Crawler classification using ant-based clustering scheme

O. Nasuji
ta
Shu Ishikura, Takehisa Yagi, Daiki Chiba,
Masayuki Murata

• Graduate school of information science and technology, Osaka university, Japan
• NTT secure platform laboratories, Japan

Web-based attacks

- The rapid expansion of web services
- More and more attacks targeting web servers that provide web services
  - E.g.:
    - Linux worm (2013/11-)
    - Apache Struts (2013/4-)
    - Shellshock (2014/10-)

- We need to collect and analyze information on web-based attacks in order to detect unknown attacks:
  - It is difficult to detect all vulnerabilities in web servers due to the rapid growth in diversity of web services
  - Detecting attacks using known vulnerabilities is insufficient for preventing all web-based attacks

Collecting attacks by Honeypots

- Web honeypots
  - Systems that collect and monitor web attacks targeting web servers deployed in accordance with types of attacks
  - Low interaction and high interaction
    - Low interaction honeypots
    - Emulate vulnerable OSs and applications
    - Have difficulty in responding to all types of attacks
  - High interaction honeypots
    - Accommodate actual OS applications
    - Collect a variety of attacks since they can actually be under attacks

- We need to identify malicious accesses from a number of accesses:
  - Honeypots receive not only malicious accesses but also normal accesses such as crawler accesses by search engines
  - Detecting vulnerability scanning is important for attack prevention
  - Accesses by crawlers are much similar to vulnerability scanning

Diversifying web services

- Conventional scheme for detecting attacks [2]
  - Identifies crawler accesses and then assumes the others to be malicious accesses
  - In crawler identification, accesses that are similar to those by well-known crawlers (e.g. Google) are identified as crawler accesses

- Diversifying web services:
  - Not only malicious accesses but also normal accesses become diverse
  - Adapting to diverse accesses is a challenging task

AntTree [5]

- A clustering scheme inspired by the behavior exhibited by ants in which they form chains with each other to construct a tree structure
  - A datum assumes a mobile agent called "ant"
  - Data (ants) chains with each other to construct tree structure

The construction of tree by ants

1. At first, all ants exist in the root of the tree (support)
2. Ants start to move away from the support one by one
3. A moving ant explore the tree for discovering an ant (a node) that is similar to itself
4. Arriving at a similar node, a moving ant becomes a descendant of the node and stops moving
5. The next ant starts to move away from the support

Design of ants (data):

- A set of ants (data): \( \{a_1, a_2, \ldots, a_n\} \)
- Each datum corresponds to an ant
- An ant determines its behavior with information about the current node \( \pi_m \) and its neighbors
- The maximum number of descendant nodes of a node: \( l \)

- Ants explore nodes that are similar to themselves
  - The similarity between ant \( a_i \) and \( a_j \): \( \text{Sim}(a_i, a_j) \)
  - The similarity/dissimilarity threshold of ant \( a_i \): \( T_{\text{Sim}}(a_i) \)
    - If \( \text{Sim}(a_i, a_j) \geq T_{\text{Sim}}(a_i) \), \( a_j \) assumes that \( a_i \) is similar to itself
    - If \( \text{Sim}(a_i, a_j) < T_{\text{Sim}}(a_i) \), \( a_j \) assumes that \( a_i \) is dissimilar itself
  - Ant \( a_j \) updates these threshold while exploring the tree
    - At first, \( T_{\text{Sim}}(a_j) = l \), \( T_{\text{DisSim}}(a_j) = 0 \)

- Bio-inspired schemes are advantageous for classifying a lot of data and for detecting unknown malicious threats
- Natural organisms behave individually and autonomously using only local information and as a result, a global pattern or behavior emerges


Construction of the tree

- The first ant becomes a descendant of the support
  - At first, the tree consists of only the support
- Following ants behave one by one in accordance with local information

Algorithm for moving away from the support (1/3)

1. When ant \(a_i\) starts to move away from the support
   - Ant \(a_i\) compares itself to descendant nodes of the support
     - If there are nodes that are similar to ant \(a_i\) among descendant nodes

Algorithm for moving away from the support (2/3)

1. When ant \(a_i\) starts to move away from the support
   - Ant \(a_i\) compares itself to descendant nodes of the support
     - If the support already has descendant nodes, ant \(a_i\) moves to the most similar node

Algorithm for moving away from node \(a^{POS}\) (1/3)

2. When ant \(a_i\) arrives at node \(a^{POS}\)
   - If node \(a^{POS}\) is similar to ant \(a_i\)
     - If all neighbors of node \(a^{POS}\) are dissimilar to ant \(a_i\)

Algorithm for moving away from node \(a^{POS}\) (2/3)

2. When ant \(a_i\) arrives at node \(a^{POS}\)
   - If node \(a^{POS}\) is similar to ant \(a_i\)
     - If there are neighbor nodes that are not dissimilar to ant \(a_i\)
Algorithm for moving away from node \( a^{\text{POS}} \) (3/3)

2. When ant \( a_i \) arrives at node \( a^{\text{POS}} \),
3. If node \( a^{\text{POS}} \) is not similar to ant \( a_i \),
   - Ant \( a_i \) moves to a neighbor randomly.

Evaluation

- We evaluated crawler classification by AntTree (an unsupervised learning).
  - Compared to
    - Random Forest (a supervised learning) is used for learning.
  - Data
    - Metrics
      - Recall: the fraction of data that are correctly classified within data to which the same label is attached.
      - Precision: the fraction of data that are correctly classified within data classified to the same category.

Data set

- HTTP communication logs collected by honeypots
  - Each log is attached a label as following:
    - Google communication logs: accesses by Google.
    - Crawler communication logs: accesses by crawlers other than Google.
    - Non-crawler communication logs: others.
  - The test data set for evaluation of our proposal (AntTree) and the conventional scheme:
    - 3,004,508 communication logs including 1,562,254 Crawler logs and 1,502,254 Non-crawler logs.
  - The learning data set for the conventional scheme:
    - 3,004,508 communication logs including 1,562,254 Google logs and 1,502,254 Non-crawler logs.

Feature vector

- We identify accesses by crawlers with HTTP communication logs
  - Request packets
    - Information on HTTP request packets that the honeypot received
      - Request information: request URL, communication method (GET, POST)
      - Packet header: UserAgent, referer, source/destination port number, communication protocol (HTTP, HTTPS)
    - Packet body: body length
    - Responses to request packets
      - Information on responses of honeypots to request packets
        - Response type: Status code (200, 404, etc.)
        - Response information: text types (HTML, CSS, etc.) and character encodings (UTF-8, ISO-8859-1, etc.) included in response packets

Result

- AntTree can classify crawler logs more precisely compared to the conventional scheme:
  - Due to the diversifying of communication services, features of crawlers are not always similar to those of Google crawler.
  - AntTree does not need the learning data set for classification.

Conventional scheme

<table>
<thead>
<tr>
<th>Label</th>
<th>Prediction</th>
<th>Crawler</th>
<th>Non-Crawler</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crawler</td>
<td>1,205,857</td>
<td>240,379</td>
<td>82.64%</td>
<td></td>
</tr>
<tr>
<td>Non-Crawler</td>
<td>1,056,362</td>
<td>1,562,254</td>
<td>82.55%</td>
<td></td>
</tr>
<tr>
<td>Precision</td>
<td>92.54%</td>
<td>84.26%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

AntTree

<table>
<thead>
<tr>
<th>Label</th>
<th>Prediction</th>
<th>Crawler</th>
<th>Non-Crawler</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crawler</td>
<td>1,258,357</td>
<td>242,271</td>
<td>83.87%</td>
<td></td>
</tr>
<tr>
<td>Non-Crawler</td>
<td>76,417</td>
<td>1,425,387</td>
<td>94.91%</td>
<td></td>
</tr>
<tr>
<td>Precision</td>
<td>94.28%</td>
<td>83.48%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Characteristic of AntTree

• AntTree can classify clusters accurately whose size is small
  • In AntTree, each datum explores similar kinds of data using only local information while moving over the tree

• AntTree can classify data whose features are minor in the entire data set although those minorities of features make us to overlook them

Example of crawler classification by AntTree
The test data set includes 50 Crawler logs and 50 Non-crawler logs

Conclusion

• Conclusion
  • We introduce an ant-based clustering scheme to crawler classification
  • We evaluate our proposal using data collected in a real network
  • Our proposal can identify accesses by crawlers more precisely than the conventional scheme
  • AntTree can classify data whose features are minor in the entire data set

• Future work
  • We will evaluate AntTree by considering the changes in communication features
  • We will use statistical features for the classification of communication logs
  • Statistical information of communication logs would be important
    • e.g.) The intervals and the distribution of packet arrivals

Statistical information of communication logs would be important
  • e.g.) The intervals and the distribution of packet arrivals