# BENCHMARKING THE SECURITY OF SOFTWARE SYSTEMS OR TO BENCHMARK OR NOT TO BENCHMARK

**DESSERT'2018**
Kiev, Ukraine
May 26th, 2018


**Marco Vieira**
mvieira@dei.uc.pt
Department of Informatics Engineering
**University of Coimbra - Portugal**


# BENCHMARKING

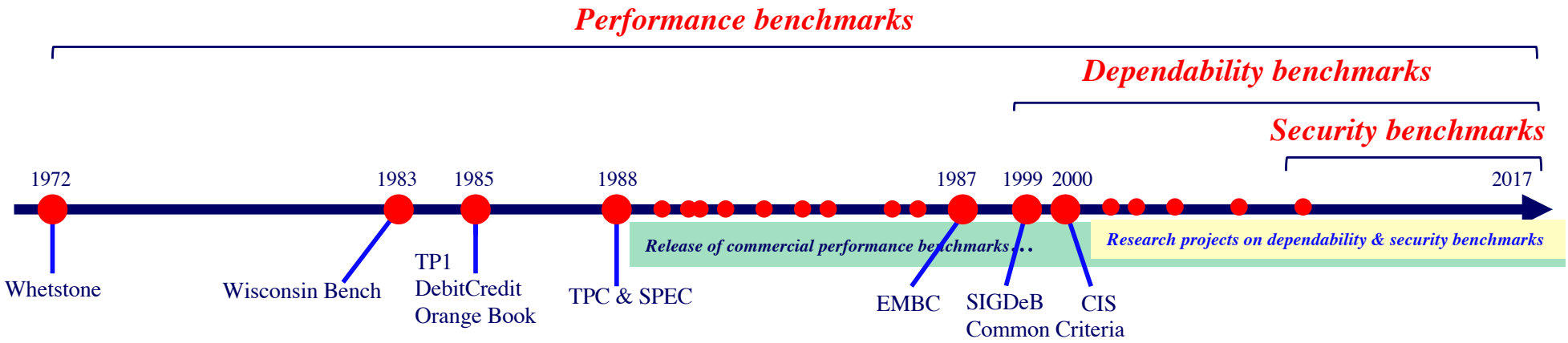
**Assessing and comparing computer systems and/or components according to specific quality attributes**

- Performance benchmarking
  - Well established both in terms of research and application
  - Supported by organizations like TPC and SPEC
  - Mostly for marketing
- Dependability benchmarking
  - Well established from a research perspective
  - No endorsement from the industry

# BENCHMARKING

**Assessing and comparing computer systems and/or components according to specific quality attributes**

- Security benchmarking
  - Several works can be found
  - No common approach available yet



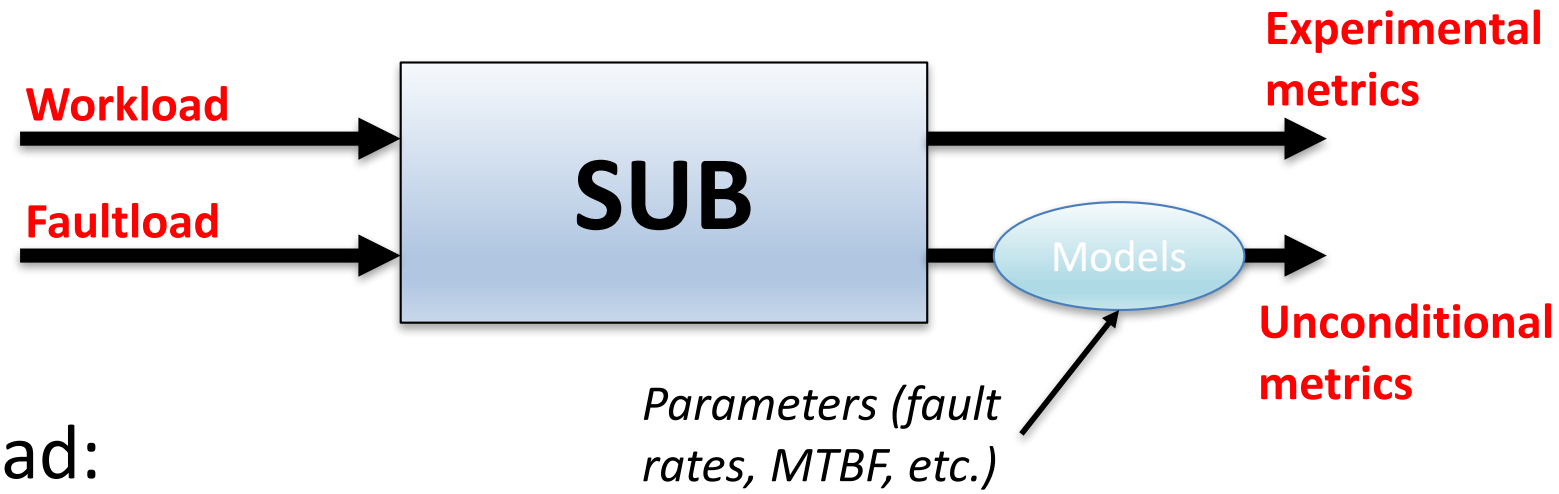
# OUTLINE

- The past: Dependability Benchmarking
- The present: Security Benchmarking
- Benchmarking the **Security of Systems**
  - Approach: Qualification + Trustworthiness Assessment
  - Example: Benchmarking Web Service Frameworks
- Benchmarking **Security Tools**
  - Approach: Vulnerability and Attack Injection
  - Example: Benchmarking Intrusion Detection Systems
- Challenges and Conclusions

# DEPENDABILITY BENCHMARKING

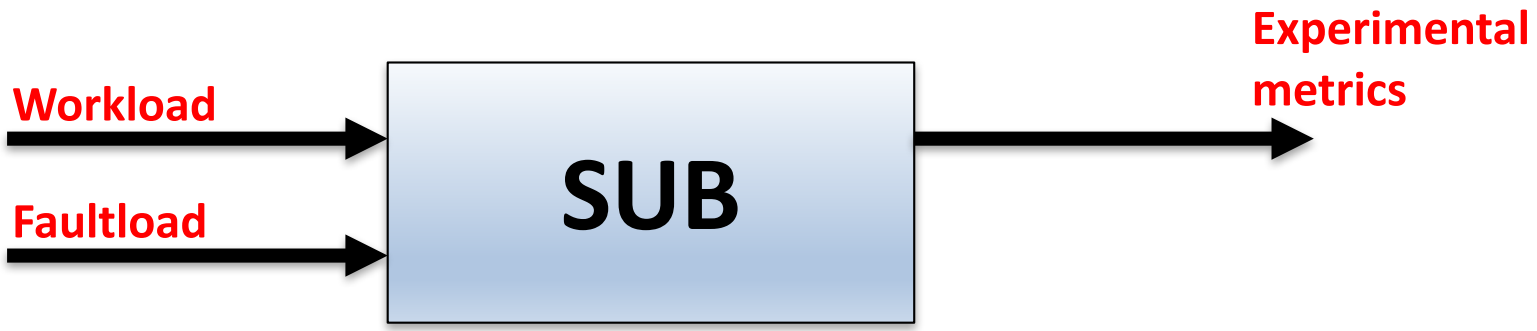
**Assessing and comparing**
**computer systems and/or components**
**considering dependability attributes**

# DEPENDABILITY BENCHMARKING




- Faultload:
  - Set of representative faults, injected into the system
- Metrics:
  - Performance and/or dependability
    - Both baseline and in the presence of faults
  - Unconditional and/or direct

# DBENCH-OLTP (2005)



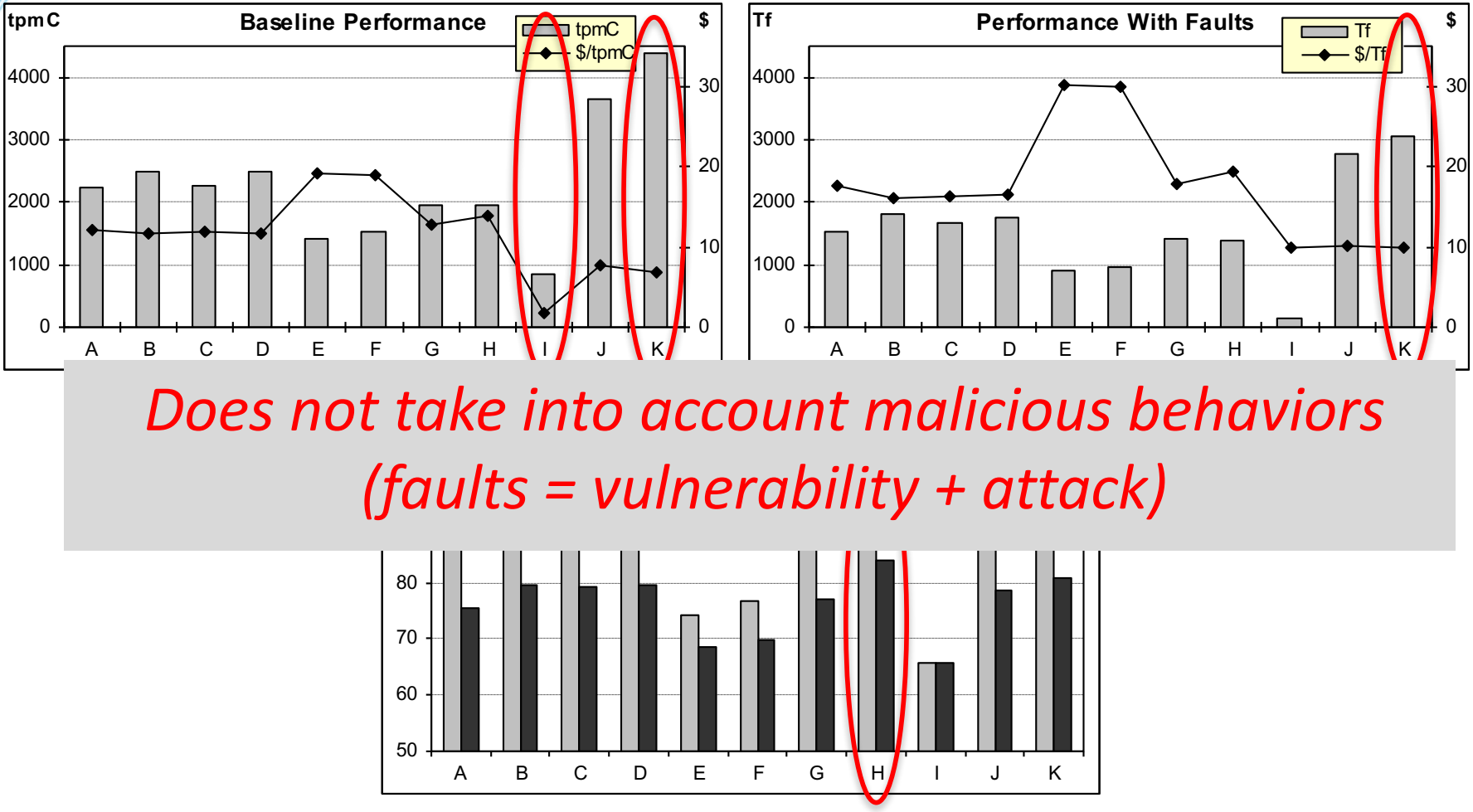

- Workload:
  - TPC-C transactions
- Faultload:
  - Operator faults + Software faults + HW component failures
- Metrics:
  - Performance: tpmC, $/tpmC, Tf, $/Tf
  - Dependability: Ne, AvtS, AvtC

# DBENCH-OLTP (2005)

| System | Operating System | DBMS | DBMS Config. | Hardware |
|---|---|---|---|---|
| A | Windows 2K Prof. SP 3 | Oracle 8i R2 (8.1.7) | Config. A | *Processor*: Intel Pentium III 800 MHz<br>*Memory*: 256MB<br>*Hard Disks*: Four 20GB/7200 rpm<br>*Network*: Fast Ethernet |
| B | Windows 2K Prof. SP 3 | Oracle 9i R2 (9.0.2) | Config. A | |
| C | Windows Xp Prof. SP 1 | Oracle 8i R2 (8.1.7) | Config. A | |
| D | Windows Xp Prof. SP 1 | Oracle 9i R2 (9.0.2) | Config. A | |
| E | Windows 2K Prof. SP 3 | Oracle 8i R2 (8.1.7) | Config. B | |
| F | Windows 2K Prof. SP 3 | Oracle 9i R2 (9.0.2) | Config. B | |
| G | SuSE Linux 7.3 | Oracle 8i R2 (8.1.7) | Config. A | |
| H | SuSE Linux 7.3 | Oracle 9i R2 (9.0.2) | Config. A | |
| I | SuSE Linux 7.3 | PostgreSQL 7.3 | - | |
| J | Windows 2K Prof. SP 3 | Oracle 8i R2 (8.1.7) | Config. A | *Processor*: Intel Pentium IV 2GHz<br>*Memory*: 512MB<br>*Hard Disks*: Four 20GB/7200 rpm<br>*Network*: Fast Ethernet |
| K | Windows 2K Prof. SP 3 | Oracle 9i R2 (9.0.2) | Config. A | |

**Faultload: Operator faults**

# DBENCH-OLTP (2005)



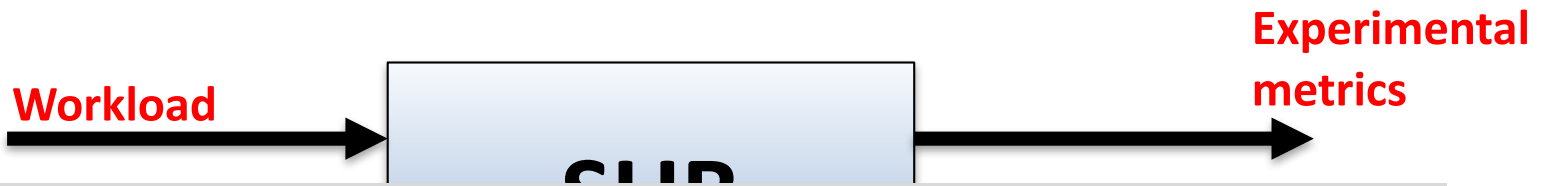

*Does not take into account malicious behaviors (faults = vulnerability + attack)*

# SECURITY BENCHMARKING

**Assessing and comparing**
**computer systems and/or components**
**considering security aspects**

- Benchmarking the Security of **Systems / Components**
  - Systems that should implement security requirements
  - OS, middleware, server software, etc.
- Benchmarking **Security Tools**
  - Tools used to improve the security of systems
  - Penetration testers, static analyzers, IDS, etc.

# BENCHMARKING SECURITY OF SYSTEMS




*Attacking what? Do we know the vulnerabilities? What are representative attacks?*

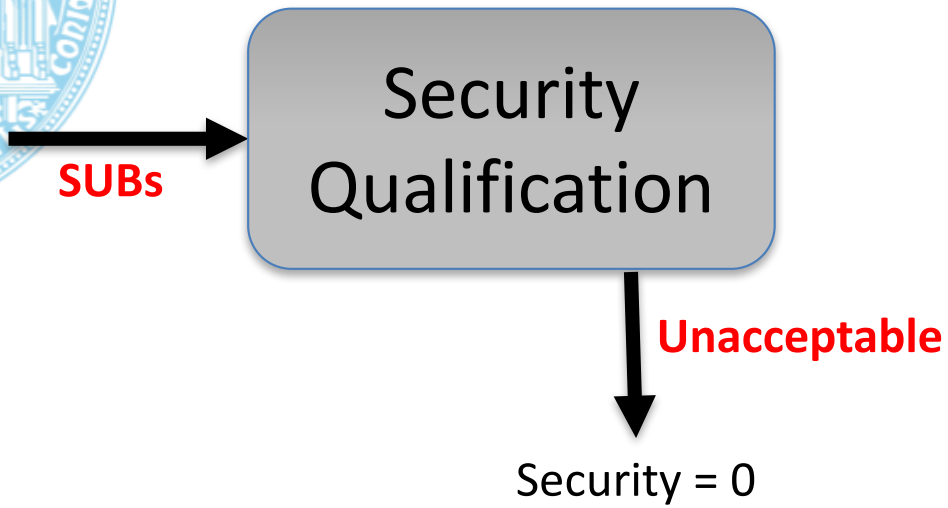
*Does not work if one wants to benchmark how secure different systems are!*

*e.g. does the number of vulnerabilities of a system represent anything?*

onal

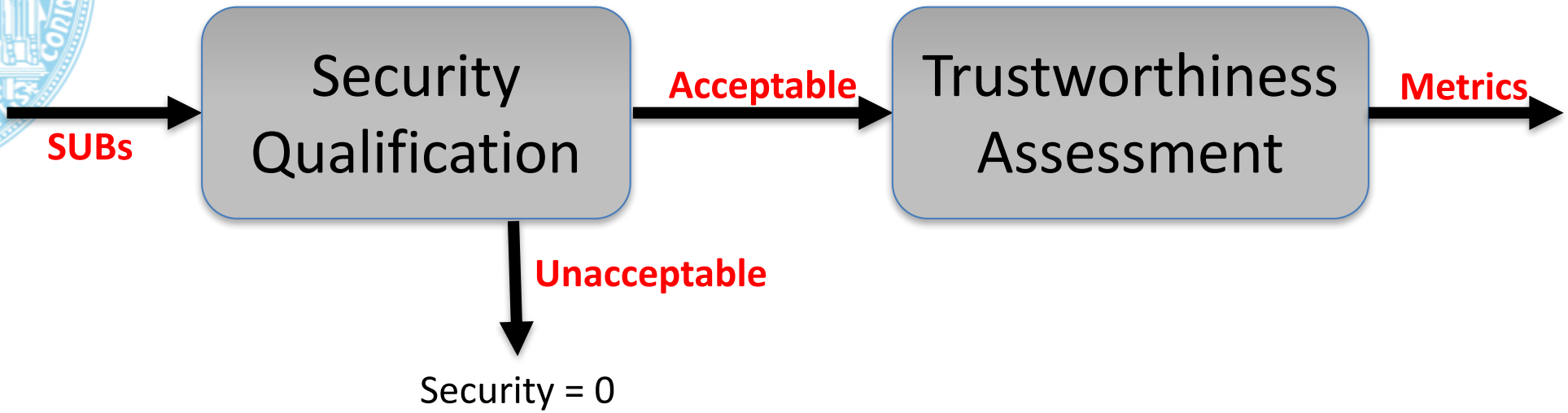
- Performance + dependability
- Security (e.g., number vulnerabilities, attack detection)

# A DIFFERENT APPROACH…



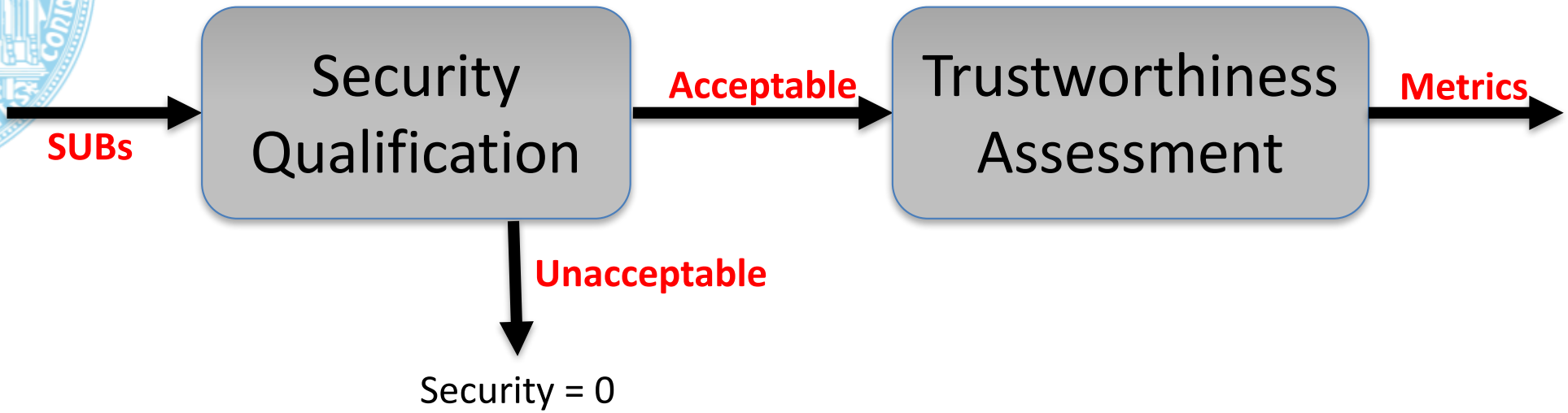

- Security Qualification:
  - Apply state-of-the-art techniques and tools to detect vulnerabilities
  - SUBs with vulnerabilities are:
    - Disqualified!
    - Or vulnerabilities are fixed…

# A DIFFERENT APPROACH…



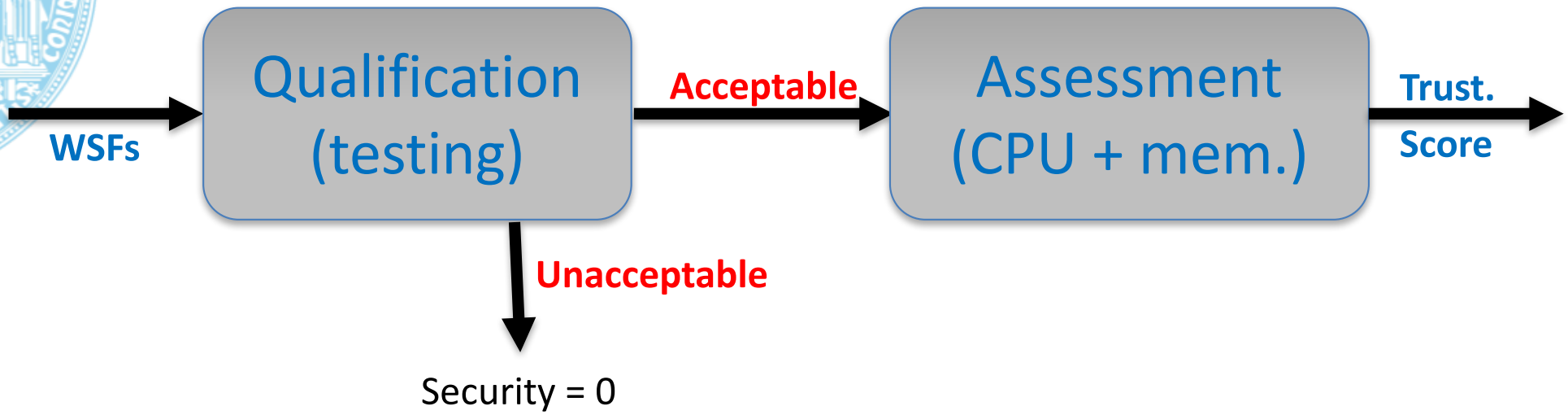

- Trustworthiness Assessment:
  - Gather evidences on how much one can trust
  - e.g., best coding practices, development process, bad smells

# A DIFFERENT APPROACH…



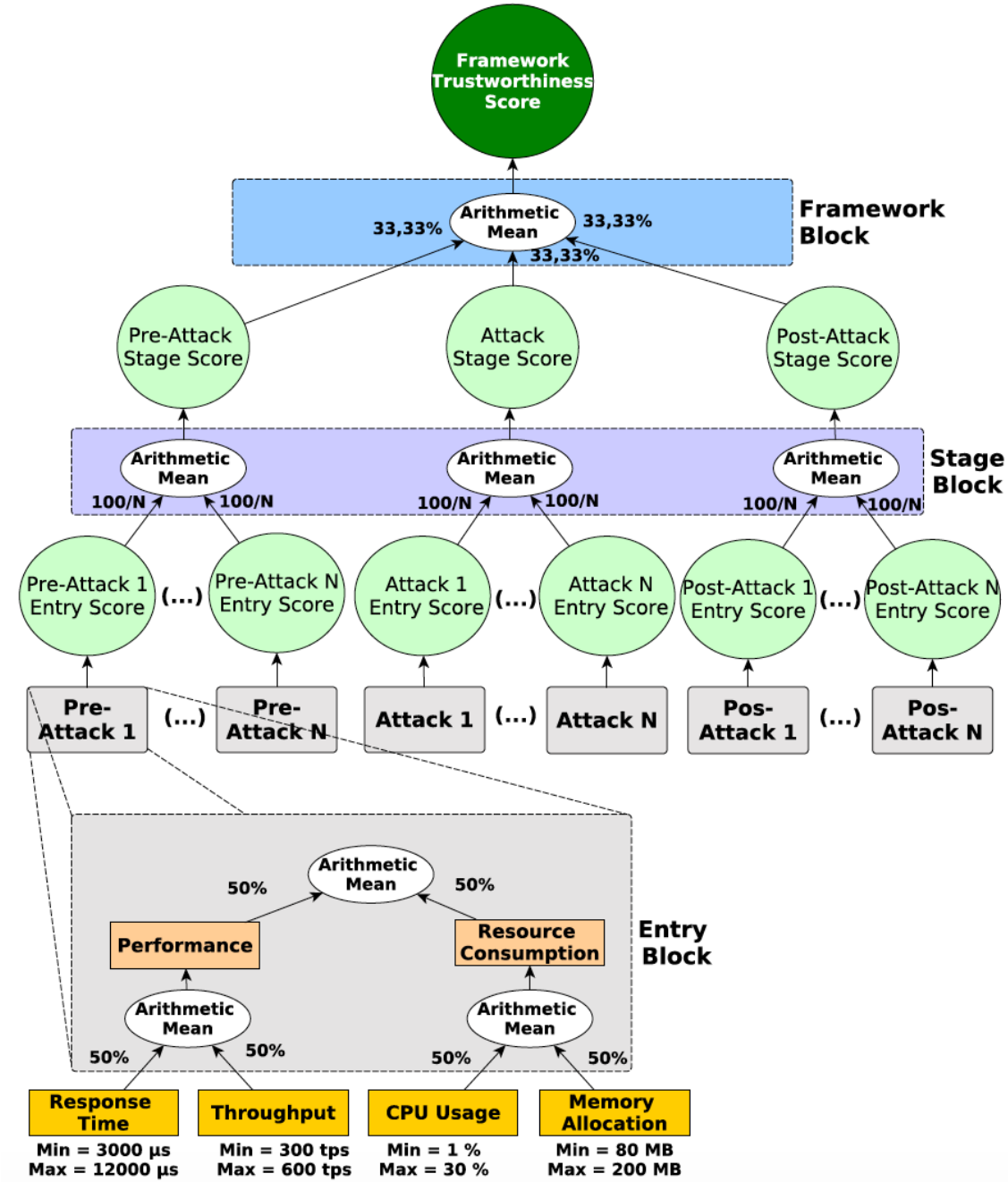

- Metrics:
  - Portray trust from a user perspective
  - Dynamic: may change over time
  - Depend on the type of evidences gathered
  - Different metrics for different attack vectors

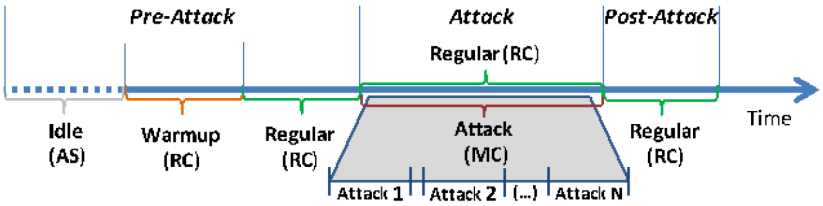
# EXAMPLE: WEB SERVICE FRAMEWORKS

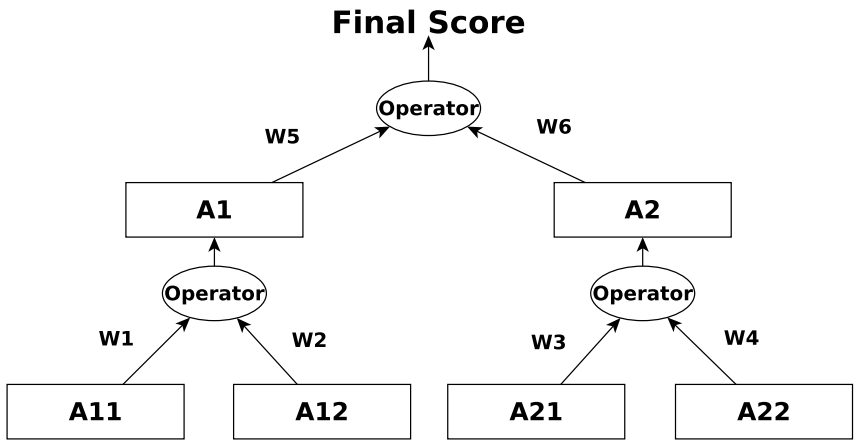



- Qualification
  - DoS Attacks
  - *Coercive Parsing, Malformed XML, Malicious Attachment, etc.*
- Trustworthiness Assessment:
  - Quality model to compute a score

# QUALITY MODEL



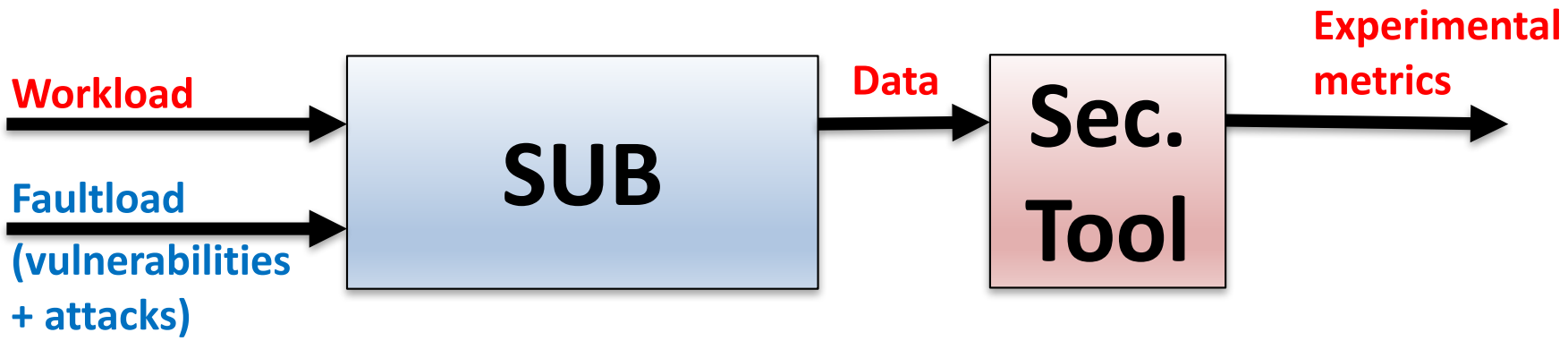
# SYSTEMS UNDER BENCHMARKING

| Framework | Version | Security Qualification |
|---|---|---|
| Apache Axis 1 | 1.4.1 | ✗ |
| Apache Axis 2 | 1.6.1 | ✓ |
| | 1.6.2 | ✗ |
| Apache CXF | 2.5.1 | ✓ |
| | 3.0.3 | ✓ |
| Oracle Metro | 2.1.1 | ✗ |
| | 2.3.1 | ✓ |
| XINS | 3.1 | ✗ |
| Spring JAX-WS | 1.9 | ✗ |
| Spring WS | 2.2.0 | ✗ |

# TRUSTWORTHINESS RESULTS

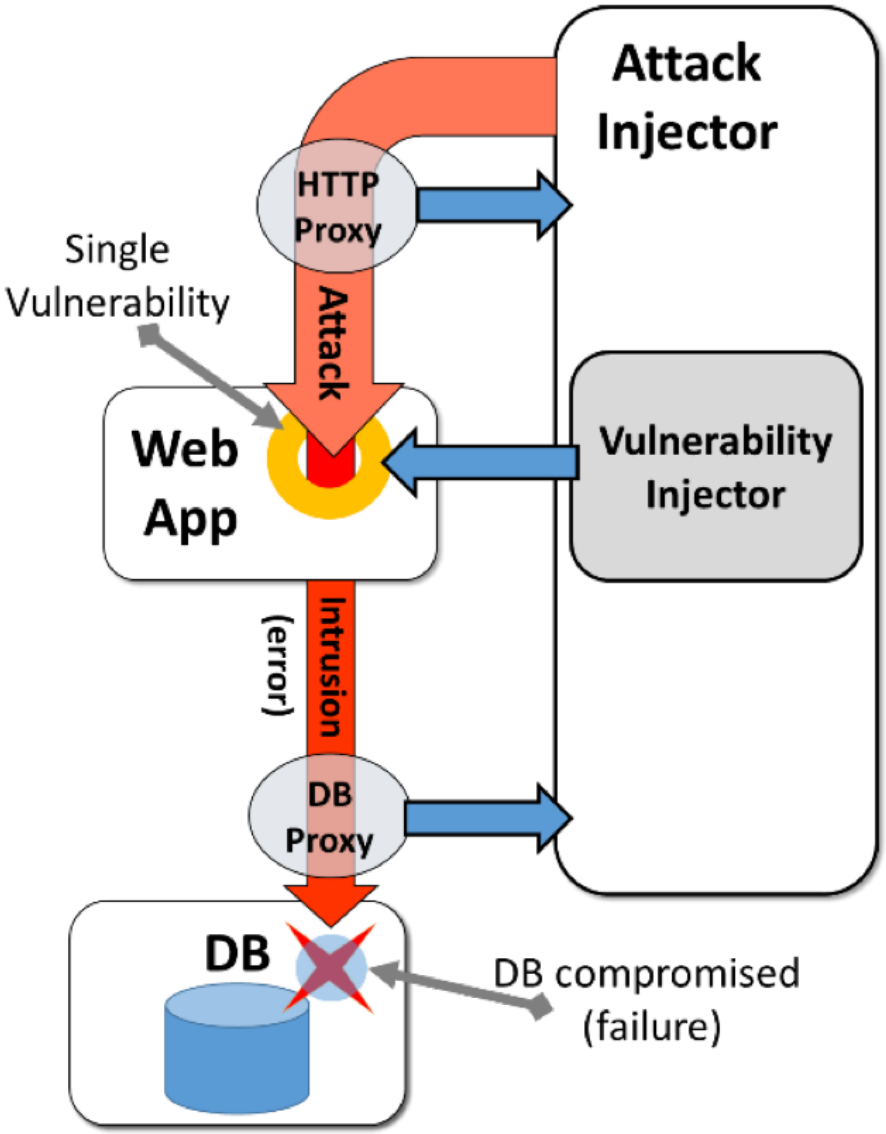
| Scenario | Axis 2 | CXF v2 | Metro | CXF v3 |
|---|---|---|---|---|
| *Neutral* | 72.3 (1) | 70.7 (2) | 58.1 (3) | 57.9 (4) |
| *Scenario1* | 73.4 (2) | 77.1 (1) | 66.5 (4) | 70.0 (3) |
| *Scenario2* | 67.4 (3) | 73.1 (1) | 66.6 (4) | 68.7 (2) |
| *Scenario3* | 61.8 (4) | 70.3 (1) | 63.6 (3) | 67.0 (2) |

# BENCHMARKING SECURITY TOOLS




- Faultload:
  - Vulnerabilities are injected
  - Attacks target the injected vulnerabilities
- Data can be collected for benchmarking security tools
  - Penetration testers, static analyzers, IDS, etc.

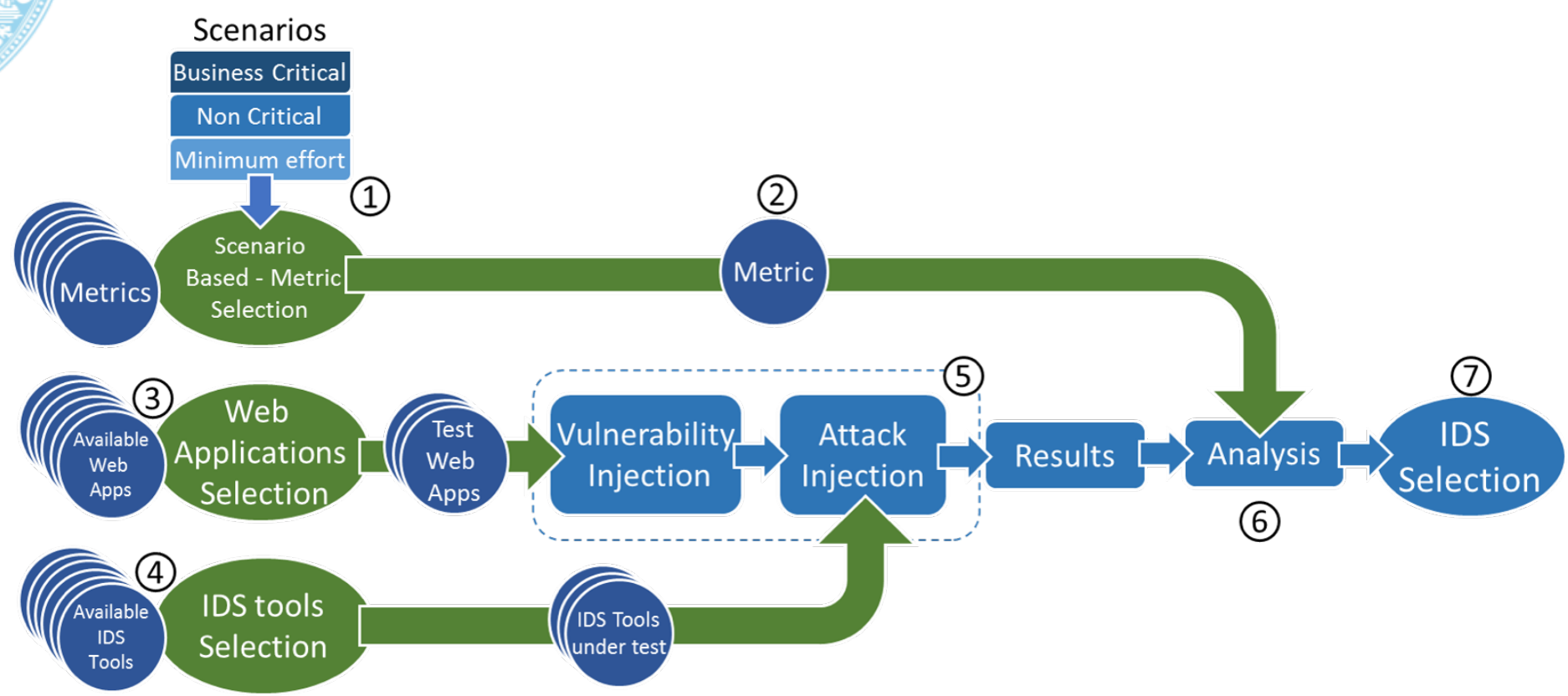
# VULNERABILITY AND ATTACK INJECTION



# EXAMPLE: BENCHMARKING IDS

- Security requires a defense in depth approach
  - Coding best practices
  - Testing
  - Static analysis
  - …
- Vulnerability-free code is hard (or even impossible) to achieve...
- Intrusion detection tools support a post-deployment approach
  - For protecting against known and unknown attacks

# Evaluation Approach



# EXAMPLES OF VULNERABILITIES INJECTED

| Original PHP code | Code with injected vulnerability | Operation performed |
|---|---|---|
| $id=intval($_GET['id']); | $id=$_GET['id']; | Removed the "intval" function allowing also non numeric values (i.e. SQL commands) in the "$id" variable |
| $page = urlencode($page); | $page = $page; | Removed the "urlencode" function allowing also alphanumeric values (i.e. SQL commands) in the "$page" variable |
| … | … | … |

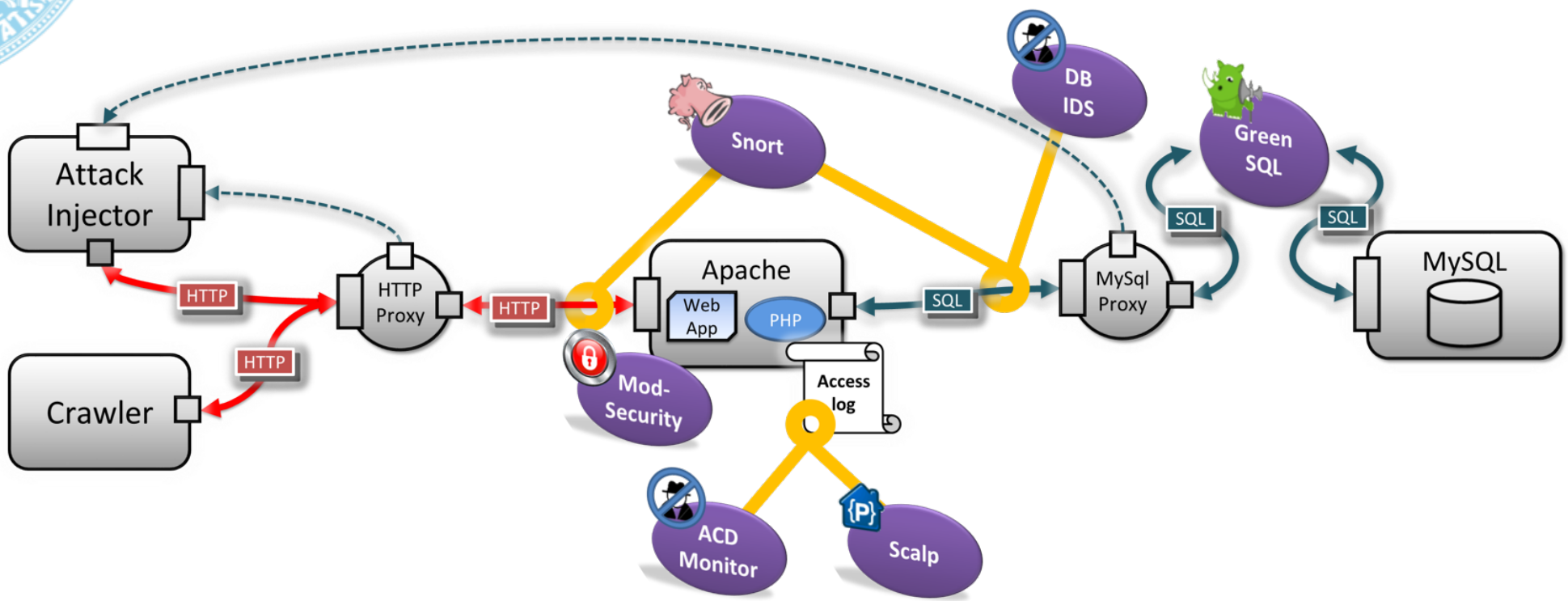
# EXAMPLES OF ATTACKS

| Attack payloads | Expected result |
|---|---|
| ' | Modifies the structure of the query; usually results in an error |
| or 1=1 | Modifies the structure of the query. Overrides the query restrictions by adding a statement that is always true. |
| ' or 'a'='a | Modifies the structure of the query. Overrides the query restrictions by adding a statement that is always true. |
| +connection_id()-connection_id() | Modifies the query result to 0 |
| +1-1 | Modifies the query result to 0 |
| +67-ASCII('A') | Modifies the query result to 0 |
| +51-ASCII(1) | Modifies the query result to 0 |
| … | … |

# SYSTEMS UNDER BENCHMARKING

| Tool | Architectural Level monitored | Detection Approach | Data Source | Known Technology Limitations |
|---|---|---|---|---|
| *ACD* | Application | Anomaly Based | Apache Log | Only GET method |
| *Apache Scalp* | Application | Signature Based | Apache Log | Only GET method |
| *ModSecurity* | Application | Signature Based | HTTP traffic | - |
| *Snort (v2.8 and v2.9)* | Network | Signature Based | Network Trafic | - |
| *GreenSQL* | Database | Signature Based | SQL Proxy Trafic | MySQL data |
| *DB IDS* | Database | Anomaly Based | SQL Sniffer Trafic | MySQL and Oracle data |

# EXPERIMENTAL SETUP



# Main Results

**All**

| lvl | Tool | Review | | | Reported | | | | Prec. | Recall | Mark. | Infor. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | P | N | Pop | TP | TN | FN | FP | | | | |
| App | ACD | 1051 | 224 | 1275 | 376 | 174 | 675 | 50 | 0.883 | 0.358 | 0.088 | 0.135 |
| App | Scalp | | | 1275 | 206 | 224 | 845 | 0 | 1.000 | 0.196 | 0.210 | 0.196 |
| App | ModSecurity | 826 | 225 | 1051 | 236 | 225 | 590 | 0 | 1.000 | 0.286 | 0.276 | 0.286 |
| Net | Snort 2.8 | 458 | 817 | 1275 | 0 | 817 | 458 | 0 | - | 0.000 | - | 0.000 |
| DB | GreenSQL | | | 1275 | 244 | 813 | 214 | 4 | 0.984 | 0.533 | 0.775 | 0.528 |
| DB | DB IDS | | | 1275 | 451 | 384 | 7 | 433 | 0.510 | 0.985 | 0.492 | 0.455 |
| Net | Snort 2.9 | 173 | 878 | 1051 | 0 | 878 | 173 | 0 | - | 0.000 | - | 0.000 |

# What is Wrong?

- Established benchmarks are mostly for marketing!
- Strict benchmarking conditions
  - Fixed workload & faultload + Small set of metrics
- Workload & faultload:
  - May not be representative of the user scenario
- Metrics:
  - Fixed! May not satisfy the user needs
  - Decision based on several metrics is difficult!

**No security benchmark endorsed by any organization or industry**

# CHALLENGES

- Satisfy industry requirements
  - Representativeness, portability, scalability, non-intrusiveness, low cost, …
  - Prevent “gaming”
- Satisfy user requirements
  - Representativeness, usefulness, simplicity of use…
  - Adaptable – allow “gaming”
- Endorsement by TPC, SPEC, …
  - **How to?**

# Is there a Future?

- Resilience Benchmarking
  - Assess and compare the behavior of components and computer systems when subjected to changes
  - Which resilience metrics?
    - Comparable, consistent, understandable, meaningful, …
  - Changeloads:
    - Representative, practical, portable, …
- Trustworthiness Benchmarking
  - What evidences to collect?
  - What metrics?
  - Dynamicity of perception… social trust...

# CONCLUSIONS

- The benchmarking concept is well established!
- Acceptance by “big” industry depends on perceived utility for marketing
- Acceptance by users requires “adaptability”
- From a research perspective, performance and dependability benchmarking are well known
- Security benchmarking approaches are weak
- New types of benchmarks will bring additional challenges!

# QUESTIONS?

**Marco Vieira**
Department of Informatics Engineering
University of Coimbra
mvieira@dei.uc.pt
http://eden.dei.uc.pt/~mvieira