LSTM-based Detection for Timing Attacks in Named Data Network

Lin Yao*[||], Binyao Jiang[†], Jing Deng[‡], and Mohammad S. Obaidat, Fellow of IEEE[§]
*International School of Information Science & Technology, Dalian University of Technology, Dalian China 116620
[†]School of Software, Dalian University of Technology, Dalian China 116620
[‡]Department of Computer Science, University of North Carolina at Greensboro, Greensboro USA
[§]College of Computing and Informatics, University fo Sharjah, UAE, Nazarbayev University, Astana, Kazakhstan, KASIT, University of Jordan, Amman, Jordan and University of Science and Technology Beijing, China
[||]Email:yaolin@dlut.edu.cn

***Abstract*—Named Data Network (NDN) is an alternative to host-centric networking exemplified by today's Internet. One key feature of NDN is in-network caching that reduces access delay and query overhead by caching popular contents at the source as well as at a few other nodes. Unfortunately, in-network caching suffers various privacy risks by different attacks, one of which is termed timing attack. This is an attack to infer whether a consumer has recently requested certain contents based on the time difference between the delivery time of those contents that are currently cached and those that are not cached. In order to prevent the privacy leakage and resist such kind of attacks, we propose a detection scheme by adopting Long Short-term Memory (LSTM) model. Based on the four input features of LSTM, cache hit ratio, average request interval, request frequency, and types of requested contents, we timely capture more important eigenvalues by dividing a constant time window size into a few small slices in order to detect timing attacks accurately. We have performed extensive simulations to compare our scheme with several other state-of-the-art schemes in classification accuracy, detection ratio, false alarm ratio, and F-measure. It has been shown that our scheme possesses a better performance in all cases studied.**

***Index Terms*—Timing Attacks, Long Short-term Memory(LSTM), Named Data Network**

I. Introduction

Internet is full of different contents. Its ever-increasing popularity and saturation into every corner of our society leads to more interests into the contents available within the Internet. As a result, the requested contents, instead of the actual carriers, become the focus of attention [1] [2] [3]. Therefore, there have been quite some studies on next generation Internet architecture in recent years [4]. Named Data Network (NDN), as a promising information-centric network architecture, requires each consumer to request the content based on its name without caring about its addresses. A requesting node broadcasts an interest packet including the content's name. The interest packet will be forwarded through the network toward the source node. If the requested content is found at an intermediate node, the content will be returned to the requesting node. Each forwarding node or router decides whether to cache the content it sees passing by. Each router in NDN contains three crucial data structures shown in Fig. 1, CS (content storage), PIT (Pending Interest Table), and FIB (Forwarding Information Base) [5]. CS is used for caching and retrieving data. PIT records the currently unsatisfied interest packets and a set of corresponding incoming interfaces. Meanwhile, FIB routes the interest packets according to their name prefixes.

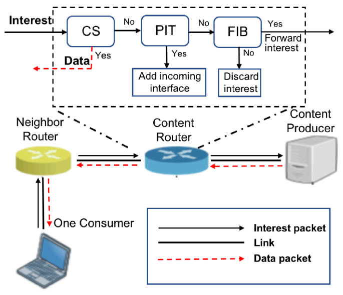


Fig. 1. Data Process in NDN

Though NDN routers can reduce overall latency and increase bandwidth utilization by caching popular contents, content caching can easily lead to privacy attacks. One of such attacks is called timing attack [6] whose aim is to infer whether some contents have been requested recently. Obviously, timing attacks expose users' partial content request history. With further background knowledge on the victims, Adversary can even infer their political inclination, favorited websites, etc. [7]. For example, if one victim is known to be the only person in the network neighborhood to speak a foreign language, frequent requests of contents in such a language can be used to pinpoint the victim's interest.

A common solution to protect users' privacy against timing attacks is to add some random delay to the response time regardless of whether an attack has occurred, since it is difficult to distinguish between a malicious request and a normal request [8]. In some other detection schemes, timing attacks are determined usually by learning attacker's traffic patterns in a fixed time window. However, the inherent features to determine timing attacks obtained from a single time window are often insufficient. In order to cope with this challenge, we divide a time window into a number of time slices to capture the detailed features and further use a machine learning technique in Recurrent Neural Network (RNN) called Long Short-Term Memory (LSTM) to distinguish attacks from normal requests. The proposed scheme, called Detect Timing Attacks with LSTM (DTAL), has a low computation cost and is robust against different attacks. The main contributions of our work are listed as follows:

(1) To the best of our knowledge, we are the first to determine timing attacks based on the inherent features of the abnormal traffic in multiple time slices. Compared with current detection schemes, more features are extracted within an equal time slice. Therefore, more important eigenvalues can be captured, leading to more accurate attack detections.
(2) We are the first to use a deep learning model to detect timing attacks based on the cache hit ratio, average request interval, request frequency, and number of types of requested contents, with nonlinear classification and higher classification accuracy.
(3) We evaluate the performance through extensive simulations. Compared with other baseline schemes [9] [7] [10], our DTAL has a superior performance with high classification accuracy, high detection ratio, low false alarm ratio, and high F-measure.

The rest of this paper is organized as follows: In Section 2, we review some related work about our study. We present our scheme in Section 3. The proposed scheme is evaluated and compared with related work in Section 4. Section 5 concludes our work.

II. Related Work

Since NDN is a new and promising architecture for the future Internet, it is limited in the number of related studies on detecting timing attacks. In this section, we only review some work related to our scheme. The related schemes are classified into ones without detection and with detection. Approaches without detection aim to prevent timing attacks even though no detetions are made. Approaches with detection start to defend the attack once attacks are detected.

Approaches without detection: Mohaisen et al. [8] proposed a user driven countermeasure scheme allowing users to mark which content was sensitive. Only when the sensitive content was requested for the first time, the router returned the content normally and the subsequent requests were responded with a random delay. Similarly, three types of cached contents were defined in [11]. It is considered unnecessary to protect non-sensitive contents. A random delay is added to sensitive contents. For the third type of contents, the router decided randomly if the content should be cached. Wu et al. [12] used random forwarding and network coding to improve the cache performance of the network, while also protecting the cache privacy. In order to mitigate timing attacks, Abani et al. [13] proposed a centrality-based caching strategy, which can cache the low-popular content at the node with high centrality.

Approaches with detection: Han et al. [14] proposed a scheme for detecting cache snooping in NDN based on the request frequency. Dogruluk et al. [7] proposed an attack detection scheme based on interest packets and cache ratios, and then used random caching strategies to mitigate timing attacks. Costa et al. [9] analyzed the features of timing attacks and proposed a detection scheme based on the values of RTT and cache hit ratio. In [15], timing attacks are determined based on the times that the same content is requested within a period of time. Some delays will be added once attacks are detected. In [16], the traffic feature is used to determine the attack. SVM model is adopted to distinguish the abnormal traffic from attackers [10].

Summary To mitigate the negative impacts of timing attacks, most of current schemes vary content response time, which have an unfortunately downside of additional delays and longer response time. Other detection schemes are based on traffic features only in a single time window. In order to obtain more detailed traffic features and to identify attacks more accurately, we propose to use a deep learning model to keep track of interest requests based on the cache hit ratio, average request interval, request frequency, and number of types of requested contents in multiple time slices within the same time window.

III. Detection Mechanism Based on LSTM

In this section, we first give an overview of our scheme, and then describe the details of DTAL.

A. Overview

To launch timing attacks successfully, attackers must send legal requests and determine whether the corresponding contents have been cached in the victim routers by analyzing the corresponding RTT. However, it is so difficult to detect timing attacks successfully due to the behavior similarity between the attacker and the normal consumer. To cope with this challenge, we propose our DTAL, a LSTM-based timing attack detection scheme shown in Fig. 2. To detect the abnormal behaviors, we collect several statistics of all interest packets: requested content names, interface where each interest came from, the arrival time of each interest packet, and the corresponding cache hit in each time window, to generate the four inherent features. In order to extract more detailed features from the traffic, we divide a fixed time window into multiple smaller slices. The vectors including the four features of all slices are regarded as the LSTM inputs sequentially. Based on the output of normal interest traffic or abnormal traffic, we can determine whether a router is under timing attacks.

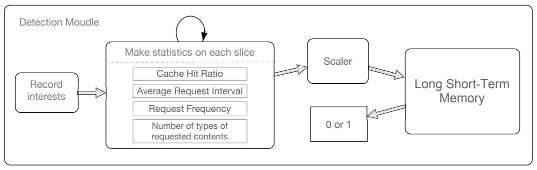


Fig. 2. Illustration of the DTAL Scheme

B. Detection of Timing Attacks

In this section, we first define the four inherent features. Then, we will present the detection algorithm specifically. The frequently used notations in this section are listed in Table I.

TABLE I
FREQUENT NOTATIONS

Notation	*Description*
t_i	The i-th time slice
F_n	The n-th interface
$\psi(t_i, F_n)$	The array of interests from the n-th interface in the i-th time slice
$\psi(t_i, F_n)[j]$	The j-th interest in $\psi(t_i, F_n)$
$\zeta(\psi(t_i, F_n)[j])$	1 means there is a cache hit. 0 means there is no hit
$h(t_i, F_n)$	The cache hit ratio
$\xi(\psi(t_i, F_n)[j])$	The arrival time of the j-th interest
$t(t_i, F_n)$	The average request interval
$\gamma(\psi(t_i, F_n))$	The set of content names in $\psi(t_i, F_n)$
$\|\psi(t_i, F_n)\|$	The array size
$f(t_i, F_n)$	The request frequency
$c(t_i, F_n)$	The number of types of requested contents

Definition 1 (Cache Hit Ratio(CHR)): Cache hit ratio is defined as the ratio of the number of cache hits to the total number of receiving interests [7] [9] [14]. Timing attacks aim to infer whether the contents have been cached. Frequent attacks will cause a higher hit ratio in the victimized router. The formula is defined as follows:

$$h(t_i, F_n) = \frac{\sum_{j=1}^{|\psi(t_i,F_n)|} \zeta(\psi(t_i, F_n)[j])}{|\psi(t_i, F_n)|}, \tag{1}$$

where $\psi(t_i, F_n)$ represents an array to store the interests from the n-th interface in the i-th time slice, and $|(\psi(t_i, F_n)|$ represents the number of received interests. $\zeta(\psi(t_i, F_n)[j])$ is a function to determine whether there is a cache hit for this interest. 1 means there is a cache hit. 0 means no hit.

Definition 2 (Average Request Interval(ARI)): The average request interval represents the average interval between requests in the same interface within the time slice. Frequent attacks will cause a smaller interval. The specific formula is defined as follows:

$$t(t_i, F_n) = \frac{\sum_{j=2}^{|\psi(t_i,F_n)|} \xi(\psi(t_i, F_n)[j]) - \xi(\psi(t_i, F_n)[j-1])}{|\psi(t_i, F_n)| - 1}, \tag{2}$$

where $\xi(\psi(t_i, F_n)[j])$ denotes the arrival time of the j-th interest.

Definition 3 (Request Frequency(RF)): It denotes the number of requests from the same interface in one time slice. This indicator is often used in detecting timing attacks [7]. Besides, in order to eliminate interference from network fluctuations and obtain accurate RTT, repeated requests and recursive attempts are common means of launching the time attack.

$$f(t_i, F_n) = |\psi(t_i, F_n)| \tag{3}$$

Definition 4 (Number of Types of Requested Contents(NTRC)): It denotes the number of types of requested contents from the same interface in one time slice. When launching timing attacks, the attacker has a high probability of requesting fewer varieties of contents to ensure that the contents can be cached in the router. For this reason, we select the number of types of request contents as the last feature of timing attacks.

$$c(t_i, F_n) = |\gamma(\psi(t_i, F_n))|, \tag{4}$$

where $\gamma(\psi(t_i, F_n))$ denotes a set of the requested contents from the n-th interface in the i-th time slice, and $|\cdot|$ denotes the set size.

In DTAL, we use the following steps to detect timing attacks:

Step 1: We mainly extract the features from the received interests and generate the feature vectors. First, we record all the interests to generate Table II in the time window of α seconds. Then, we classify the interests in Table II according to the interface number. To achieve precise statistics, we divide the time window of α seconds into β time slices and make statistics in each slice. Finally, we generate a vector including the cache hit ratio, average request interval, request frequency and number of types of requested contents for each slice and generate the following matrix for β time slices:

$$\begin{pmatrix} CHR_1 & ARI_1 & RF_1 & NTRC_1 \\ CHR_2 & ARI_2 & RF_2 & NTRC_2 \\ \cdots & \cdots & \cdots & \cdots \\ CHR_{\beta-1} & ARI_{\beta-1} & RF_{\beta-1} & NTRC_{\beta-1} \\ CHR_\beta & ARI_\beta & RF_\beta & NTRC_\beta \end{pmatrix} \tag{5}$$

where the i-th row represents the vector of the i-th time slice, $i = 1, 2, \cdots, \beta$.

TABLE II
INTEREST INFORMATION TABLE (IIT)

Arrival Time	Content Name	Cache hit	Interface ID
0.080713	/domain1.com/files/movie1.mp4/v2/s1	No	1
0.081746	/domain1.com/files/movie1.mp4/v2/s2	No	1
...	...	...	...
0.181244	/domain5.com/pictures/pic6.jpg/s10	Yes	5
0.182244	/domain5.com/pictures/pic6.jpg/s11	Yes	5
...	...	...	...

Step 2: LSTM model is adopted to determine whether the time attack has been launched on some interfaces in the time window of α seconds. Although, the traffic from an interface is defined as a matrix of β * 4 in **Step 1**, the different eigenvalues in the matrix have different domains. For example, the domain of the cache hit ratio is between 0 to 1, but the request frequency may be a value of hundreds or thousands. The request frequency will dominate the learning algorithm on account of its large value, which may influence the learning of other features. Common normalization methods, such as StandardScaler and MinMaxScaler, cannot cope with this problem

well. Therefore, we design a new normalization method to normalize values in different domains to between -1 and 1. Besides, we can guarantee that a large value remains large after normalization. The specific formula is as follows:

$$Scal_i = \frac{f_i - Ave_i}{max(f_i, Ave_i)}, \tag{6}$$

where $Scal_i$ represents the i-th normalized feature , f_i denotes the i-th original feature, and Ave_i represents the average value of the i-th feature without any attack. Then, we input the normalized feature matrix into the trained LSTM model to detect timing attacks.

IV. Performance Evaluation

In order to evaluate the performance of our detecting scheme, we conduct our simulations in ndnSIM1.0 [17], a popular open-source NDN simulator based on NS-3. All the experiments are simulated in ndnSIM1.0 on a local computer equipped with an Intel Core i7 2.50GHz CPU, 16 GB RAM and MacOS 10.13.2. Primary parameters used in our simulations are provided in Table III.

TABLE III
Simulation Parameters

Simulation Parameters	Value
Link(Router to Router)	50Mb/s
Link Delay(Router to Router)	5ms
Number of contents	10,000
CS size	100
PIT size	12,000
Cache Strategy	LRU
Legitimate Request	1,000 interests/s
Content Size	1MB

In our simulations, we adopt the same parameters in our paper [5]. The total number of the content items M is 10,000 and the cache size is equal to 1% of M, i.e., 100. Each content item has the same size (1 MB). The PIT size of each router is 12,000 entries. The cache replacement policy is LRU (Least Recently Used) [18], which is commonly used by in-network caching. The popular AS-3967 network topology is used as our network setting [19], as illustrated in Fig. 3. AS-3967 network topology is an actual ISP network topology snapshot. We set the rate of legitimate consumers requesting contents as 1,000 interests per second, which follows the Zipf-like distribution.

Similar to the method in [9], we perform a total of 2,600 seconds of simulation experiments. We select eight leaf nodes as adversaries from A_0 to A_7 and eight normal nodes, including C_3, C_{10}, C_{16} and C_{40} that share the same edge routers with malicious nodes, and C_0, C_4, C_{22} and C_{35} otherwise. Intuitively, the request frequency of adversaries are higher than that of normal nodes. We set the request frequency of each attacker to be a random value between 800/s and 1,500/s. Besides, to increase the difficulty of detection, the attack duration of eight adversaries is 0.1s, 0.2s, 0.3s, 0.4s, 0.5s, 0.6s, 0.8s, and 1s, respectively. All attackers follow a randomized uniform pattern. In the first 100 seconds, all leaf nodes would

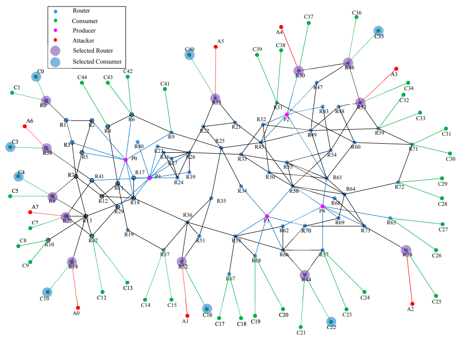


Fig. 3. AS-3967 Topology

perform normally. Starting at 100th second, adversary nodes will launch timing attacks every ten seconds.

A. Performance Metrics

Our main comparisons are made between the proposed DTAL scheme, DTARC [9], DTAIC [7] and DAPSVM [10]. Our DTAL scheme leverages the LSTM model to detect timing attacks by dividing a time window into several small time slices and making statistics in each slice. DTAIC calculates cache hit ratio and request frequency in a time window to detect timing attacks. Similar to DTAIC, DTARC detects timing attacks by calculating RTT and cache hit ratio in a time window. If these values exceed the corresponding thresholds, attacks will be determined. DAPSVM determines timing attacks by adopting SVM to classify the traffic into normal traffic and abnormal traffic. In our simulations, we mainly use the following performance metrics:

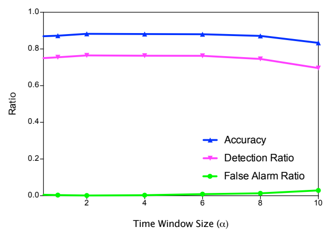
- **False Alarm Ratio** It is defined as the ratio of the number of normal requests which are misclassified as attacks to the total number of normal requests.
- **Detection Ratio** It is defined as the ratio of the number of correctly identified malicious requests to the number of all malicious ones.
- **Classification Accuracy** It is defined as the ratio between the number of correctly identified requests and total detected requests.
- **F-measure** It represents the harmonic mean of the precision rate and the recall rate.

B. Performance Comparison

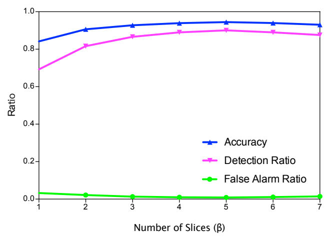
In this section, we first determine the optimum values of the time window size $\boldsymbol{\alpha}$ and the number of slices $\boldsymbol{\beta}$. Then, we compare DTAL with DTARC, DTAIC and DAPSVM on our data set.

In this module, we evaluate the impact of α on our scheme and set β to be 1. Fig. 4(a) shows that, when $\alpha > 2$, the detection performance of our scheme decreases as α increases. In general, α can be chosen as 2-4 seconds. We will choose $\alpha = 2$ seconds for the rest of the simulations unless specified otherwise.

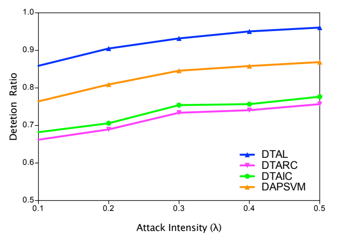
In Fig. 4(b), when $\beta < 5$, the performance of our scheme improves as β increases. It indicates that capturing eigenvalues



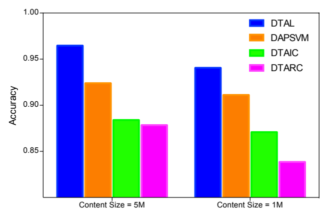
(a) Impact of α on DTAL



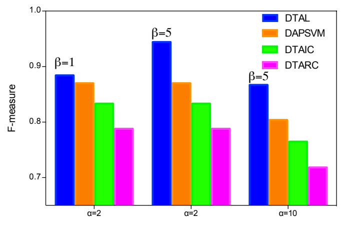
(b) Impact of β on DTAL



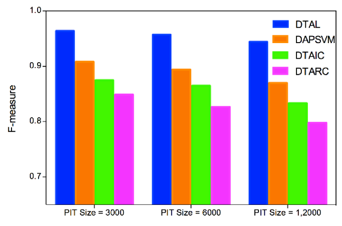
(c) Detection Ratio vs. Attack Intensity



(d) Classification Accuracy vs. Content Size



(e) F-measure with α and β



(f) F-measure vs. PIT Size

Fig. 4. Performance Evaluation

TABLE IV
FALSE ALARM RATIO

Scheme	$C3$	$C22$
DTARC	0.093	0.009
DTAIC	0.081	0.008
DAPSVM	0.038	0.004
DTAL	0.024	0.002

from multiple time slices can improve the accuracy of attack detection. Consequently, we set β = 5 in the following simulations. Further increasing β beyond 5 gets diminishing results, in addition to heavier computation cost.

False Alarm Ratio: We select $C3$ shared the same edge router with the malicious node $A6$ and $C22$, which is away from malicious nodes, to evaluate the false alarm ratio.

As Table IV shows that the interests generated by $C3$ is more likely to be misclassified as malicious interests than the interests generated by $C22$. Regular nodes that are close to attackers are classified as attackers mistakenly because their interest requests got mixed up with those from attackers (node $C3$). Instead, nodes that are far away from attackers ($C22$) are less likely to be mistaken as attackers.

Detection Ratio: Fig. 4(c) shows the relationship between the detection ratio and the attack intensity λ, which is defined as the ratio between malicious interests and the total number of interests. It is obvious that the detection ratio increases with λ. Furthermore, the detection ratio of our DTAL can remain at around 85% even when λ gets the minimum value. It demonstrates our scheme can detect the attack easily even under the slightest attack.

Classification Accuracy: In Fig. 4(d), all schemes can achieve higher classification accuracy as the content size increases. As the size increases, less kinds of content will be cached, which results in a lower cache hit ratio and therefore causes a higher ratio of correct detection. Since we leverage more features to detect timing attacks, our DTAL scheme distinguished normal traffic and abnormal traffic more accurately.

F-measure: In Fig. 4(e), F-measure of our DTAL increases with β when α is equal to 2. F-measure of all schemes decreases as α increases. The superiority of the proposed DTAL scheme is clearly shown.

Fig. 4(f) shows F-measure of all schemes decreases as PIT size increases. This could have been caused by the larger ratio of items that are cached by intermediate nodes as PIT size increases. Less items are retrieved directly from the original sources and it becomes harder to detect timing attacks accurately without raising false alarm rates.

In summary, our DTAL can obtain good performance with respect to all metrics and significantly outperform the baseline schemes.

C. Algorithm Complexity Analysis

In this section, we attempt to compare the time complexity of four algorithms in Table V.

TABLE V
ALGORITHM COMPLEXITY

Scheme	Complexity
DTARC	$O(N)$
DTAIC	$O(N)$
DAPSVM	$O(N^3)$
DTAL	$O(WN)$

Since both DTARC and DTAIC are only simple statistics and comparisons, their time complexity is $O(N)$ where N denotes the number of requests. DAPSVM adopts a machine

TABLE VI
COMPLEXITY EVALUATION

Scheme	Process Time	Memory Usage
DTARC	0.031852	18MB
DTAIC	0.030211	17MB
DAPSVM	0.178926	137MB
DTAL	0.167502	175MB

learning model SVM to classify traffic within a period of time. The complexity of this scheme comes from two aspects mainly. On the one hand, the router needs to count the number of bytes received and sent, the number of packets, and the number of interest packets periodically, which requires $O(N)$ complexity. Besides, the complexity of SVM itself is $O(N^3)$ [20]. Therefore, the overall complexity of this scheme is $O(N^3)$. Our DTAL scheme uses cache hit ratio, average request interval, request frequency and the number of types of requested contents to train a LSTM-based model to identify attack behaviors. In our scheme, feature extraction incurs a complexity of $O(N)$. Furthermore, the time complexity of LSTM at each time step is $O(W)$ [21], where W denotes the total number of parameters in the LSTM network. Therefore, the time complexity of our detection model is $O(WN)$.

Complexity Evaluation: In order to verify our conclusion on time complexity, we conduct simulations to calculate running time and memory usage of four schemes. 1,000 requests are send each second. The simulation results are summarized in Table VI. We can observe that DTARC and DTAIC consume similar processing time and memory usage, because they do no involve complex calculations. However, their performance of detecting the attack is worst. Compared with DAPSVM, our scheme has lower running time and better performance as discussed before.

V. CONCLUSION

In this work, we have proposed DTAL, a new LSTM-based scheme to detect timing attacks in Named Data Networks by dividing time windows into multiple equal time slices and extract the features in each time slice iteratively. The features are then vectorized and fed to train a nonlinear classifier to detect timing attacks' behaviors by adopting LSTM model. Our extensive simulations have demonstrated the superior performance of the proposed scheme in classification accuracy, detection ratio, false alarm ratio, and F-measure.

ACKNOWLEDGMENT

This research is sponsored by National Natural Science Foundation of China (contract/grant numbers: 61872053), National Key Research and Development Project of China (2017YFC0704100) and the Fundamental Research Funds for the Central Universities (DUT19GJ204)

REFERENCES

[1] D. Saxena, V. Raychoudhury, N. Suri, C. Becker, and J. Cao, "Named data networking: a survey," *Computer Science Review*, vol. 19, pp. 15–55, 2016.

[2] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. Claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named data networking," *Acm Sigcomm Computer Communication Review*, vol. 44, no. 3, pp. 66–73, 2014.

[3] A. Seetharam, A. Seetharam, and A. Seetharam, "On caching and routing in information-centric networks," *IEEE Communications Magazine*, vol. PP, no. 99, pp. 1–6, 2017.

[4] "National science foundation of future internet architecture (fia) program. http://www.nets-fia.net/."

[5] L. Yao, Z. Fan, J. Deng, X. Fan, and G. Wu, "Detection and defense of cache pollution attacks using clustering in named data networks," *IEEE Transactions on Dependable and Secure Computing*, 2018.

[6] G. Acs, M. Conti, P. Gasti, C. Ghali, G. Tsudik, and C. Wood, "Privacy-aware caching in information-centric networking," *IEEE Transactions on Dependable and Secure Computing*, vol. PP, no. 99, pp. 1–14, 2017.

[7] E. Dogruluk, A. Costa, and J. Macedo, "Evaluating privacy attacks in named data network," in *Computers & Communication*, 2016.

[8] A. Mohaisen, H. Mekky, X. Zhang, H. Xie, and Y. Kim, "Timing attacks on access privacy in information centric networks and countermeasures," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 6, pp. 675–687, 2015.

[9] E. Dogruluk, A. Costa, and J. Macedo, "Identifying previously requested content by side-channel timing attack in ndn," in *International Conference on Future Network Systems & Security*, 2018, pp. 33–46.

[10] D. Goergen, T. Cholez, J. Franois, and T. Engel, "Security monitoring for content-centric networking," *Lecture Notes in Computer Science*, vol. 7731, pp. 274–286, 2013.

[11] G. Acs, M. Conti, P. Gasti, C. Ghali, and G. Tsudik, "Cache privacy in named-data networking," in *Distributed Computing Systems (ICDCS), 2013 IEEE 33rd International Conference on*. IEEE, 2013, pp. 41–51.

[12] Q. Wu, Z. Li, G. Tyson, S. Uhlig, M. A. Kaafar, and G. Xie, "Privacy-aware multipath video caching for content-centric networks," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 8, pp. 2219–2230, 2016.

[13] N. Abani and M. Gerla, "Centrality-based caching for privacy in information-centric networks," in *MILCOM 2016-2016 IEEE Military Communications Conference*. IEEE, 2016, pp. 1249–1254.

[14] N. Ntuli and S. Han, "Detecting router cache snooping in named data networking," in *International Conference on Ict Convergence*, 2012.

[15] N. Kumar and S. Srivastava, "A triggered delay-based approach against cache privacy attack in ndn," in *2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS)*. IEEE, 2018, pp. 22–27.

[16] R. Tourani, T. Mick, S. Misra, and G. Panwar, "Security, privacy, and access control in information-centric networking: A survey," *arXiv preprint arXiv:1603.03409*, 2016.

[17] A. Afanasyev, I. Moiseenko, L. Zhang *et al.*, "ndnsim: Ndn simulator for ns-3," *University of California, Los Angeles, Tech. Rep*, vol. 4, 2012.

[18] E. J. Rosensweig, J. Kurose, and D. Towsley, "Approximate models for general cache networks," in *2010 Proceedings IEEE INFOCOM*. IEEE, 2010, pp. 1–9.

[19] H. Salah, M. Alfatafta, S. SayedAhmed, and T. Strufe, "Comon++: Preventing cache pollution in ndn efficiently and effectively," in *IEEE Conference on Local Computer Networks*. IEEE, 2017, pp. 1–9.

[20] A. Abdiansah and R. Wardoyo, "Time complexity analysis of support vector machines (svm) in libsvm," *International Journal of Computer Applications*, vol. 128, no. 3, pp. 28–34, 2015.

[21] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," *arXiv preprint arXiv:1402.1128*, 2014.