A Historical and Statistical Study of the Software Vulnerability Landscape

Assane Gueye  
assaneg@andrew.cmu.edu  
Assistant Teaching Professor  
Carnegie Mellon University Africa  
Kigali, Rwanda

Peter Mell  
peter.mell@nist.gov  
Senior Computer Scientist  
National Institute of Standards and Technology  
Gaithersburg, MD-USA

The Seventh International Conference on Advances and Trends in Software Engineering  
SOFTENG 2021  
April 18, 2021 to April 22, 2021 - Porto, Portugal
Who Am I?

Assane Gueye joined Carnegie Mellon University Africa on August 1st, 2020. Prior to joining CMU Africa, he was a faculty member at the ICT Department at the University Alioune Diop of Bambe, Senegal, where he also leads the research group “Technologies de l’Information et de la Communication pour le Développement” (TIC4Dev). Gueye also holds a guest researcher position with the National Institute for Standards and Technology, Gaithersburg, Maryland, USA.

Assane completed his Ph.D. in electrical engineering and computer science from UC Berkeley in March 2011. He holds a Master’s degree in communication systems engineering from Ecole Polytechnique Fédérale de Lausanne, Switzerland.

His research focuses in two main areas: performance evaluation and security of large-scale communication systems, and information and communication technologies for development (ICT4D). Assane is a Fellow of the Next Einstein Forum (Class of 2016). In 2019 he was nominated as a member of the European Alliance for Innovation (EAI) inaugural Fellow Class.

Assane Gueye
ASSISTANT TEACHING PROFESSOR

Department(s): CMU-Africa
Email: assaneg@andrew.cmu.edu
CyLab-Africa
Enabling trust, equity, and financial opportunity through security and privacy research and education

For engagement opportunities contact:
Michael Lisanti
Director of Partnerships, CyLab
mlisanti@cmu.edu
Faith Rugema
Director of Partnerships, CMU-Africa
frugema@andrew.cmu.edu

https://www.africa.engineering.cmu.edu/research/cylab.html
Motivation

Understanding the landscape of software vulnerabilities is key for developing effective security solutions.

If the most significant of these types can be identified, developers of programming languages, software, and security tools can focus on preventing them to diminish the quantity and severity of newly discovered vulnerabilities.
### Approach (1)

**Common Vulnerabilities Scoring System (CVSS) Dataset**

<table>
<thead>
<tr>
<th>CVSS v3 Metrics</th>
<th>Metric Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attack Vector (AV)</td>
<td>Network (N), Adjacent (A), Local (L), Physical (P)</td>
</tr>
<tr>
<td>Attack Complexity (AC)</td>
<td>Low (L), High (H)</td>
</tr>
<tr>
<td>Privileges Required (PR)</td>
<td>None (N), Low (L), High (H)</td>
</tr>
<tr>
<td>User Interaction (UI)</td>
<td>None (N), Required (R)</td>
</tr>
<tr>
<td>Scope (S)</td>
<td>Unchanged (U), Changed (C)</td>
</tr>
<tr>
<td>Confidentiality (C)</td>
<td>High (H), Low (L), None (N)</td>
</tr>
<tr>
<td>Integrity (I)</td>
<td>High (H), Low (L), None (N)</td>
</tr>
<tr>
<td>Availability (A)</td>
<td>High (H), Low (L), None (N)</td>
</tr>
</tbody>
</table>

**CVSS:**

- **AV:** Network (N), Adjacent (A), Local (L), Physical (P)
- **AC:** Low (L), High (H)
- **PR:** None (N), Low (L), High (H)
- **UI:** None (N), Required (R)
- **S:** Unchanged (U), Changed (C)
- **C:** High (H), Low (L), None (N)
- **I:** High (H), Low (L), None (N)
- **A:** High (H), Low (L), None (N)

**CVSS:3.1:** `/AV:N/AC:L/PR:H/UI:N/S:U/C:L/I:L/A:N`
Approach (2)

• Experiments

  • Score (numerical) distributions

  • Metric values distributions

  • Relative rankings of the most frequent metric values

  • The most prevalent patterns of co-occurrence of the metric values
Results and Analysis (1)

### Score Distribution


**Some Insights:**

Predominance of certain vectors (groupings of vulnerability characteristics) in the real world!

---

**Producing Numerical Score**

<table>
<thead>
<tr>
<th>Metric</th>
<th>Metric Value</th>
<th>Numerical Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attack Vector / Modified Attack Vector</td>
<td>Network</td>
<td>0.85</td>
</tr>
<tr>
<td></td>
<td>Adjacent</td>
<td>0.62</td>
</tr>
<tr>
<td></td>
<td>Local</td>
<td>0.55</td>
</tr>
<tr>
<td></td>
<td>Physical</td>
<td>0.2</td>
</tr>
</tbody>
</table>

**ISS =**

\[ 1 - [(1 - Confidentiality) \times (1 - Integrity) \times (1 - Availability)] \]

- **Impact**
  - If Scope is Unchanged: \(6.42 \times \text{ISS}\)
  - If Scope is Changed: \(7.52 \times (\text{ISS} - 0.029) \times 3.25 \times (\text{ISS} - 0.029)^{1/3}\)

- **Exploitability**
  - \(8.22 \times \text{AttackVector} \times \text{AttackComplexity} \times \text{PrivilegesRequired} \times \text{UserInteraction}\)

- **BaseScore**
  - If Impact <= 0: 0, else
  - If Scope is Unchanged: Roundup(Minimum(Impact \times Exploitability), 10)
  - If Scope is Changed: Roundup(Minimum(1.08 \times (Impact \times Exploitability), 10))
Results and Analysis (2.1)

- Metric Values Distribution
Results and Analysis (2.2)

**Metric Values Distribution**

<table>
<thead>
<tr>
<th>Metric</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attack Vector (AV)</td>
<td>Mostly network (N), some local (L)</td>
</tr>
<tr>
<td>Attack Complexity (AC)</td>
<td>Low (L)</td>
</tr>
<tr>
<td>Privilege Required (PR)</td>
<td>Mostly none (N), sometime low (L)</td>
</tr>
<tr>
<td>User Interaction (UI)</td>
<td>Dominantly not required (N)</td>
</tr>
<tr>
<td>Scope (S)</td>
<td>Unchanged (U)</td>
</tr>
<tr>
<td>Confidentiality (C)</td>
<td>Dominated by high (H)</td>
</tr>
<tr>
<td>Integrity (I)</td>
<td>Dominated by high (H)</td>
</tr>
<tr>
<td>Availability (A)</td>
<td>Dominated by high (H)</td>
</tr>
</tbody>
</table>

**Some Insights:**

Some metrics values have dominated the landscape!
Results and Analysis (3)

- **Metrics Values Ranking (Top 10 over the years)**

  ![Top10 Association](image)

  The size of the circle is proportional to the number of times that metric value appeared in a score in that year.

  **Some Insights:**
  
  - Same top 10 values appeared from 2016 to 2019 (confirming domination by some values)
  
  - Metrics values ranked almost the same over the years
    
      - Top 2 are constant and in the same order over the time period
    
      - Top 4 and the bottom 4 (including the 11th appended value) are also constant
Associations

Some insights:

- Impact metrics (C:H), (I:H), and (A:H) form a clique. Whenever one of the metrics is highly impacted the others are also highly impacted.

- (S:C), (C:L) and (I:L) form a clique. When clique values are true:
  - AV is likely to be network (AV:N),
  - A is likely not impacted (A:N),
  - User interaction required (UI:R).

When (UI:R), no privileged (PR:N) is needed.

- When C is not impacted (C:N) or PR is low (PR:L) UI is likely not needed (UI:N)
Observations:

- Vulnerability landscape constantly dominated by a few vulnerability types
- Overwhelming majority of software vulnerabilities exploitable over the network
- Most vulnerabilities requiring no/low sophistication to be exploited
- No spill-over effect for attacks

Conclusion:

- As a community, we have not been successful fixing what seems to be the most prevalent software vulnerabilities
- Either:
  - We are incapable of fixing them
  - We are focusing on the wrong ones (i.e., our security metrics are flawed)
- In either case we need to “stop and think”: about the ways we are developing software and/or the methods we use to identify vulnerabilities
Thank you!

**Contact:** assaneg@andrew.cmu.edu