How Good is Openly Available Code Snippets Containing Software Vulnerabilities to Train Machine Learning Algorithms?

Kaan Oguzhan
Dr. rer. nat. Tiago Espinha Gasiba
Akram Louati

Siemens AG
Munich
Background

Machine Learning

Appearance of both keywords "Cybersecurity" and "Machine Learning" in Academic Papers according to Scopus

Background

Machine Learning

Data

Model is as good as your data

Quality & Quantity

Motivation

Goal: Developing software vulnerability detection in source code by means of ML Algorithms

Training sounds straightforward, but

“Model is as good as your data”
Motivation

Research questions:

- Where can we find code snippets to train ML models to detect software vulnerabilities?
- What is the quality of the code snippets which are openly available on the internet for training ML Models?
  - Can they be used to train ML models?

NOTE: in our work we use industry standard categories of software vulnerabilities
Overview

➔ Publicly available code snippets

Quality & Quantity

➔ Analysis measures
  i. Categories
    ○ Programming Language
    ○ OWASP TOP 10
    ○ PCI-DSS
    ○ CWE (Common Weakness Enumeration)
  ii. Fitness for ML

➔ Conclusion
Publicly Available Snippets per Source

- Source 1: https://samate.nist.gov/SRD/testsuites/juliet
- Source 2: https://wiki.sei.cmu.edu
- Source 3: https://vulncat.fortify.com/
- Source 4: https://github.com/returntocorp/semgrep-rules/
- Source 5: https://cwe.mitre.org/
- Source 6: https://www.sonarqube.org/
- Source 9: https://github.com/snoopysecurity/Vulnerable-Code-Snippets
- Source 10: https://github.com/conikee/seeve
- Source 11: https://github.com/fkie-cad/cwe_checker/tree/master/test/artificial_samples
Non-Compliant snippets per language
OWASP TOP 10 - Years

Number of OWASP TOP 10 snippets per year

- Year 2004: 1819
- Year 2007: 1528
- Year 2010: 1555
- Year 2013: 1701
- Year 2017: 2191
- Year 2021: 2336

Unrestricted
OWASP TOP 10 - Years & Categories

Number of OWASP TOP 10 snippets and their categories per year

Year & Category

Number of snippets
PCI-DSS Version

Number of PCI-DSS snippets per version (X)

Unrestricted
PCI-DSS Categories

Number of PCI-DSS snippets per version (X.X)

Unrestricted
PCI-DSS Sub-Categories

Number of PCI-DSS snippets per version (X.X-X)
Category Analysis Conclusion

➔ Uniformity of data on high level is not enough

➔ Neither snippet count for OWASP TOP 10 nor PCI-DSS is uniform on a sub category analysis

➔ Training on snippets for OWASP TOP 10 or PCI-DSS
  ➔ Results in heavily biased models towards some (sub)categories
## Juliet Dataset

### Table II

<table>
<thead>
<tr>
<th>ID</th>
<th>C Snippet count</th>
<th>ID</th>
<th>Java Snippet count</th>
<th>ID</th>
<th>C# Snippet count</th>
<th>ID</th>
<th>C++ Snippet count</th>
</tr>
</thead>
<tbody>
<tr>
<td>CWE 121</td>
<td>5906</td>
<td>CWE 190</td>
<td>6555</td>
<td>CWE 197</td>
<td>7695</td>
<td>CWE 762</td>
<td>5180</td>
</tr>
<tr>
<td>CWE 78</td>
<td>5600</td>
<td>CWE 191</td>
<td>5244</td>
<td>CWE 190</td>
<td>5643</td>
<td>CWE 122</td>
<td>4948</td>
</tr>
<tr>
<td>CWE 190</td>
<td>5040</td>
<td>CWE 129</td>
<td>4104</td>
<td>CWE 191</td>
<td>3762</td>
<td>CWE 36</td>
<td>3500</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>CWE 674</td>
<td>2</td>
<td>CWE 499</td>
<td>1</td>
<td>CWE 397</td>
<td>1</td>
<td>CWE 562</td>
<td>1</td>
</tr>
<tr>
<td>CWE 562</td>
<td>2</td>
<td>CWE 248</td>
<td>1</td>
<td>CWE 366</td>
<td>1</td>
<td>CWE 468</td>
<td>1</td>
</tr>
<tr>
<td>CWE 561</td>
<td>2</td>
<td>CWE 111</td>
<td>1</td>
<td>CWE 248</td>
<td>1</td>
<td>CWE 440</td>
<td>1</td>
</tr>
</tbody>
</table>
Juliet Dataset Analysis Conclusion

➔ Has huge number of snippet examples

➔ Very valuable resource for testing tools

➔ Not good for training Machine learning models
  ◆ Underlying snippet bias
Conclusion

➔ Where can code snippets be found?
  ◆ 11 possible sources of information
  ◆ Not all represented the same (most prominent: C, Java, C#, C++)

➔ What is the quality of the code snippets?
  ◆ Varies with the programming language
  ◆ Within a programming language → imbalance between vulnerability categories

Main conclusion:
  ◆ Some programming languages hugely underrepresented
  ◆ Juliet set - mostly synthetic data
  ◆ Not clear how good the snippets are to train ML models

Further work:
  ◆ Investigate “real-world” code snippets based on check-in comment
# Keyword Occurrences

<table>
<thead>
<tr>
<th>Top-5</th>
<th>2-gram</th>
</tr>
</thead>
<tbody>
<tr>
<td>(improper, neutralization)</td>
<td></td>
</tr>
<tr>
<td>(integer, overflow)</td>
<td></td>
</tr>
<tr>
<td>(buffer, overflow)</td>
<td></td>
</tr>
<tr>
<td>(special, elements)</td>
<td></td>
</tr>
<tr>
<td>(integer, underflow)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Top-5</th>
<th>3-gram</th>
</tr>
</thead>
<tbody>
<tr>
<td>(overflow, or, wraparound)</td>
<td></td>
</tr>
<tr>
<td>(neutralization, of, special)</td>
<td></td>
</tr>
<tr>
<td>(special, elements, used)</td>
<td></td>
</tr>
<tr>
<td>(integer, underflow, wrap)</td>
<td></td>
</tr>
<tr>
<td>(numeric, truncation, error)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Top-5</th>
<th>4-gram</th>
</tr>
</thead>
<tbody>
<tr>
<td>(integer, overflow, or, wraparound)</td>
<td></td>
</tr>
<tr>
<td>(neutralization, of, special, elements)</td>
<td></td>
</tr>
<tr>
<td>(improper, neutralization, of, special)</td>
<td></td>
</tr>
<tr>
<td>(command, os, command, injection)</td>
<td></td>
</tr>
<tr>
<td>(improper, validation, of, array)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Top-5</th>
<th>5-gram</th>
</tr>
</thead>
<tbody>
<tr>
<td>(improper, neutralization, of, special, elements)</td>
<td></td>
</tr>
<tr>
<td>(integer, underflow, wrap, or, wraparound)</td>
<td></td>
</tr>
<tr>
<td>(os, command, os, command, injection)</td>
<td></td>
</tr>
<tr>
<td>(improper, validation, of, array, index)</td>
<td></td>
</tr>
<tr>
<td>(use, of, externally-controlled, format, string)</td>
<td></td>
</tr>
</tbody>
</table>
Thank you for Listening

Kaan Oguzhan
kaan-oguzhan@siemens.com
Dr. rer. nat. Tiago Espinha Gasiba
tiago.gasiba@siemens.com
Akram Louati
akram.louati@siemens.com