Acquisitional Rule-based Engine for Discovering Internet-of-Things Devices

Xuan Feng, Qiang Li, Haining Wang, Limin Sun
Aug 15, 2018
Outline

- Background and Motivation
- Rule Miner (ARE)
- Design and Implementation
- Evaluation
- ARE-based Applications
- Conclusion
Internet-of-Things (IoT) Devices

- Various IoT devices connected to the Internet
  - cameras, routers, printers, TV set-top boxes,
  - industrial control systems and medical equipment.

- Estimated number – reported by Gartner
  - 5.5 million new IoT devices every day
  - 20 billion by 2020

- Meanwhile, these IoT devices also yield substantial security challenges
  - device vulnerabilities
  - mismanagement
  - misconfiguration
Security Concerns

• Mirai botnet: IoT devices being compromised and exploited as parts of a “botnet”, attacking critical national infrastructures
  – October, 2016
  – attacking the Dyn Services
  – causing Internet service disruptions across Europe and the United States

• Hackers Turn IoT devices (DVRs) Into Worst Bitcoin Miners

Map of areas most affected by Mirai attack
Security Concerns

• Mirai botnet: IoT devices being compromised and exploited as parts of a “botnet”, attacking critical national infrastructures
  – October, 2016
  – attacking the Dyn Services
  – causing Internet service disruptions across Europe and the United States

• Hackers turn compromised IoT devices (DVRs) into worst Bitcoin miners
Annotating IoT Devices

- There are two basic approaches to addressing security threats:
  - reactive defense
  - proactive prevention
    - more efficient than the reactive defense against large-scale security incidents

- To protect IoT devices in a proactive manner
  - a *prerequisite* step: discovering, cataloging, and annotating IoT devices.
Device Annotation

• The device annotation contains:
  – IoT device type (e.g., routers/camera),
  – vendor (e.g., Sony, CISCO),
  – product model (e.g., TV-IP302P).

• Fingerprinting-based Discovery.
  – high demand for training data and a large number of device models

• Banner-grabbing Discovery
  – examples: Nmap and Ztag
  – a manual fashion with technical knowledge
  – impossible for large-scale annotations
  – hard to keep the discovery updated

Regular expression used in Nmap

Rules used in Ztag (Censys)
Key Observation

- Manufacturers usually hardcode the correlated information into IoT devices to distinguish their brands.
  - TL-WR740/TL-WR741ND in HTML file

- There are many websites describing device products such as product reviews.
  - Amazon and NEWEGG websites provide the device annotation descriptions.

- Our work is rule-based.
  - the automatic rule generation is mainly based on the relationship between the application data of IoT devices and the corresponding description websites.
Technical Challenges

• Two major challenges:
  – the application data is hardcoded by its manufacturer.
  – there are massive device annotations in the market.

• Notably, manufacturers would release new products and abandon outdated products.
  – manually enumerating every description webpage is impossible.
Rule Miner

- **Transaction set**
  - application-layer data and the relevant webpages

- **Device entity recognition (DER)**
  - contexter and local dependency

- **Apriori algorithm**
  - learn the relationship form Transactions
Transaction

• Transaction definition:
  – a transaction is a pair of textual units, consisting of the application-layer data of an IoT device and the corresponding description of the IoT device from a webpage.

• A rule is \( \{A \Rightarrow B\} \).
  • the association between a few features (A) extracted from the application-layer data and the device annotation (B) extracted from relevant webpages
Device Entity Recognition (DER)

- DER is a combination of the corpus-based and rule-based.
  - corpus-based: device types and vendor names.
  - rule-based: use regular expressions to extract the product name entity.

<table>
<thead>
<tr>
<th>Entity</th>
<th>Context terms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device Type</td>
<td>nvr, nv, video server, video encoder, video recorder</td>
</tr>
<tr>
<td></td>
<td>diskstation, rackstation, printer, copier, scanner</td>
</tr>
<tr>
<td></td>
<td>switches, modem, switch, gateway, access point</td>
</tr>
<tr>
<td>Vendor</td>
<td>1,552 vendor names</td>
</tr>
<tr>
<td>Product</td>
<td>[A-Za-z]+[-]?[A-Za-z]!*[0-9]+[0-9]?[-]?[A-Za-z-0-9]</td>
</tr>
</tbody>
</table>
Device Entity Recognition (DER)

• Poor performance :
  – high false positives in terms of device type and product name.
  – an irrelevant webpage may include keyword of device type such as “switch”.
  – a phrase that meets the requirement of regex for a product name.

• True IoT entities always have strong dependence upon one another.
  – (1) the vendor entity first appears, followed by the device-type entity, and finally the product entity;
  – (2) the vendor entity first appears, and the product entity appears second without any other object between the vendor entity, and the device-type entity follows

The local dependency of the device entity
## Rule Generation

- **Apriori algorithm**
  
  \[
  sup(A) = \left| \sum_{i=1}^{n} A \in t_i \right| / |T| 
  \]

  \[
  conf(A \Rightarrow B) = sup(A \cup B) / sup(A) 
  \]

- **Parameters**
  
  - *support* is used to indicate the frequency of the variable (A) appearance
  
  - *confidence* is the frequency of the rules (A \Rightarrow B) under the condition in which the A appears
  
  - \( sup(A) = 0.1\% \) and \( conf(A \Rightarrow B) = 50\% \) work well.

### Illustrating Rules

| “Panasonic”, “KX-HGW500-1.51” | {IPCam, Panasonic, KX-HGW500} |
| “TL-WR1043ND”, “Wireless”, “Gigabit” | {Router, TP-Link, WR1043N} |
| “00a9”, “Webserver”, “Welcome”, “ZyXEL” | {Router, Zyxel, P-600HN} |
| “P-660HN-51”, “micro_httpd” | {Gateway, Juniper, SRX210} |
| “Juniper”, “Web” | “SRX210HE”, “00a9” |
| “Brother”, “HL-3170CDW” | “Device”, “Manager” |
| “seriesHL-3170CDW”, “seriesPlease”, “debut/1.20” | {Printer, Brother, HL-3170} |

A few example rules learned for IoT devices.
Design and Implementation

• Transaction collection
  – response data collection.
  – web crawler.
• Rule miner
• Rule library
  – store each rule \( \{A \Rightarrow B\} \)
• Planner.
  – update the rule library

Acquisitional Rule-based Engine (ARE) architecture for learning device rules.
Real-world Evaluation

• Data sets
  – First dataset:
    • randomly choose 350 IoT devices from the Internet.
    • 4 different device types (NVR, NVS, router, and IP camera) 64 different vendors, and 314 different products
  – Second dataset:
    • 6.9 million IoT devices that our application collects on the Internet.
    • randomly sample 50 IoT devices iteratively for 20 times.
    • 1,000 devices across 10 device types and 77 vendors.
Real-world Evaluation

• Number of rules
  – generate 115,979 rules in one week.
  – in comparison with 6,514 from Nmap
  – 92.8% of rules - (device type, vendor, product).
  – 7.2% of rules just label device type and vendor.
  – about 30% of rules in Nmap with a fine-grained annotation.

• Precision of rules
  – first dataset: 95.7%
  – second dataset: 97.5%

• Coverage of rules
  – 94.9% coverage
  – given the same number of response packets, ARE achieves a larger coverage than Nmap

<table>
<thead>
<tr>
<th>Category</th>
<th>Num</th>
<th>Percentage %</th>
</tr>
</thead>
<tbody>
<tr>
<td>(device type, vendor, product)</td>
<td>107,627</td>
<td>92.8</td>
</tr>
<tr>
<td>(device type, vendor, null)</td>
<td>8,352</td>
<td>7.2</td>
</tr>
</tbody>
</table>

Rules generated by ARE.

<table>
<thead>
<tr>
<th></th>
<th>Precision</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>The first dataset</td>
<td>95.7%</td>
<td>94.9%</td>
</tr>
<tr>
<td>The second dataset</td>
<td>97.5%</td>
<td>—</td>
</tr>
</tbody>
</table>

Precision and coverage of rules on the dataset.
Real-world Evaluation

• Dynamic rule learning
  – the number of rules is increasing as ARE learns with the increase of network space.

• Overhead of ARE
  – Windows 10, 4vCPU, 16GB of memory, 64-bit OS
  – time cost of ARE for automatic rule generation is low in practice
ARE-based Applications

- Internet-wide measurement for IoT devices.
- Detecting compromised IoT devices.
- Detecting underlying vulnerable IoT devices.
Internet-wide Device Measurement

• Three application-layer datasets from Censys
  – HTTP, FTP, and Telnet.
• Deploying our collection module on the Amazon EC2
  • RTSP application-layer data.
• Using ARE, found 6.9 million IoT devices
  – 3.9M HTTP, 1.5M FTP, 1M Telnet, and 0.5 M RTSP.
• Discovery:
  – a large number of visible and reachable IoT devices on the Internet
  – the long-tail distribution is common for IoT devices (31% in Top 10)
  – many devices should not be visible or reachable from the external networks (camera/DVR).
Compromised Device Detection

• Deploy honeypots as vantage points for monitoring traffic on the Internet.
• Annotating the captured IP addresses
  – a normal IoT device should never access honeypots.
  – an IoT device accesses our honeypots due to misconfigured or compromised.
• Honeypots
  – 4 countries, 7 cities
  – the duration is two months
• Discovery:
  – 50 compromised IoT devices every day.
  – In total, 2,000 compromised IoT devices among (12,928 IP addresses)
  – Device type: DVR, NAS and router
  – Also, some smart TV boxes exhibit malicious behaviors.
Vulnerable Device Analysis

• Finding underlying vulnerable devices
  – cross match the exposed IoT devices with the vulnerability information from NVD

• Discovery:
  – a large number of underlying vulnerable devices in the cyberspace
  – most vulnerabilities is about improper implementation
    • Path Traversal, Credentials Management, and Improper Access Control
    • Could be easily avoided if a developer pays more attention to security.

<table>
<thead>
<tr>
<th>CWE ID</th>
<th>Weakness Summary</th>
<th>Number of IoT devices</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Information Disclosure</td>
<td>573,656</td>
</tr>
<tr>
<td>22</td>
<td>Path Traversal</td>
<td>363,894</td>
</tr>
<tr>
<td>352</td>
<td>CSRF</td>
<td>348,031</td>
</tr>
<tr>
<td>264</td>
<td>Permission, Privileges, Access Control</td>
<td>345,175</td>
</tr>
<tr>
<td>255</td>
<td>Credentials Management</td>
<td>342,215</td>
</tr>
<tr>
<td>79</td>
<td>Cross-site Scripting</td>
<td>331,649</td>
</tr>
<tr>
<td>119</td>
<td>Buffer Overflow</td>
<td>149,984</td>
</tr>
<tr>
<td>399</td>
<td>Resource Management Errors</td>
<td>93,292</td>
</tr>
<tr>
<td>284</td>
<td>Improper Access Control</td>
<td>69,229</td>
</tr>
<tr>
<td>77</td>
<td>Command Injection</td>
<td>64727</td>
</tr>
</tbody>
</table>

Top 10 CWE of online IoT devices
Conclusion

• We propose the framework of ARE
  – automatically generate rules for IoT device recognition without human effort and training data.

• We implement a prototype of ARE and evaluate its effectiveness.
  – ARE generates a much larger number of rules within one week and achieves much more fine-grained IoT device discovery than existing tools.

• We apply ARE for three different IoT device discovery scenarios. Our main findings include
  – (1) a large number of IoT devices are accessible on the Internet
  – (2) thousands of overlooked IoT devices are compromised
  – (3) hundreds of thousands of IoT devices have underlying security vulnerabilities and are exposed to the public.
Thank you!  Q&A