Crawler classification using ant-based clustering scheme

○Naomi Kuze[1], Shu Ishikura[1], Takeshi Yagi[2], Daiki Chiba[2], Masayuki Murata[1]

[1] Graduate school of information science and technology, Osaka university, Japan

[2] 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 (2014/4-)
    - 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 [1]
  - 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

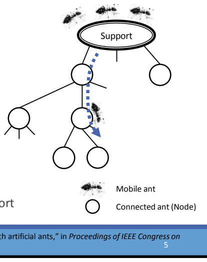
- **We adopt a bio-inspired clustering scheme for the crawler classification**
  - Bio-inspired schemes are advantageous **for classifying a lot of data** and **for detecting unknown malicious threats**
    - Natural organisms behave individually and autonomously using only local information and as a result, a global pattern or behavior emerges

[1] J. P. John, F. Yu, Y. Xie, A. Krishnamurthy, and M. Abadi, "Heat-seeking honeypots: Design and experience," in *Proceedings of the 20th International Conf. on World Wide Web*, Mar. 2011, pp. 207-216.

AntTree [5]

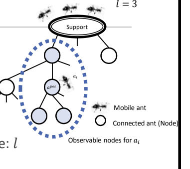
- A clustering scheme inspired by the behavior exhibited by ants in which they form chains with each other to construct a tree structure
  - A datum assumes a mobile agent called "ant"
  - Data (ants) chains with each other to construct tree structure

- 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
     1. A moving ant explore the tree for discovering an ant (a node) that is similar to itself
     2. Arriving at a similar node, a moving ant becomes a descendant of the node and stops moving
     3. The next ant starts to move away from the support



[5] H. Azzag, N. Monmarche, M. Slimane, and G. Venturini, "AntTree: a new model for clustering with artificial ants," in *Proceedings of IEEE Congress on Evolutionary Computation (CEC2003)*, vol. 4, Dec. 2003, pp. 2642-2647.

Design of ants

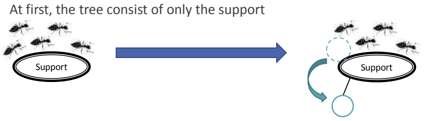


- A set of ants (data):
$$\{a_1, a_2, \ldots, a_N\}$$
  - Each datum corresponds to an ant
  - An ant determines its behavior with information about the current node a^{pos} and its neighbors
  - The maximum number of descendant nodes of a node: l

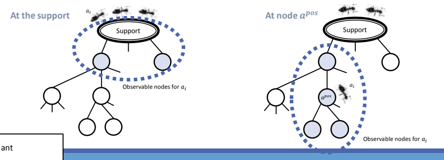
- Ants explore nodes that are similar to themselves
  - The similarity between ant a_i and a_j: $Sim(a_i, a_j)$
  - The similarity/dissimilarity threshold of ant a_i:
$$T_{Sim}(a_i), T_{Dissim}(a_i)$$
    - If $\boldsymbol{Sim(a_i, a_j) \geq T_{Sim}(a_i)}$, a_i assumes that **a_j is similar** to itself
    - If $\boldsymbol{Sim(a_i, a_j) < T_{Dissim}(a_i)}$, a_i assumes that **a_j is dissimilar** itself
    - Ant a_i updates these threshold while exploring the tree
      - At first, $T_{Sim}(a_i) = 1,\ T_{Dissim}(a_i) = 0$

Construction of the tree

- The first ant becomes a descendant of the support
  - At first, the tree consist of only the support



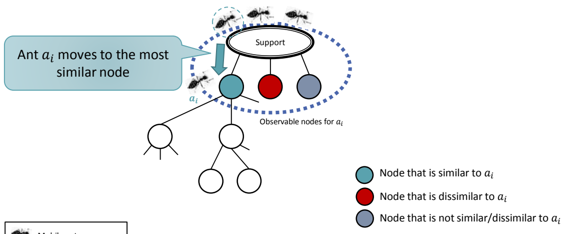
- Following ants behave one by one in accordance with local information



Mobile ant
Connected ant (Node)

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
   a. If there are nodes that are similar to ant a^{pos} among descendant nodes

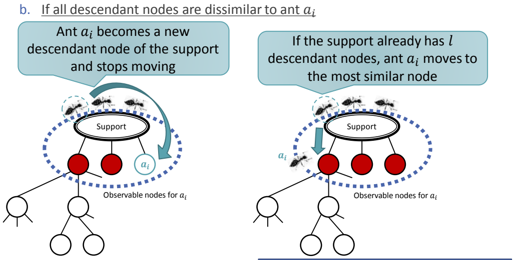


Mobile ant
Connected ant (Node)

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

Similar node
Dissimilar node
Other node

1. When ant a_i starts to move away from the support
   - Ant a_i compares itself to descendant nodes of the support
   b. If all descendant nodes are dissimilar to ant a_i



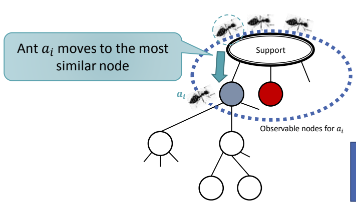
Update of the threshold $T_{Sim}(a_i) \leftarrow T_{Sim}(a_i) \times \alpha_1$

Mobile ant
Connected ant (Node)

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

Similar node
Dissimilar node
Other node

1. When ant a_i starts to move away from the support
   - Ant a_i compares itself to descendant nodes of the support
   c. If both a. and b. are satisfied



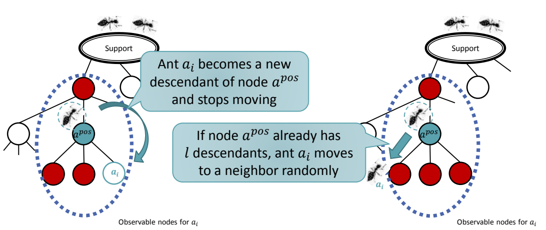
Updates of thresholds
$T_{Sim}(a_i) \leftarrow T_{Sim}(a_i) \times \alpha_1$
$T_{Dissim}(a_i) \leftarrow T_{Dissim} + \alpha_2$

Mobile ant
Connected ant (Node)

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

Similar node
Dissimilar node
Other node

2. When ant a_i arrives at node a^{pos}
   a. If node a^{pos} is similar to ant a_i
      i. If all neighbors of node a^{pos} are dissimilar to ant a_i

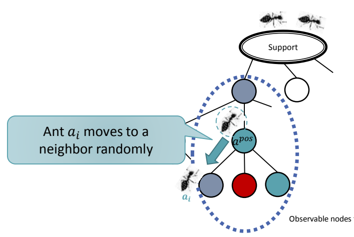


Mobile ant
Connected ant (Node)

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

Similar node
Dissimilar node
Other node

2. When ant a_i arrives at node a^{pos}
   a. If node a^{pos} is similar to ant a_i
      ii. If there are neighbor nodes that are not dissimilar to ant a_i

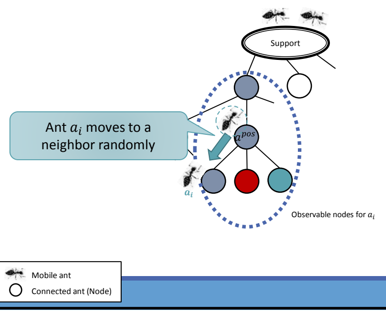


Updates of thresholds
$T_{Sim}(a_i) \leftarrow T_{Sim}(a_i) \times \alpha_1$
$T_{Dissim}(a_i) \leftarrow T_{Dissim} + \alpha_2$

Mobile ant
Connected ant (Node)

Algorithm for moving away from node a^{pos} (3/3)

2. When ant a_i arrives at node a^{pos}
   b. If node a^{pos} is not similar to ant a_i

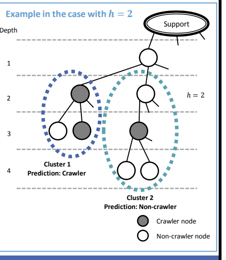


Application of AntTree to crawler classification

- Similarity $Sim(a_i, a_j)$ between ant a_i and a_j
  - Ant a_i (a datum) has M features $\{v_{i_1}, \dots, v_{i_M}\}$

$$Sim(a_i, a_j) = 1 - \underbrace{\sqrt{\frac{1}{M}\sum_{k=1}^{M}\left(v_{i_k} - v_{j_k}\right)^2}}_{\text{The Euclidean distance between ant } a_i \text{ and } a_j \text{ in the feature vector space}}$$

- Cluster interpretation
  - A cluster corresponds to a subtree whose root is an h depth node of the tree
  - A cluster is classified according to which type of data is a majority in the cluster



Evaluation

- We evaluated crawler classification by AntTree (an unsupervised learning)
  - Compared to
    - The conventional scheme [1] using accesses by well-known crawlers for identifying accessed by other crawlers
      - Random Forest (a supervised learning) is used for learning
  - Data
    - HTTP communication logs collected by 37 web honeypots [14] from 2013/8/29 to 2014/1/14
  - Metrix
    - Recall: the fraction of data that are correctly classified within data to which the same label is attacked
    - Precision: : the fraction of data that are correctly classified within data classified to the same category

Recall= $\frac{|L_A \cap C_A|}{|L_A|}$, Precision= $\frac{|L_A \cap C_A|}{|C_A|}$

L_A: A set of data to which label A is attached
C_A: A set of data which are classified to category A

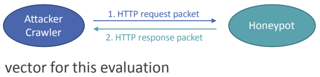
[1] J. P. John, F. Yu, Y. Xie, A. Krishnamurthy, and M. Abadi, "Heat-seeking honeypots: Design and experience," in *Proc. of the 20th International Conf. on World Wide Web*, Mar. 2011, pp. 207-216.
[14] T. Yagi, N. Tanimoto, and T. Hariu, "Intelligent high-interaction web honeypots based on url conversion scheme," *IEICE transactions on communications*, vol. 94, no. 5, pp. 1339-1347, May 2011.

Data set

- HTTP communication logs collected by honeypots
  - Each log is attached a label as following
    - ***Google***: communication logs of accesses by Google
      - Google logs are easy to identify with public information of Google (UserAgents and source IP addresses)
    - ***Crawler***: communication logs of accesses by crawlers other than Google
      - Crawler logs are classified manually by researchers and engineers
    - ***Non-crawler***: communication logs of with others
      - Non-crawler logs includes malicious logs
- **The test data set** for evaluation of our proposal (AntTree) and the conventional scheme
  - 3,004,508 communication logs including 1,502,254 Crawler logs and 1,502,254 Non-crawler logs
- **The learning data set** for the conventional scheme
  - 3,004,508 communication logs including 1,502,254 Google logs and 1,502,254 Non-crawler logs

Feature vector

- We identify accesses by crawlers with HTTP communication logs



- Feature vector for this evaluation
  - Request packets
    - Information on HTTP request packets that the honeypot received
      - Request information: request URL, communication method (GET, POST)
      - Packet header: UserAgent, referrer, 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: StatusCode (200, 404, etc.)
      - Response information: text types (HTML, CSS, etc.) and character encodings (UTF-8, ISO-8859-1, etc.) included in response packets

Result

Parameter settings
The maximum number of nodes l: 5, The depth of the root of each cluster h: 3
Parameters for updates of the similarity/dissimilarity threshold (α_1, α_2): (0.95,0.2)

- 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

		Prediction: Crawler	Prediction: Non-Crawler	Recall
Label	Crawler	1,241,437	260,817	82.64%
	Non-Crawler	105,952	1,396,302	92.95%
	Precision	92.14%	84.26%	

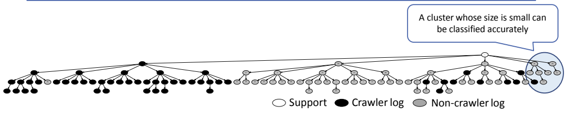
AntTree

		Prediction: Crawler	Prediction: Non-Crawler	Recall
Label	Crawler	1,259,976	242,278	83.87%
	Non-Crawler	76,417	1,425,837	94.91%
	Precision	94.28%	85.48%	

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 these 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