metric Formulation (1)

$$P_j\,(\text{compromise}|\text{opportunity}) = \left(\frac{\text{cvss}(j)}{10}\right)^2$$



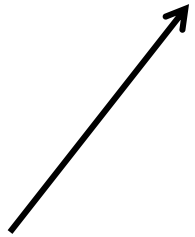
The conditional probability that a device will become compromised via vulnerability j when attacked is a function of the severity of j as estimated by the CVSS

Momentary Metric Formulation (2)

$$P(A) = a$$

$$P(B) = b$$

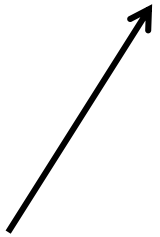
$$P(A \text{ or } B) = 1 - (1 - a) \cdot (1 - b)$$



To compute the probability that at least one of two independent events occurs is computed via the complementary product

Momentary Metric Formulation (3)

$$P\,(\text{opportunity}) = 1 - e^{-\lambda \cdot l}$$



The probability that the attacker has an opportunity to attack the SUT depends on where the SUT is placed and on how long it remains. The parameter λ will describe the environment and l will describe the duration of the test mission.

This formula describes a Poisson arrival process and the formulate describes the matching exponential distribution of at least one event occurring.

Momentary Metric Formulation

$$P\left(\text{compromise}\right) = P\left(\text{opportunity}\right) \cdot P\left(\text{compromise}|\text{opportunity}\right)$$

$$P\left(\text{compromise}\right) = \left(1 - e^{-\lambda \cdot l}\right) \cdot \left[1 - \prod_{j \in V} \left(1 - \left(\frac{\text{cvss}(j)}{10}\right)^2\right)\right]$$

Putting it all together.

In point of fact, we will have to compute different opportunity probabilities for different attack types (server-side, client-side, USB,…) but these can be computed independently and combined.

Design Metric Formulation (1)

$$\Delta^{i}_{\text{arrival}} = \left[\sum_{j \in V(i,w)} \left(\frac{\text{cvss}(j)}{10} \right)^2 \right] \cdot |w|^{-1}$$

↑

The mean of arrivals of vulnerabilities for a given package is computed from the historical rate of vulnerabilities for that package. For a given package we computed the normalized total of vulnerabilities over a standard period dividing by the length of that period. Suggested time period is 3 years.

$$\lambda^{i}_{\text{arrival}} = \left(\Delta^{i}_{\text{arrival}} \right)^{-1}$$

↑

The mean is inverted to compute rate.

Design Metric Formulation (2)

$$\lambda^i_{\text{patch}}$$

↑

The rate of release of patches after vulnerability discover should be set from historical records when available. If not available it may be found in Service Level Agreements. If no such agreements exist, it should be take as an approximation of the time to get the vulnerability fixed.

$$\lambda^i_{\text{install}}$$

↑

The rate of patch installation may be informed by history. This needs parameter needs significant discussion – please see speaker notes.

Design Metric Formulation

$$P\left(\text{vulnerable}_i\right) = \frac{\lambda^i_{\text{arrival}}}{\lambda^i_{\text{arrival}} + \lambda^i_{\text{patch}} + \lambda^i_{\text{install}}}$$

$$P\left(\text{vulnerable}\right) = 1 - \prod_{i \in S} \left(1 - P\left(\text{vulnerable}_i\right)\right)$$

The probability that a SUT will be vulnerable due to software on the SUT will be computed as a complementary product that it will be vulnerable due to each package. The set S will describe the software.

Example Metric Computation: Vulnerabilities

- **SUT has two exposed packages and three identified vulnerabilities**

Package	CVE	CVSS
Apache Web Server	CVE-2008-0456	2.6
Drupal Content Manager	CVE-2008-0273	4.3
	CVE-2008-0276	4.3

- **Per agreement at program start time**
  - **SUT to be placed on NIPR Net**
  - **NIPR attack rate = 1 attack / day**
  - **Mission duration 1 week**
- **Momentary**
  - **Probability that a system will become compromised given opportunity**
    - **$1 - (1 - 0.26^2) \cdot (1 - 0.43^2) \cdot (1 - 0.43^2) \approx 38\%$**
  - **Probability that there will be an opportunity**
    - **$1 - \exp(7 \text{ days} \cdot (1 \text{ attack/day})) \approx 99\%$**
  - **Probability of system compromise**
    - **38% · 99% ≈ 38%** ✗

Example Design Metric Computation: Vulnerabilities

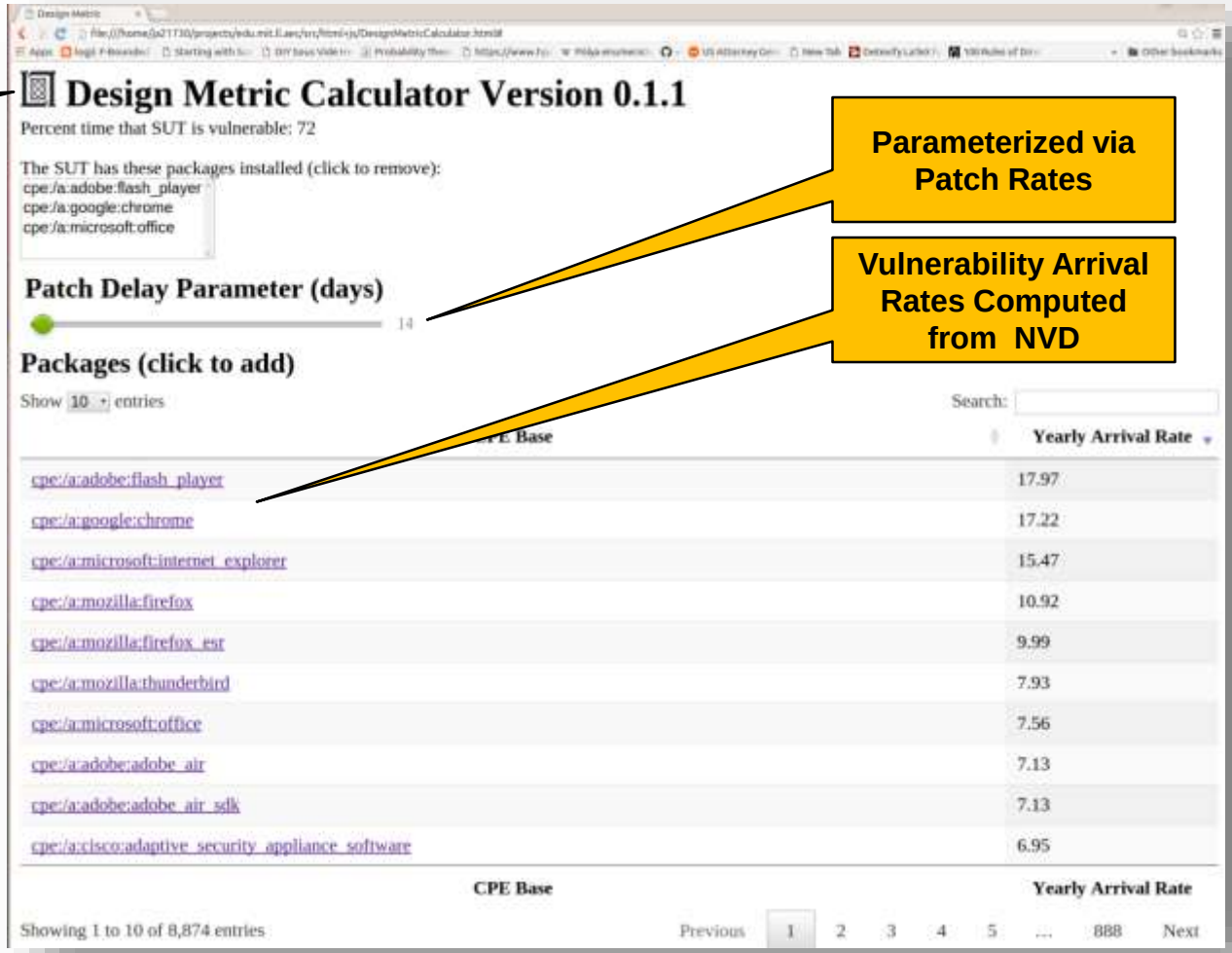
- **SUT has two install packages: Apache and Drupal**

Apache	Drupal
$\lambda_{arrival}$ = 0.112 vulnerability / week	$\lambda_{arrival}$ = 0.33 vulnerability / week
λ_{patch} = 1 patch / week	λ_{patch} = .4 patch / week
$\lambda_{install}$ = 2 install / week	$\lambda_{install}$ = 0.2 install / week
$P(\text{vulnerable}_{apache}) = \frac{0.112}{0.112+1+2} \approx 0.036$	$P(\text{vulnerable}_{drupal}) = \frac{0.33}{0.33+0.4+0.2} \approx 0.35$

- **Computation**
  - **Probability that a system will become compromised given opportunity**
    - **1 – (1- 0.036)·(1-0.35) ≈ 0.37**
  - **Probability that there will be an opportunity**
    - **1 – exp(7 days · (1 attack/day)) ≈ 99%**
  - **Probability of system compromise**
    - **37% · 99% ≈ 37%** ✗
- **Variation if SUT were on SIPR Net instead of Internet**
  - **Probability that there will be an opportunity**
    - **1 – exp(7 days · (1 attack / 31 day)) ≈ 20%**
  - **Probability of system compromise**
    - **37% · 20% ≈ 7.6%** ✓

Design Metrics Initial Implementation

Implemented as Web Page to meet AEC Requirements



Parameterized via Patch Rates

Vulnerability Arrival Rates Computed from NVD

Path to Metric Driven Testing

- **Steps necessary to advance metrics driven computation**
  - **Establish and publish core metric parameters**
  - **Establish and publish vulnerability rates from historical data (NVD)**
    - **Clean up this data**
  - **Extend metrics to other identified concerns (misconfigurations & identification/authentication)**
  - **Document data matrix requirements (from ontology)**
  - **Refine metrics according to continuously developing understanding**
  - **Provide further documentation of approach to earlier stages of acquisitions**

Cybersecurity Metrics – Lessons Learned

- **Useful cybersecurity metrics should be:**
  - **Readily implementable within the relevant process**
  - **Designed to provide actionable information to a specific audience**
  - **Tailored to address specific, relevant cybersecurity issues**
  - **Support an improved decision-making process for the audience**
  - **Be implemented within a specific scope and timeframe**
  - **Developed through stakeholder collaboration and peer review**
  - **Be associated with "pass" and "fail" criteria, to facilitate discussion**

- **Carefully define all of the above before designing cybersecurity metrics!**