Detection of Vulnerability Scanning Using Features of Collective Accesses Based on Information Collected from Multiple Honeypots

Naomi Kuze*, Shu Ishikura*, Takeshi Yagi†, Daiki Chiba† and Masayuki Murata*
*Graduate School of Information Science and Technology, Osaka University, Japan
Email: {n-kuze, s-ishikura, murata}@ist.osaka-u.ac.jp
†NTT Secure Platform Laboratories, Japan
Email: {yagi.takeshi, chiba.daiki}@lab.ntt.co.jp

***Abstract*—Attacks against websites are increasing rapidly with the expansion of web services. An increasing number of diversified web services make it difficult to prevent such attacks due to many known vulnerabilities in websites. To overcome this problem, it is necessary to collect the most recent attacks using decoy web honeypots and to implement countermeasures against malicious threats. Web honeypots collect not only malicious accesses by attackers but also benign accesses such as those by web search crawlers. Thus, it is essential to develop a means of automatically identifying malicious accesses from mixed collected data including both malicious and benign accesses. Specifically, detecting vulnerability scanning, which is a preliminary process, is important for preventing attacks. In this study, we focused on classification of accesses for web crawling and vulnerability scanning since these accesses are too similar to be identified. We propose a feature vector including features of collective accesses, e.g., intervals of request arrivals and the dispersion of source port numbers, obtained with multiple honeypots deployed in different networks for classification. Through evaluation using data collected from 37 honeypots in a real network, we show that features of collective accesses are advantageous for vulnerability scanning and crawler classification.**

Keywords-Intrusion detection, web-based attacks, classification, features of collective accesses

I. Introduction

Prevention of web-based attacks is a challenging and essential task for realizing secure network systems. Not only web-based attacks on many unspecified individuals but also cyber-terrorism targeting specific governments, corporations, and other large organizations are increasing, which is becoming a serious problem. However, it is difficult to detect all vulnerabilities in web servers, which can be targets of web-based attacks, due to the rapid growth in the diversity of web services. In other words, detecting attacks using known vulnerabilities is insufficient for preventing all web-based attacks. Therefore, we must collect and analyze information on web-based attacks to detect unknown attacks.

To collect web-based attack information, decoy systems called *web honeypots*, which collect and monitor web attacks targeting web servers, are deployed [1], [2]. There are two types of web honeypots, *low interaction* and *high interaction* [3]. Low-interaction honeypots emulate vulnerable OSs and applications, whereas high-interaction honeypots accommodate actual OS applications. High-interaction web honeypots can actually be under attack; therefore, are used to collect and analyze a variety of web-based attacks [1], [2]. This is the reason why we focus on high-interaction web honeypots. The honeypots referred to in this paper are high-interaction web honeypots.

Honeypots monitor not only malicious accesses but also normal accesses such as crawler accesses by search engines. Therefore, we need to identify malicious accesses from a large number of collected accesses. Researchers and engineers usually manually identify malicious accesses in most cases, but this is becoming difficult due to the rapid increase in traffic. This leads to the necessity of a method for automatically identifying malicious accesses. To prevent web-based attacks, it is important to detect vulnerability scanning, which involves attempting to access web servers automatically for identifying and attacking web servers with vulnerable OSs and applications. A related study proposed a method of detecting web attacks that involves first identifying accesses by crawlers then assuming the others to be malicious accesses [1]. However, accesses for vulnerability scanning are similar to those by crawlers that attempt to systematically browse web pages for web service indexing. As a result, information obtained from a communication log consisting of a request to a web server and a response to the request is insufficient for classifying vulnerability scanning and crawlers.

Despite the similarity, vulnerability scanning and crawlers have different objectives. On one hand, crawlers browse web pages periodically and continually for web service indexing. On the other hand, in vulnerability scanning, malicious hosts attempt to accesses several web servers in a bursty manner for identifying vulnerable OSs and applications. Most of these accesses for vulnerability scanning randomly select source ports to conceal the existence of malicious hosts. Moreover, since URLs requested by these accesses are randomly generated, whether these URLs are available or not, most of these requests are not accepted by web servers. Therefore, we can identify crawlers and vulnerability scanning by using the features of collective accesses by the same hosts. In this study, we used features of collective accesses obtained from information collected from multiple honeypots deployed in different networks for the classification of vulnerability scanning and crawlers.

Specifically, we first collect communication logs obtained with multiple honeypots deployed in different networks and grouped communication logs sent from the same Autonomous System (AS) in the same group. We then analyzed features of each group such as intervals and the burstiness of accesses and the dispersion of source port numbers to use them for vulnerability scanning and crawler classification.

For this study, we designed and proposed a feature vector including not only features of individual accesses but also features of collective accesses obtained from information collected from multiple web servers for identifying vulnerability scanning and crawlers and show advantages of it through evaluation. We evaluated classification with the proposed feature vector using data collected in a real network and argue that classification with features of collective accesses is more precise than that without them.

The remainder of this paper is as follows. We first introduce related work in Section II. We then explain features of individual accesses obtained with a single web server in Section III and discuss features of collective accesses obtained with multiple web servers in Section IV. In Section V, we discuss our evaluation of crawler and vulnerability scanning classification with the proposed feature vector using data collected from a real network. Finally, in Section VI, we conclude this paper and mention future work.

II. Related Work

There are two types of methods for detecting web attacks: *signature detection*, which uses signatures from known malicious threats, and *anomaly detection*, which detects unknown malicious threats by collecting and analyzing web attacks. The rapid expansion in the number of web applications makes it difficult to detect all vulnerabilities in applications. Therefore, it is necessary to collect and analyze accesses to detect unknown attacks. One method achieves that purpose by capturing traffic on user PCs and servers [4], [5], and another method collects web attacks by using honeypots [6]. The former method enables us to precisely detect malicious accesses, but there are problems in implementing this type of method. For example, capturing traffic on user PCs invade the users' privacy and reduces service quality. Therefore, the latter method is more appropriate, and we focus on it in this paper.

As mentioned in Section I, it is important to classify accesses by crawlers for identifying malicious accesses from data collected from honeypots. Since Google opens information of the source IP addresses and the UserAgents of its crawlers, accesses by Google crawlers are easy to identify. Google crawlers try various inputs to many types of programs. More exactly, they send requests `wget -r -np -l 0 http://example.com/<directory_name>/` to the host `example.com`. On the contrary, malicious hosts try various inputs to vulnerable programs. In vulnerability scanning, in particular, malicious hosts usually input benign values to these vulnerable programs. Consequently, accesses for vulnerability scanning and those by crawlers are too similar to be classified.

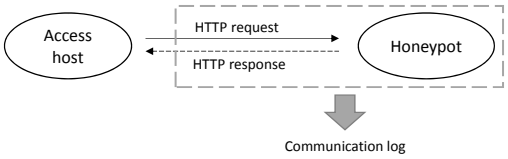

Figure 1. HTTP communication log

III. Features of individual accesses

For identifying accesses for web crawling and vulnerability scanning, we used HTTP communication logs collected from honeypots. An HTTP communication log includes information of an HTTP request received by a honeypot and an HTTP response to it by the honeypot (Fig. 1). In this section, we explain features of individual communication logs, which can be obtained with a single honeypot.

A. Overview

In this study, we used the following features of requests for classification.

- Request information: request URL, HTTP method (GET, POST, etc.)
- HTTP header: UserAgent, referer, source/destination port number, communication protocol (HTTP or HTTPS)
- HTTP body: body length

On the contrary, we used the following features of HTTP responses to HTTP requests.

- Response type: StatusCode (200, 404, etc.) defined in RFC 2616
- Response information: text type (HTML, CSS, etc.) and character encoding (UTF-8, ISO-8859-1, etc.) of contents in HTTP responses

These features are easy to obtain from information collected from a single honeypot but are insufficient for precisely classifying vulnerability scanning and crawlers. Therefore, in Section IV, we discuss features of collective accesses obtained from information collected from multiple honeypots deployed in different networks.

B. Example of Features

We explain how to use the request URL below for classification as an example.

`http://www.example.com/test/index.php?id=1`

For classification, We use the path part `/test/index.php` and the query part `id=1` of the request URL. The path part gives the location (path) of the requested resource, and the query part specifies parameters. We deeply explain features of the path and query parts below.

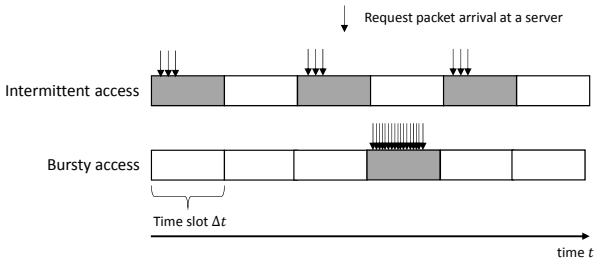

Figure 2. Intermittent accesses and bursty accesses

We use the number of characters of the path and query parts as features. The path part includes 12 characters (`test`, `index`, and `php`), and the query part includes 3 characters (`id` and 1). Note that delimiters such as "/" and "." are excluded from the character count. We also use the number of levels in the path part and the number of parameters in the query part as features. The path part is constructed with 2 levels (`/test` and `/index.php`). The query part includes 1 parameter (`id=1`). We then calculate the average number of characters of each level in the path part and the average number of characters of each parameter in the query part, and use them as features. The average number of characters of each level in the path is 6 which is calculated by dividing the number of characters of the path part by the number of levels. Similarly, the average number of characters of each parameter in the query part is 3 calculated by dividing the number of characters of the query part by the number of parameters.

Moreover, we focus on types of characters of the path and query parts. We describe the path and query parts of the request URL in the form of a regular expression `<the string type (string, integer, hex, or base64); the string length>` [7] for converting text information to numerical information. Specifically, we convert the path and query parts into `/<string; 4>/<string; 5>.<string; 3>` and `<string; 2>=<int; 1>`, respectively. We then use the types of character strings (string, integer, hex, and base64) and the ratios of strings of each type in the regular expression as features.

IV. Features of collective accesses

In this section, we discuss features of collective accesses, which are obtained from information collected with multiple honeypots, of the proposed feature vector for classification of vulnerability scanning and crawlers.

First, we collect communication logs obtained with multiple honeypots deployed in different networks and group a set of communication logs of accesses from the same AS in the same group. The source ASs can be identified using the source IP addresses of the requests. We then analyze features of each group.

For classifying vulnerability scanning and crawlers, we focus on intervals of requests, destination IP addresses, source port numbers, and StatusCodes to requests. We explain the details of each feature below.

A. Request arrivals

A set of communication logs of accesses by the same AS is sorted by the arrival time of requests in honeypots, and then is separated by time slot Δt. Accesses by crawlers that attempt to browse web servers periodically and accesses for vulnerability scanning that attempt to visit vulnerable web servers in a bursty manner are different from each other in forms of request arrivals, as shown in Fig. 2. Therefore, we obtain intervals and burstiness of request arrivals from the number of requests in each time slot and the number of time slots in which requests arrive at honeypots and use them as features. Since bursty accesses for vulnerability scanning by a malicious host continue only for several minutes, we set Δt to 300 sec in this study.

B. Destination IP addresses

Our analysis shows that malicious hosts attempt to access web servers continually in order of destination IP addresses for vulnerability scanning. Therefore, we analyze and use the continuity of destination IP addresses of accesses by the same host as features.

C. Source port numbers

Requests for vulnerability scanning usually have random source port numbers to conceal the existence of malicious hosts. Therefore, we use the continuity of source port numbers of accesses by the same host as a feature.

D. Responses from web servers

Malicious hosts usually attempt to request URLs generated randomly whether the URLs are available or not. This is because many of accesses for vulnerability scanning fail. Therefore, we obtain the error rate of accesses by the same host from StatusCodes and use it as a feature.

V. Evaluation and Discussion

A. Overview

We evaluated the classification of vulnerability scanning and crawlers with features of collective accesses using data collected from multiple honeypots in a real network. As mentioned in Section II, accesses by Google crawlers are easy to identify since Google opens the source IP addresses and UserAgents of its crawlers. Therefore, we first identified accesses by Google crawlers in collected communication logs then classified communication logs of accesses by crawlers other than Google and those by non-crawlers. Next, we classified accesses for vulnerability scanning in communication logs classified to those of accesses by non-crawlers. In Subsections V-C and V-B, we explain in detail the feature vector and the data set used in this evaluation.

Table I
CONNECTION LOGS COLLECTED FROM HONEYPOTS

Label		Number
Google		$8,276,246$
Crawler		$1,502,254$
Non-crawler	Scan	$4,830,615$
	Non-scan	$6,717,124$
Other		$710,708$
Total		$22,036,947$

In this evaluation, we evaluated the crawler and vulnerability scanning classification with features both of individual and collective accesses explained in Sections III and IV, respectively, compared to the case only with features of individual accesses. In this evaluation, we adopted RandomForest [8] as the learning algorithm for constructing classification models of vulnerability scanning and crawlers. The RandomForest achieves accurate classification and needs only a short calculation time even with a large number of features. Note that we used the RandomForest scheme provided by RapidMiner 5.

B. Feature Vector

We designed a feature vector and use it for crawler and vulnerability scanning classification. In this evaluation of vulnerability scanning classification, we compared classification with features of individual and collective accesses explained in Sections III and IV to that with features of individual accesses explained in Section III.

All features explained in Sections III and IV are normalized and described by real values in $[0, 1]$ for equalizing the weights of features. Feature v_{i_j} of data i ($i \in [1, \cdots, N]$, $j \in [1, \cdots, M]$) is normalized by

$$\frac{v_{i_j} - \min_{n \in [1,\cdots,N]} v_{n_j}}{\max_{n \in [1,\cdots,N]} v_{n_j} - \min_{n \in [1,\cdots,N]} v_{n_j}}.$$

In this evaluation, we used 126 features of individual accesses (89 features of HTTP requests and 37 features of HTTP responses) and 21 features of collective accesses.

C. Data Set

We used HTTP communication logs collected in a real network for our evaluation. These logs were collected from 37 honeypots [9] from August 29, 2013 to January 14, 2014. As mentioned in Section III, an HTTP communication log includes information of a request received by a honeypot and a response to it by the honeypot. We attached the following labels to about 220 million logs collected from honeypots.

- *Google* (about 8.27 million): Communication logs of accesses by Google crawler are labeled as "Google." Google logs are easy to be identified because information about source IP addresses and UserAgent of Google crawlers are open to the public.
- *Crawler* (about 1.50 million): Communication logs of accesses by crawlers other than Google are labeled as "Crawler." Crawler logs are identified in accordance with source IP addresses and UserAgents that we detect by manually analyzing communication logs. Some examples of the detected crawlers include Baidu, Bing, and Microsoft.
- *Non-crawler* (about 11.5 million): Communication logs of accesses by others are labeled as "Non-crawler." Non-crawler logs include malicious logs.
- *Other* (about 0.710 million): Other logs correspond to communication logs that are not categorized as either of the aforementioned labels. Most Other logs lack information for analysis. Therefore, we did not use Other logs in the evaluation.

Non-crawler logs have the following additional labels attached.

- *Scan* (about 4.83 million): Communication logs of accesses for vulnerability scanning are labeled as "Scan." We assume that hosts accessing 37 honeypots in an indiscriminate manner attempt to scan vulnerability of web servers. Specifically, source hosts of Scan accesses satisfy the following conditions.
  - Accesses by a host attempt to access honeypots more than 5 times.
  - More than 50% of accesses by the host fail due to errors (e.g., requested URLs are not available).
  - The ratio of continual accesses to the same web server to all accesses by the host is less than 20%.
- *Non-scan* (about 6.71 million): Communication logs of other accesses are labeled as "Non-scan."

Table I gives details of communication logs collected from 37 honeypots.

We used $3,004,508$ communication logs including $1,502,254$ Crawler logs and $1,502,254$ Non-crawler logs as the data set for this evaluation. The $1,502,254$ Non-crawler logs consist of $751,127$ Scan logs and $751,127$ Non-scan logs. Note that we sampled and used only $751,127$ Scan logs and $751,127$ Non-scan logs out of all of the Non-crawler logs. To evaluate the classification accuracy regardless of the access distribution over time, we sampled Scan and Non-scan logs randomly.

With the $3,004,508$ communication logs, we evaluate crawler and vulnerability scanning classification as follows.

1) We classify Crawler and Non-crawler logs using the data set, which includes $1,502,254$ Crawler logs and $1,502,254$ Non-crawler logs, mentioned above with features of individual accesses in Section III, or with features of individual and collective accesses explained in Sections III and IV.
2) We classify Scan and Non-scan logs using logs classified as Non-crawler with features of individual accesses or with features of individual and collective accesses as the test data set. For this evaluation of scan detection, we use features of individual HTTP requests (Section III), features of individual HTTP

Table II
CRAWLER DETECTION WITH FEATURES OF INDIVIDUAL ACCESSES

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

Table III
CRAWLER DETECTION WITH FEATURES OF COLLECTIVE ACCESSES

		Prediction		Recall
		Crawler	Non-crawler	
Label	Crawler	$1,456,596$	$45,658$	96.96%
	Non-crawler	$61,701$	$1,440,553$	95.89%
Precision		95.94%	96.93%	

requests and responses (Section III), and features of individual HTTP requests/responses and collective accesses (Sections III and IV).

D. Results and Discussion

We now discuss our evaluation in which we compared crawler and vulnerability scanning classification performance. In this evaluation, we used recall and precision as evaluation metrics. The definitions of these metrics are as follows.

- Recall: the fraction of data correctly classified within data to which the same label is attached. Given set $\mathcal{L}_A$ of data with label A and set $\mathcal{C}_A$ of data classified to A, recall of A is calculated as
$$\text{Recall} = \frac{|\mathcal{L}_A \bigcap \mathcal{C}_A|}{|\mathcal{L}_A|}.$$
- Precision: the fraction of data correctly classified within data classified as the same category. Given $\mathcal{L}_A$ and $\mathcal{C}_A$, the precision of A is calculated as
$$\text{Precision} = \frac{|\mathcal{L}_A \bigcap \mathcal{C}_A|}{|\mathcal{C}_A|}.$$

Note that *label* corresponds to the label attached to each log, as explained in Subsection V-B.

1) Crawler Detection: We first evaluated crawler classification using the test data set including $3,004,508$ communication logs explained in Subsection V-C. The results of the crawler classification with features of individual accesses and with features of individual and collective accesses are listed in Tables II and III, respectively. In these tables, *label* corresponds to the label attached to each log, as explained in Subsection V-C, and *precision* corresponds to the result of classification with features of individual accesses or with features of individual and collective accesses.

The precisions and recalls of Crawler and Non-crawler logs were higher when using features both of individual accesses and collective accesses than when using only features of individual accesses. This indicates that Crawler logs can be classified more precisely with features of collective accesses than without them. This is because it is difficult to identify Crawler and Scan logs only with features of individual accesses due to the similarity of them.

Moreover, we found that several features of collective features, such as the number of request arrivals and continuity of source port numbers and destination IP addresses, make a better contribution than most other features. Consequently, features of collective accesses obtained from information collected from multiple honeypots is advantageous for crawler classification.

2) Scan Detection: We next evaluated scan detection. We used $1,657,119$ communication logs classified as Non-crawler only with features of individual accesses and $1,486,211$ logs classified as Non-crawler with features of individual accesses and collective accesses in Subsection V-D1 as test data sets for the evaluation of scan detection.

Tables IV$\sim$VI list the results of scan detection using $1,657,119$ logs classified as Non-crawler with features of individual accesses (Table II) as the test data set. Similarly, Tables VII$\sim$IX list the results of scan detection using $1,486,211$ logs classified as Non-crawler with features of individual accesses and collective accesses (Table III) as the test data set. We evaluated scan detection with features of individual HTTP requests, with features of individual HTTP requests and responses, and with features of individual HTTP requests/responses and collective accesses for each test data set.

First, only with request information, the accuracy of the scan detection was low regardless of the test data set, as shown in Tables IV and VII. This indicates that request information is insufficient for classifying Scan logs. Moreover, the results show that many Scan logs were incorrectly classified as Non-scan logs. In fact, $171,442$ and $187,169$ Scan logs were classified as Non Scan logs among $1,657,119$ logs classified as Non-crawler with features of individual accesses and $1,486,211$ logs classified as Non-crawler with features of individual accesses and collective accesses, respectively, as shown in Tables IV and VII. This is because Scan logs and Crawler logs that are similar to each other were classified as Non-scan logs together. From the above results, request information is not sufficient for identifying Crawler logs and Scan logs.

Next, we discuss scan detection with request and response information. Using communication logs classified as Non-crawler with features of individual accesses as the test data set, the precisions and recalls of Scan and Non-scan logs with request and response information were as low as those only with request information, as shown in Tables IV and V. On the contrary, using communication logs classified as Non-crawler with features of individual and collective accesses as the test data set, Scan and Non-scan logs were classified more accurately with request and response information than with only request information. This is because communication logs classified as Non-crawler with features of individual and collective accesses included less Crawler

Table IV
SCAN DETECTION WITH FEATURES OF INDIVIDUAL REQUESTS WITH TEST DATA SET OBTAINED WITH FEATURES OF INDIVIDUAL ACCESSES

		Prediction		Recall
		Scan	Non-scan	
Label	Scan	547,393	171,442	76.15%
	Non-scan	3,798	747,270	99.49%
Precision		99.31%	81.34%	

Table V
SCAN DETECTION WITH FEATURES OF INDIVIDUAL REQUESTS/RESPONSES WITH TEST DATA SET OBTAINED WITH FEATURES OF INDIVIDUAL ACCESSES

		Prediction		Recall
		Scan	Non-scan	
Label	Scan	470,604	248,231	65.47%
	Non-scan	2,241	748,627	99.67%
Precision		99.48%	75.10%	

Table VI
SCAN DETECTION WITH FEATURES OF INDIVIDUAL REQUESTS/RESPONSE AND COLLECTIVE ACCESSES WITH TEST DATA SET OBTAINED WITH FEATURES OF INDIVIDUAL ACCESSES

		Prediction		Recall
		Scan	Non-scan	
Label	Scan	717,595	1,240	99.83%
	Non-scan	10,996	740,072	98.54%
Precision		98.49%	99.83%	

Table VII
SCAN DETECTION WITH FEATURES OF INDIVIDUAL REQUESTS WITH TEST DATA SET OBTAINED WITH FEATURES OF INDIVIDUAL AND COLLECTIVE ACCESSES

		Prediction		Recall
		Scan	Non-scan	
Label	Scan	563,958	187,169	75.08%
	Non-scan	4,079	747,048	99.46%
Precision		99.28%	79.97%	

Table VIII
SCAN DETECTION WITH FEATURES OF INDIVIDUAL REQUESTS/RESPONSES WITH TEST DATA SET OBTAINED WITH FEATURES OF INDIVIDUAL AND COLLECTIVE ACCESSES

		Prediction		Recall
		Scan	Non-scan	
Label	Scan	626,188	124,939	83.37%
	Non-scan	6,116	745,011	99.19%
Precision		99.03%	85.64%	

Table IX
SCAN DETECTION WITH FEATURES OF INDIVIDUAL REQUESTS/RESPONSES AND COLLECTIVE ACCESSES WITH TEST DATA SET OBTAINED WITH FEATURES OF INDIVIDUAL AND COLLECTIVE ACCESSES

		Prediction		Recall
		Scan	Non-scan	
Label	Scan	741,678	10,328	98.74%
	Non-scan	9,449	740,799	98.62%
Precision		98.63%	98.74%	

logs than communication logs classified as Non-crawler with features of individual accesses, as shown in Tables II and III. Consequently, request and response information is advantageous for identifying Scan and Non-scan logs, but is insufficient for identifying Crawler and Scan logs.

Finally, as shown in Tables VI and IX, with features of individual requests/responses and collective accesses, Scan logs were classified with high accuracy even when communication logs classified as Non-crawler with features of individual accesses, which included many Crawler logs, are used as the data set. This indicates that features of collective accesses are advantageous for the classification of vulnerability scanning and crawlers.

In conclusion, vulnerability scanning and crawlers are classified with high accuracy with features of collective accesses obtained from information collected from multiple honeypots, whereas it is difficult to classify them precisely only with request and response information obtained from information collected with a single honeypot.

VI. CONCLUSION AND FUTURE WORK

Due to the rapid growth in the diversity of web services, it is becoming increasingly difficult to classify a large number of communication logs using only known features that are detected manually. In particular, detecting vulnerability scanning is important for preventing web-based attacks since vulnerability scanning is a preliminary process. For vulnerability scanning detection, we proposed a feature vector that includes features of collective accesses obtained from information collected from multiple honeypots deployed in different networks. Through our evaluation using data collected from honeypots in a real network, we proved that accesses for vulnerability scanning are classified more precisely with features of collective accesses than without them.

For future work, we will adopt a clustering scheme inspired by biological behavior for adapting dynamically changing access features.

REFERENCES

[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 Conference on World Wide Web*, Mar. 2011, pp. 207–216.

[2] D. Canali and D. Balzarotti, "Behind the scenes of online attacks: an analysis of exploitation behaviors on the web," in *Proceedings of 20th Annual Network & Distributed System Security Symposium (NDSS 2013)*, Feb. 2013.

[3] N. Provos and T. Holz, *Virtual honeypots: from botnet tracking to intrusion detection*. Addison-Wesley Professional, Jul. 2007.

[4] C. Kruegel, G. Vigna, and W. Robertson, "A multi-model approach to the detection of web-based attacks," *Computer Networks*, vol. 48, no. 5, pp. 717–738, Feb. 2005.

[5] T. H. Kim, K. Kim, J. Kim, and S. J. Hong, "Profile-based web application security system with positive model selection," in *Proceedings of the 2nd Joint Workshop on Information Security*, Aug. 2007.

[6] T. Yagi, N. Tanimoto, and T. Hariu, "Design of provider-provisioned website protection scheme against malware distribution," *IEICE transactions on communications*, vol. 93, no. 5, pp. 1122–1130, May 2010.

[7] T. Nelms, R. Perdisci, and M. Ahamad, "ExecScent: Mining for new C&C domains in live networks with adaptive control protocol templates." in *Proceedings of 22nd USENIX Security Symposium*, Aug. 2013, pp. 589–604.

[8] A. Liaw and M. Wiener, "Classification and regression by randomforest," *R news*, vol. 2, no. 3, pp. 18–22, Dec. 2002.

[9] 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.