Towards the Detection of Inconsistencies in Public Security Vulnerability Reports

Ying Dong, Wenbo Guo, Yueqi Chen, Xinyu Xing, Yuqing Zhang, Gang Wang

USENIX Security 2019
August 15th
Challenges Faced by Security Operations Engineers

1. Keep an eye on new vulnerabilities that affect their systems
2. Patch vulnerable softwares as soon as possible
Inconsistent Information → Confusion
A New Vulnerability (CVE-2018-0852) is Exposed

Microsoft outlook 2007 SP3 - NOT listed.
Research Problems

1. Is inconsistency issue prevalent?
2. What are the characteristics of inconsistent info?
3. Reasons for inconsistency?
4. Security implications of inconsistency?
Measuring Inconsistency of Vulnerability Reports

1999 - 2018

Over 20 years
Across websites
of 13 categories
In This Paper:

Part I: VIEM - an automatic system

extract vulnerable software name and versions

Part II: Large-scale Measurement

quantify inconsistency and interesting findings
Traditional NLP Tools Don’t Work Well (Validated)

1. Dictionary-based method (CNLL ’06, EMNLP ’13)
2. Pre-defined rules (SIGSOFT ’12, CCS ’17, FSE ’17)
3. Regular-expression based technique (CCS ’17, FSE ’17)
4. Techniques handling single entity (ISESE ’14, CCS ’17, FSE ’17)
5. Semfuzz (CCS ’17)

Reason: Unique characteristics of vulnerability reports
Why This Is Hard

1. Previously unseen vulnerable softwares (Ruby on Rails)
   -> Dictionary-based ✗

2. Both vulnerable (2.3.x) and non-vulnerable versions (3.0.0 and later) exist
   -> Pre-defined rules ✗

3. Reports are highly unstructured
   -> Regular-expression based ✗
4. **Multiple interested entities**
   - Existing tools handling single entity ✗

5. **Diverse vulnerability types**
   - Tools for certain vulnerability types (e.g., recall < 40%) ✗

---

**Why This Is Hard (cont.)**

In Windows Vista SP2 and Windows Server 2008 SP2, the Windows font library in .NET Framework 3.0 SP2, 3.5, 3.5.1, 4, 4.5, 4.5.1, 4.5.2, and 4.6; Skype for Business 2016; Lync 2010; Lync 2013 SP1; and Silverlight 5 allows remote attackers to execute arbitrary code via a crafted embedded font, aka "Graphics Memory Corruption Vulnerability."

Publish Date: 2015-12-09 Last Update Date: 2017-09-12
VIEM - NER/RE Model

"The Microsoft VBScript 5.7 and 5.8 engines, as used in Internet Explorer 9 through 11 …"

1. Bi-directional RNN
2. word/character embedding
3. Gazetteer

1. One-hot encoding
2. Hierarchical Attention-Network
VIEM - Transfer Learning

- Shorten training cycle
- Resolve inadequate training data of some vulnerability categories

NER/RE Model

Memory Corruption

NER/RE Model
SQL Injection

NER/RE Model
File Inclusion

NER/RE Model
...
VIEM - Dataset

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Vulnerability Reports</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>70,569</td>
</tr>
</tbody>
</table>

1. Over past 20 years (1999-2018)
VIEM - Evaluating NER/RE models

<table>
<thead>
<tr>
<th>Metric</th>
<th>Precision</th>
<th>Recall</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Result</td>
<td>0.9411</td>
<td>0.9932</td>
<td>0.9764</td>
</tr>
</tbody>
</table>

Over “Memory Corruption” Category

1. *G*-truth dataset (3,448 CVE IDs) with a ratio 8:1:1 for training, validation, and testing
2. Near 100% accuracy, the state-of-the-art is no higher than 90%
### VIEM - Evaluating Transfer Learning

Public vulnerability databases such as the Common Vulnerabilities and Exposures (CVE) and the National Vulnerability Database (NVD) have achieved great success in providing vulnerability disclosure and mitigation. While these databases have an important role in ensuring the security and reliability of software systems, there is a growing concern about their data quality and consistency. In this paper, we propose an automated system to detect and correct these inconsistencies in the CVE descriptions and the NVD dataset.

#### Abstract

Public vulnerability databases such as the Common Vulnerabilities and Exposures (CVE) and the National Vulnerability Database (NVD) have achieved great success in providing vulnerability disclosure and mitigation. While these databases have an important role in ensuring the security and reliability of software systems, there is a growing concern about their data quality and consistency. In this paper, we propose an automated system to detect and correct these inconsistencies in the CVE descriptions and the NVD dataset.

1. **Teacher Model** - “Memory Corruption” Category (3448 reports)
2. **Student Model** - other 12 categories (145 reports per category)
3. Solved inadequate training dataset issue, and improved accuracy
In This Paper,

**Part I**: VIEM - an automatic system

extract vulnerable software name and versions

**Part II**: Large-scale Measurement

quantify inconsistency and interesting findings
Metrics

1. **Match software names** - # of same words > # of different words

   “Internet Explorer” and “Microsoft Internet Explorer” ✓

1. **Measure version consistency** - Strict match vs. Loose match

   CPE directory from NIST

   “1.1” and “from 1.0 to 1.4” ----------------------> “[1.1]” and “[1.0, 1.1, 1.2, 1.3, 1.4]”

   Strict match (Exact match) ✗

   Loose match (One covers another) ✓
Inconsistency Exists Among All Vuln. Report Websites

Matching against NVD - official vulnerability report database maintained by U.S. government

Posted after NVD entries were created.
Inconsistency Exists For All Vulnerability Categories

More complex and requires longer time to reproduce and validate.

Matching rate for different vulnerability categories - (CVE + 5 websites) vs. NVD
Inconsistency: Overclaim vs. Underclaim

Compared against CVE, NVD overclaims/underclaims vulnerable versions
Overclaim/Underclaim Are Both Common

Percentage of Underclaim/Overclaim using loose match: (CVE + 5 websites) vs. NVD

NVD either suffers from delays to update or fails to keep track of the external information.
Inconsistency Rate Varies Over Time

Consistency rate over time: (CVE + 5 websites) vs. NVD

NVD are getting better at summarizing vulnerability versions.
Reasons of Inconsistency - 1

- Typos

<table>
<thead>
<tr>
<th>Software</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Videolan VLC media player</td>
<td>0.8.6</td>
</tr>
</tbody>
</table>

NVD data / CVE summary

<table>
<thead>
<tr>
<th>Software</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Videolan VLC media player</td>
<td>0.6.8</td>
</tr>
</tbody>
</table>

SecurityFocus

CVE-2010-0364
Reasons of Inconsistency - 2

- Most reports are seldom updated once created
  - 66.3% of the NVD entries have never been updated

![Diagram showing NVD and SecurityFocus KDPics 1.16 and 1.11 and 1.16 updated between 2006 and 2010 with CVE-2006-6516]
Security Implications - Case Study

- 7 real-world vulnerability, 47 reports, from 5 websites
- 3 security researchers, 185 versions, 4 months’ manual verification
- 64 versions are confirmed, 12 newly discovered vulnerable versions
<table>
<thead>
<tr>
<th>CVE ID</th>
<th>Ground truth</th>
<th>Intersection Of 5 Sites</th>
<th>Union Of 5 Sites</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVE-2008-2950</td>
<td>0.5.9 - 0.8.4 (16)</td>
<td>1.9.15 (1)</td>
<td>1.9.15 and possibly others (40)</td>
</tr>
<tr>
<td>poppler</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CVE-2009-5018</td>
<td>2.4.2 - 2.5.6 (13)</td>
<td>0.99 - 2.5.3 (36)</td>
<td>≤ 2.5.3 (36)</td>
</tr>
<tr>
<td>gif2png</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CVE-2015-7805</td>
<td>1.0.15 - 1.0.25 (11)</td>
<td>1.0.25 (1)</td>
<td>1.0.25 (1)</td>
</tr>
<tr>
<td>libsndfile</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CVE-2016-7445</td>
<td>1.5 - 2.1.1 (7)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>openjpeg</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CVE-2016-8676</td>
<td>11.0 - 11.8 (9)</td>
<td>≤ 11.8 (47)</td>
<td>11.3, 11.4, 11.5, 11.7 (4)</td>
</tr>
<tr>
<td>libav</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CVE-2016-9556</td>
<td>7.0.3.1 - 7.0.3.7 (7)</td>
<td>11.3, 11.4, 11.5, 11.7, 11.8, 11.9 (4)</td>
<td>1.0.25 (1)</td>
</tr>
<tr>
<td>ImageMagick</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Conclusion

1. VIEM - an automatic tool to detect inconsistency in Vul. reports
2. A large-scale measurement of the information consistency
3. Case study - validated inconsistent information (and show its impact)

Open Challenges

1. Standardize vulnerability reporting procedure
2. Design a fully automated system to verify the vulnerability reported
Thank you

Code & Data

https://github.com/pinkymm/inconsistency_detection

Presenter: Yueqi (Lewis) Chen

http://www.personal.psu.edu/yxc431/