Mitigating Cache Pollution Attack in Information Centric Mobile Internet

Jia Chen[1*], Liang Yue[2], and Jing Chen[1]
[1] National Laboratory of Next Generation Internet Interconnection Device, Department of Electronic and Information Engineering, Beijing Jiaotong University
Beijing, China
[e-mail: chenjia@bjtu.edu.cn]
[2] China Unicom Network Technology Research Center,
Beijing, China
[e-mail:yuel10@chinaunicom.cn]
*Corresponding author: Jia Chen

Received January 22, 2019; revised June 5, 2019; accepted June 27, 2019; published November 30, 2019

Abstract

Information centric mobile network can significantly improve the data retrieving efficiency by caching contents at mobile edge. However, the cache pollution attack can affect the data obtaining process severely by requiring unpopular contents deliberately. To tackle the problem, we design an algorithm of mitigating cache pollution attacks in information centric mobile network. Particularly, the content popularity distribution statistic is proposed to detect abnormal behavior. Then a probabilistic caching strategy based on abnormal behavior is applied to dynamically maintain the steady-state distribution for content visiting probability and achieve the purpose of defense. The experimental results show that the proposed scheme can achieve higher request hit ratio and smaller latency for false locality content pollution attack than the CacheShield approach and the baseline approach where no mitigation approach is applied.

Keywords: Information Centric, Cache Pollution Attacks, Mobile Internet, Content Popularity, False locality

This research was supported by a research grant from the NSFC on Grant no. 61471029 and National Science and Technology Major Project under grant no. 2013ZX03006002

http://doi.org/10.3837/tiis.2019.11.022 **ISSN : 1976-7277**

1. Introduction

The rapidly increasing amount of wireless traffic has driven the development of mobile network. Particularly, 4G or LTE are currently widely adopted for most mobile operators. And it is expected that 5G will be commercialized in the near future. On the other side, for improving the users data obtaining experience, one of the promising design of Internet architecture-information centric networking (ICN) has been proposed[1]. As opposed to the traditional user centric networking, such as IP network, the ICN names content objects, and uses the Content object Name (CoN) for data obtaining. This is fundamentally different from IP network, where data are obtained through user address. In addition, the content objects can be cached in the network node, or content router, so as to reduce the content obtaining distance and delay.

Several ICN architectures have been proposed, such as TRIAD[2], Data Oriented Network Architecture[3], PURSUIT[4], Named Data Network (NDN)[5], and SINET[6]. Caching functionality is applied in all above ICN architectures, while the main differences among them are naming rules and transmission policy for inquiring packets and data packets. Content centric networking (CCN), based on which the NDN project is developed, is a represented ICN architecture, where inquiry packets are transmitted toward content provider using content object name (CoN) in the routing table, namely forwarding information base (FIB)[7]. And data packets are forwarded in the reverse path of the inquiry packets. However, the large size of CoN based routing table has driven the development of scalable routing table in CCN. Particularly, a scalable CCN architecture is proposed, where a name mapping system is proposed for storing the relationship between CoN and router based names [8,9]. One of our previous work has also addressed the forwarding algorithm design to improve the link failure resiliency in the scalable CCN system[10]. The use of ICN and caching in wireless network is promising considering the advantage of improving the user's Quality of Service. On the other side, wireless core network (such as wireless gateway) and wireless access network (such as base stations) are capable of caching data objects, in order to reduce the transmission delay and wireless traffic load[11,12]. Based on this, we consider a scalable CCN based information centric mobile Internet architecture in this paper.

However, in information centric mobile Internet, new security issues, such as cache pollution attacks are introduced[13]. In cache pollution attack, the attackers require content object with less popularity more often in order to disrupt content popularity distribution [14]. The attackers can either require continuously new and unpopular content objects (locality disruption), or repeatedly require unpopular content objects (false locality). By this means, unpopular content objects are cached in base stations and wireless gateways. And the latency for retrieving data for legal users increases. Several research works have been proposed to mitigate cache pollution attack in ICN. In [15], the authors propose CacheShield, in which a shielding function is applied and returned content objects are cached with a probability depending on the frequency the content is visited. CacheShield can improve the cache robustness for locality disruption. However, false locality content pollution attack cannot be solved by this approach. In [16], the author propose an adaptive Neuro-Fuzzy Inference System for cache replacement method for mitigating false-locality and locality disruption content pollution attacks in NDN. However, the system involves high computation overhead for computing and storing statistics. Moreover, as far as our knowledge, no existing research has considered the design and implementation of mitigating content pollution attack in information centric mobile Internet scenarios.

In this paper, we aim to solve the false locality based content pollution attack in the information centric mobile Internet scenarios. Particularly, we propose to detect cache pollution attack based on the calculated and predicted *Content Popularity statistics* and defend the attack by using *Probabilistic Caching* mechanism (CPPC algorithm). The main contributions of this paper can be summarized as follows.

Firstly, an attack detection algorithm based on content popularity distribution statistics is proposed. Particularly, we propose to store and analyze the content historical popularity and predict users future content request pattern by using multiple linear regression fitting method.

Secondly, a probabilistic based caching strategy is designed for defending cache pollution attack, which dynamically adjusts caching strategy on the basis of the degree of abnormality of CoN visiting behavior.

Thirdly, we build a prototype and implement the above detecting and defending algorithm in an LTE based information centric mobile network. We compare our proposed solution with CacheShield approach [15] and a baseline approach ALWAYS. The experimental results verify the advantages of the proposed method in terms of improving cache hit ratio of legal users and latency reduction for legal users.

The rest of this paper is organized as follows: Section 2 briefly discusses the related works. Section 3 describes the system model and general design. Section 4 presents the attack detection design based on content popularity distribution statistics. Section 5 presents *probabilistic caching* based attack defending algorithm. Section 6 provides the experiment results. Section 7 concludes the paper.

2. Related Work

Limited research has been performed for mitigating content pollution attach in ICN. In [13], randomness check of a matrix is proposed for detecting content pollution attack in CCN. A real-time scheme is proposed which can detect low-rate attacks on caches. A binary matrix is constructed based on the content objects stored in the cache. And the rank value of the binary matrix is used to detect the change of the content object pattern in the content store. A statistical sequential analysis based on cumulative sum is used and the attack is detected when a predefined threshold of the matrix-rank is achieved after some consecutive time cycles. However, the above approaches cannot detect false locality attacks.

CacheShield is proposed for enhancing cache robustness in CCN [15]. CacheShield mainly includes two components in the design: a content object name record component and a shielding function component. The content object name record component records any content name required by the consumer which has not been stored in the content store. The shielding function component is designed to decide the probability that whether the returned data objects can be cached. When cache is hit in the content router, no action is performed. On the other hand, when no cache is hit, the content router will forward the request to the upstream routers. When the new data object is returned, data object is stored in the cache with a probability given by the shielding function $\Psi(r)=\frac{1}{1+e^{(p-r)/q}}$. Here r is the number of requests for the data object, which is obtained through the content object name record in the content router. p and q are constants which can be adjusted. If the cache is stored, the CoN record can remove the corresponding item. Otherwise, the counter in the name record r is updated for the returned data object for the purpose of evaluation of the shielding function. However, the attacker can predict the value of r and request unpopular contents based on r to perform the content pollution attack.

Furthermore, Conti et al. proposed a lightweight mechanism with the aid of machine learning technique [17]. The attack detection process involves a learning process and an attack test. A set of content objects are selected through random sampling. And the content objects belonging to the set are monitored and analyzed for abnormal behavior detection. However, the above method only considers locality disruption content pollution attack.

An adaptive Neuro-Fuzzy Inference System is proposed as a cache replacement method for mitigating false-locality and locality disruption content pollution attacks in NDN [16]. The authors take advantage of both artificial neural networks and fuzzy systems for obtaining a nonlinear relationship between the inputs and outputs. The characteristics and patterns of the cached data are considered as the input and the type of attack is considered as the output. However, the data preparation for extracting statistics for every content object envolves significant system cost. Giulia *et al.* consider a false locality based attacking senerio, in which an attacker provider intents to pollute a content router by distributing zombies to require its content objects [18]. Accordingly, a honeypot is proposed to be installed for the node with degraded performance. And malicious content object names can be obtained and sent to the upstream content routers as blacklist. However, additional stations are required for performing monitoring functions.

Content popularity based caching methods are well investigated in the literature. Wei Koong Chai et al. demonstrate that selective caching is more efficient than ubiquitous caching [19]. In addition, a centrality-based caching algorithm through using the betweenness centrality is devised to improve the caching gain, where only part of the nodes are used to cache data. Based on this consideration, many researches have taken content popularity as a consideration in selective caching strategies [20,21]. In [20], WAVE strategy is proposed, which is a popularity-based cooperative caching strategy. The caching of content chunk is decided by the access frequency. The number of content chunks to be cached increases exponentially with the content chunk access frequency. And the upstream routers indicate the downstream router whether a content chunk can be cached. In [21], IPEC strategy is proposed which is an edge-priority cache strategy and considers difference of content chunk popularity for the same file. The forepart of the streaming file which is usually more popular is cached in the edge router. And a cache pipeline is constructed from the edge router to the central server based on the popularity of different parts of the file. However, none of the above works consider content pollution attack detection and mitigation method.

3. System Model and General Design

3.1 System Model and Assumptions

Fig. 1 depicts the system model of the LTE-based information centric mobile Internet. We use CCN with name mapping as the ICN architecture [8-10]. The mobile access network is responsible for terminal access control and authentications. The CCN with name mapping is responsible for interest and data forwarding. The source Access IDentification (AIDs) is used to identify terminal source and the Routing IDentification (RID) is used to route in information centric mobile Internet. The PGW is located between the mobile network and the Internet. We assume that contents are allowed to be cached in the content store (CS) for both the mobile network (eNodeB and PGW) and the core Internet (content router). Mobile users can send interest packet including the CoN to the network and accordingly, data objects are returned through the same reversed path. The cache for eNodeB, PGW and content routers has limited size. And least recently used (LRU) method is used for replacing the content when the cache is

full. The attackers are assumed to access to the Internet through the eNodeB and has the capability to compromising the performance for a set of the mobile user (MU)s by issuing unpopular interest requests through the eNodeB. The attacker MUs can change the CoN and request frequency. However, the attacker cannot affect the PIT, R-FIB information in the routers. It is assumed that eNodeB, PGW and content routers have the capability for mitigating the attack. Based on the above system model and assumptions, the basic communication process is stated in the following.

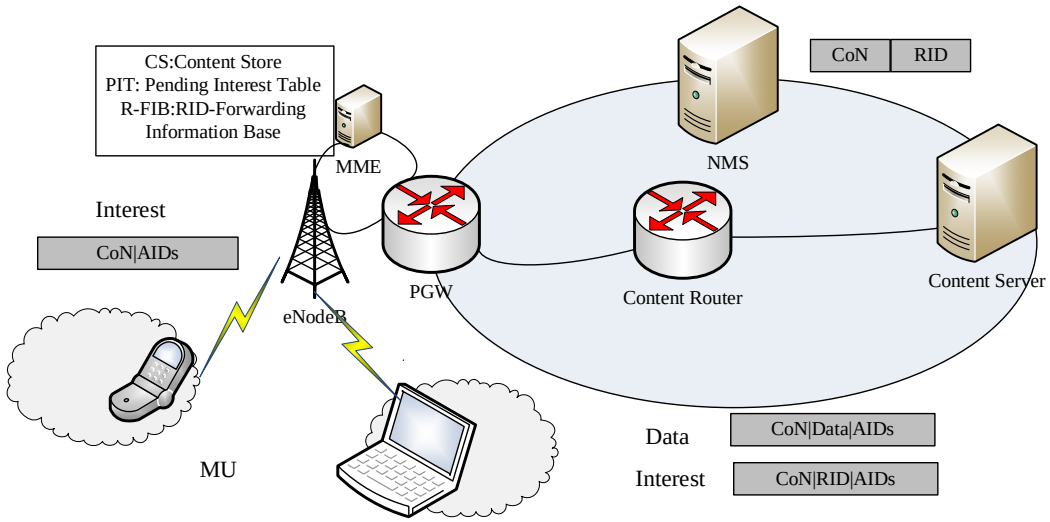


Fig. 1. The system model for the LTE based information centric mobile Internet

Step 1: An MU sends interest to the eNodeB. The interest packet includes CoN and AIDs.

Step 2: The eNodeB processes interest packet and matches the CoN on eNodeB. If the CoN is matched, corresponding data packet is returned to MU. Otherwise, the interest packet is forwarded to PGW.

Step 3: SGW/PGW processes the interest package. Similarly, if the CoN is matched, the data packet is returned back to the eNodeB. Otherwise, the CoN is sent to name mapping system (NMS).

Step 4: NMS will resolve CoN to RID by looking up its database and sending back the information. Then a new interest packet is constructed and sent in the core Internet, in which RID is used for routing the interest.

Step 5: When the interest packet reaches the content provider, or the CoN is matched on the content router, data packet is constructed and returned back to the PGW.

Step 6: PGW will further send the data packet back to the eNodeB and MU.

3.2 Problem Statement and General Design

In the above system, cache pollution attacks will occur when illegal MUs destroy the original stability of the cache by maliciously sending request packets with low popularity. This can reduce the validity of caching in both mobile network and Internet and increase the delay of legal users for obtaining data. Therefore, an attack detection and defense system is necessary to improve cache efficiency and defend cache pollution attacks.

The design consideration for CPPC method includes the following two aspects: attack detection and attack defending.

First, the attack detection is required to detect abnormal request packets. Particularly, content requiring behaviors are recorded and content popularity distribution statistics are collected. In addition, the content requiring trend is predicted by using the linear regression fitting method using historic content popularity information. The abnormal malicious attack can be detected by comparing the predicted content requiring popularity and the current content requiring behavior.

Second, the attack defending is required to selectively cache the data content. Once the malicious attack is detected, the attack defending function is performed by caching the returned data content with a probability based on the attacking behavior, or the difference between the predicted content requiring probability and the actual content requiring behavior. In addition, the status of the content requesting statics is continually calculated even for abnormal CoN. If no abnormal behavior is detected after some period, the CoN requesting behavior is recovered.

Note that both the content pollution detection and defending procedures are performed distributed over the routers (including eNodeB, PGW and content routers) which own the capabilities for caching data objects in the content stores. Thereby, the design requirement for the mitigation approach is to allow the routers including eNodeB, PGW, and content routers to perform the attack detection and defending mechanisms. Cooperation among the routers is not required.

4. Content Popularity based Attack Detection Design

The proposed CPPC algorithm includes two parts: content popularity statistics based attack detection and probabilistic caching based defending mechanism. In this section, content popularity based attack detection algorithm is presented. The algorithm includes four steps: the method for calculating content popularity statistics, popularity prediction algorithm, weight parameter correction method and abnormal behavior determination method.

4.1 The Method of Calculating Content Popularity Statistics

Content popularity table (CPT) is used to store the calculated content popularity by counting the CoN requesting number within a period of time. It is assumed that the content sets in the cache nodes contain contents with a total of K class popularity. The number of request packages received by the network node at time t is represented by $N(t,k)$, where k represents the content popularity of the k-th type and is sorted in descending order. Smaller k represents higher content popularity. Then the CoN request probability of the interest belonging to the k-th type can be calculated using the following equation as

$$f = \frac{N(t,k)}{\sum N(t,k)}. \tag{1}$$

In the actual network, if there is no attack, the request rate of users to the content follows the Zipf-like distribution [22,23].

$$f(k,t) \sim \frac{1}{k^{-\mu}}. \tag{2}$$

In **Eqn. (2)**, μ represents the degree of concentration of the requested content. Larger μ causes more concentrated requests for the content with smaller *k*. If there is an attack, the malicious request destroys the original distribution by requesting unpopular content.

According to the definition of content popularity, we propose to calculate CPT as follows. CPT is periodically calculated by dividing time into access cycles. The time duration for each time cycle is denoted as ΔT. Within each ΔT, the requesting number of the CoN index *i* is presented by $N_{T(i)}$, where *T* is the current time instant. At the end of each time period, content popularity of CoN with index *i* is calculated by using

$$P_{T(i)} = \frac{N_{T(i)}}{N_{T(1)} + N_{T(2)} + \ldots\ldots N_{T(I)}}. \tag{3}$$

where *I* is the total number of requesting CoN items within ΔT.

If a CoN has been requested in the previous time cycle, but no request is received in this cycle, the CoN item is also included in the current time cycle. Accordingly, the CPT will mark the corresponding result for the CoN as *0*. If a CoN has not been requested for a successive *S* cycles, the CoN item in CPT is deleted.

4.2 Predicting Content Popularity Statistics

Predicting content popularity is crucial in mitigating cache pollution attack. The accuracy of prediction directly determines the accuracy of the attack detection mechanism. We propose to use a multiple linear regression fitting method as the predicting method. In addition, we propose to adjust weights of different historical statistics to improve the accuracy of predicting results.

The fitting algorithm is based on the fact that multiple historical popularity statistics and popularity prediction values are taken as independent variables and dependent variables respectively. The dependent variables are affected by multiple independent variables simultaneously, and there is no definite linear relationship among independent variables. At this point, a linear relationship between the independent variables and dependent variables with unknown parameters can be established. Accordingly, the unknown parameters can be calculated according to the value of known independent variables. Hence the regression parameters in the multiple linear regression model are derived [24].

To increase the accuracy for multiple linear regression model, more independent variables should be used. However, considering the computational complexity and memory cost, in this paper, the historical values in a finite period are adopted to predict the content popularity of the next cycle.

Mathematically, the linear relationship between dependent variables and independent variables is expressed as follows.

$$y_i = \alpha_0 + \alpha_1 x_{i1} + \alpha_2 x_{i2} + \ldots + \alpha_m x_{im} + \varepsilon_i \tag{4}$$

In **Eqn. (4)**, $x_{im}(i = 1, 2 \ldots\ldots I)$ represents known value or observed variable in the *i*-th set. y_i represents the dependent variable affected by the *m* variables in the *i*-th set. The total number of time cycles used to calculate the value of y_i is denoted as *m*. $\alpha_j(\mathrm{j} = 0, 1, 2 \ldots\ldots m)$ represents unknown parameters that describe the linear relationship between the dependent variable and each independent variable. ε_i represents the error value for the *i*-th set, which conforms to the standard normal distribution $\mathbb{N}(0,1)$. To calculate

dependent variable y_i, the unknown regression parameters $\alpha_j (j = 0,1,2.....m)$ are required. Similarly, we have the following relationships for all I elements for the whole set presented as

$$\begin{cases} y_1 = \alpha_0 + \alpha_1 x_{11} + \alpha_2 x_{12} + \ldots + \alpha_m x_{1m} + \varepsilon_1 \\ y_2 = \alpha_0 + \alpha_1 x_{21} + \alpha_2 x_{22} + \ldots + \alpha_m x_{2m} + \varepsilon_2 \\ \ldots \\ y_I = \alpha_0 + \alpha_1 x_{I1} + \alpha_2 x_{I2} + \ldots + \alpha_m x_{Im} + \varepsilon_I \end{cases}. \tag{5}$$

The matrix notation is shown as follows.

$$\mathbf{y} = \mathbf{X}\boldsymbol{\alpha}^{-1} + \boldsymbol{\varepsilon},$$

or

$$\begin{pmatrix} y_1 \\ y_1 \\ \vdots \\ y_I \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1m} \\ 1 & x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{I1} & x_{I2} & \cdots & x_{Im} \end{pmatrix} \times \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_m \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_I \end{pmatrix}. \tag{6}$$

Eqn. (6) can be solved by standard least square method, and regression parameters $\alpha_j (\mathrm{j} = 0,1,2.....m)$ can be approximated as

$$\boldsymbol{\alpha} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \tag{7}$$

We apply the above method to predict content popularity. x_{im} is the data of historical popularity for the CoN i. Y_i is the current content popularity for the CoN i. The process of calculating unknown regression parameters $\alpha_j (\mathrm{j} = 0,1,2.....m)$ is actually to calculate the influence of historical popularity in each period on the current value. Larger α_j indicates more influence of the popularity value of the corresponding time period.

When using the above method for predicting content popularity, for the first few time cycles, popularity value for each time period are collected. When enough popularity values are collected, the regression parameters can be calculated. We further illustrate the method for predicting popularity of the CoN with index i as an example.

The content popularity of the current time period is denoted as $P_i(t)$, and the historical content popularity are denoted as $P_i(t-1), P_i(t-2), P_i(t-3), \ldots\ldots$ The popularity of the next period is denoted as $\hat{P}_i(t+1)$.

The content popularity value in the continuous m periods before current period are used as the observation value to calculate regression parameters. Taking m as 2 as an example, we can rewrite **Eqn. (5)** as

$$\begin{cases} P_i(t) = \alpha_0 + \alpha_1 P_i(t-1) + \alpha_2 P_i(t-2) + \varepsilon_1 \\ P_i(t-1) = \alpha_0 + \alpha_1 P_i(t-2) + \alpha_2 P_i(t-3) + \varepsilon_2 \\ P_i(t-2) = \alpha_0 + \alpha_1 P_i(t-3) + \alpha_2 P_i(t-4) + \varepsilon_3 \\ P_i(t-3) = \alpha_0 + \alpha_1 P_i(t-4) + \alpha_2 P_i(t-5) + \varepsilon_4 \end{cases}. \tag{8}$$

in which $\alpha_j (j = 0,1,2)$ can be derived by using **Eqn. (7)**.

Accordingly, the predicted value of the content popularity of the *i*-th content in the (*t*+1)-th period is expressed as.

$$\hat{P}_i(t+1)=\alpha_0+\alpha_1 P_i(t)+\alpha_2 P_i(t-1) \tag{9}$$

The content popularity in each period *t* has two values. One is the predicted value $\hat{P}_i(t)$ calculated by the historical value of the content popularity during the (*t-1*)-th time period. The other is the actual value $P_i(t)$ that is calculated in the *t-th* time period. Obviously, $\hat{P}_i(t)$ is obtained about one time period ahead of $P_i(t)$.

Actually, both $\hat{P}_i(t)$ and $P_i(t)$ are used in the attack detection and defense mechanism. Therefore, both $\hat{P}_i(t)$ and $P_i(t)$ should be stored for a period of time for later calculations.

4.3 Weight Parameter Correction Method

The regression parameters calculated α_j reflects the influence of the previous period on the popularity prediction value. Larger value indicates higher influence. The premise of calculating α_j accurately is that no attack occur for the historic period. When the cache pollution attack occurs within a certain period, the content popularity in that period is not accurate and should not be used as the basis to predict future content popularity. So, the weight of the content popularity in the attacking period should be adjusted to reduce the impact on the predicted value.

Therefore, we propose a weight adjustment parameter β shown in **Eqn. (10)**.

$$\beta=\frac{1}{\left|P_i(t)-\hat{P}_i(t)\right|} \tag{10}$$

Based on this, the content popularity is recalculated with the consideration of factor β when storing the historic popularity. Accordingly, the CoN popularity is recalculated as

$$\overline{\overline{P}}_i(t)=\beta\times P_i(t). \tag{11}$$

Substituting **Eqn. (10)** into **(11)**, we can devise

$$\overline{\overline{P}}_i(t)=\frac{1}{\left|P_i(t)-\hat{P}_i(t)\right|}\times P_i(t). \tag{12}$$

Therefore, the prediction calculation is revised as

$$\hat{P}_i(t+1)=\alpha_0+\alpha_1\overline{\overline{P}}_i(t)+\alpha_2\overline{\overline{P}}_i(t-1). \tag{13}$$

4.4 Algorithm of Detecting Abnormal Behavior

A deviation function is proposed to determine whether there exists malicious attack in actual request by comparing the actual request and predicted value. Particularly, the deviation function is designed as

$$\varepsilon_i(t)=\frac{P_i(t)-\hat{P}_i(t)}{\hat{P}_i(t)} \tag{14}$$

Where $P_i(t)$ and $\hat{P}_i(t)$ represent the actual content popularity and the predicted popularity of the CoN for index *i* at the *t*-th time cycle. Note that $\hat{P}_i(t)$ is derived at previous time cycle *t-1*. The deviation function can reflect the deviation between actual and estimated popularity. If the deviation reaches the threshold, or $\varepsilon_i(t) > \varepsilon_0$, it indicates that the request for CoN with index *i* has abnormal behavior in the time period *t*. In other words, if $\varepsilon_i(t) > \varepsilon_0$ (null hypothesis), we assume that content pollution attack occurs. However, there exist some situations/probabilities that the content pollution attack occurs when $\varepsilon_i(t) \le \varepsilon_0$ (alternate hypothesis). Therefore, the settings for threshold value is very crucial for the detection performance of the proposed method. Intuitively, we want to select the value of the threshold for maximizing the detection rate, or minimizing the undetected probability by decreasing the value of ε_0. Another important factor for selecting the threshold is the false alarm rate (FAR), which indicates the probability that no attack occurs when $\varepsilon_i(t) > \varepsilon_0$. This implies the probability that the detection has mistakenly determine that attack occurs when in the real situation no attack actually occurs. Both detection rate and FAR needs to be considered for deciding the value of ε_0. In the experimental evaluation, we first set the value of threshold to be 0.5 in the evaluation. And we also compare the detection rate and FAR by alternating the threshold.

5. Probabilistic Caching based Defending Algorithm

In this section, the probabilistic caching based cache pollution attack defending mechanism is introduced. Particularly, we propose to use the abnormal degree to adjust the cache probability. The deviation results obtained during abnormal behavior detecting process are used to determine different cache probabilities for different CoNs.

5.1. Probabilistic Cache Decision Algorithm

Based on the previous section, $\varepsilon_i(t)$ is calculated at the end of each time cycle are stored in the CPT. When the data object with the CoN is transmitted to the nodes, the caching policy of the data object with the CoN index *i* is adjusted based on the value of deviation function. Particularly, we propose to associate caching probability with the deviation function, which is described as follows.

1) For each time cycle *t* and CoN index *i*, compare the deviation value $\varepsilon_i(t)$ with the threshold ε_0.
2) For time cycle *t+1*, If $\varepsilon_i(t) \le \varepsilon_0$, the returned data object is cached.
3) If $\varepsilon_i(t) > \varepsilon_0$, the returned data object for CoN index *i* is cached with a probability of $f_i(t+1)$ for the next time cycle *t+1* where

$$f_i(t+1) = 1 - \varepsilon_i(t) \tag{15}$$

In **Eqn. (15)**, the range of $\varepsilon_i(t)$ is [0,1]. When $\varepsilon_i(t)$ approaches *1*, it indicates an increase of the probability of abnormal request behavior and higher possibility of pollution attack. This will lead to smaller value for caching probability of CoN index *i*.

5.2 User Reputation Management

The detection results shown in the previous section can also be applied for establishing a user reputation management system in the information centric mobile Internet, which can further improve the cache robustness and trace the source of abnormal users.

Since the interest request packet sent by the MU includes both CoN and AID information. AID can be used for identifying the user access identity. When an MU registers and enters the network, PGW assigns an IP address to the MU. The IP address is used as the connection identifier AID. And the IMSI code, user registration information and AID are actually stored in user service reputation management table in PGW. Once a CoN is found to be likely polluted, it is feasible to trace user behavior because of the established relationship among IMSI, CoN and AID/IP. Accordingly, a reputation level can be associated to the MU.

5.3 Limitations of CPPC algorithm

The CPPC method we proposed has introduced some complexities for the system design in terms of both storage complexity and computing complexity. The storage complexity mainly comes from the steps for storing the results of the content popularity for the CoN. Thereby, the storage complexity is directly proportional to the amount of CoN in the content store (number of items in CS). The storage complexity also depends on the linear regression model we are using. Increasing the historic period that is used before the current period will also cause the linear increases of the storage complexity. However, this will improve the accuracy for the calculation. Note that in the experimental evaluation part, we set the historic period value m=2.

On the other side, the computing complexity of the method mainly comes from predicting the content popularity using linear regression model and calculating the parameters using **Eqn. (7)**. The computing complexity relies on the size of X, in which the total number of row element is the amount of the CoN set and the total number of column element is the number of total time period m+1.

From the above analysis, it can be concluded that the CPPC method can introduce some storage complexity and computing complexity. Both complexities depend on two factors: the number of period used and the amount of CoN set. Therefore, one way to reduce the complexity is to reduce the number of time period used for calculating and predicting the popularity statistics. In our experimental implementation, we use the historic period value m=2. Another way to reduce the complexity is to cluster the CoN based on the similarity of the popularity value. We can also enlarge the time period for calculating and predicting the popularity statistics, which will result in less sensitive detection results.

6. Experimental Results

In this section, experimental results obtained through implementing the proposed algorithm are presented. First, the experimental environment and evaluation methodology used for evaluating the algorithm is presented. Then content hit ratio and round trip time (RTT) are plotted. In addition, we compare the proposed approach CPPC with a baseline approach ALWAYS caching approach and the CacheShield approach in [15].

6.1 Experiment Environment and Evaluation Methodology

We perform experimental study for evaluating the proposed CPPC method in the paper. The topology of the prototype LTE-based information centric network is shown in **Fig. 2**, which

consists of two MUs, an eNodeB/MME, an S-PGW, a Name Mapping Server, a content router and a content provider. The MUs are connected to the eNodeB by wireline connection at the current stage. The hardware platform for implementation includes Intel Xeon E3-1234 processor, 8G DDR and 8M cache. Linux is used as the operating system with kernel version 2.6.28. The basic EPC code for LTE is obtained from nwEPC [25]. And we also use the caching function of the content store module from CCNx [26]. Referring to the above code, the preliminary design for interworking between LTE and SINET is achieved. Based on this, we further implement the proposed CPPC method.

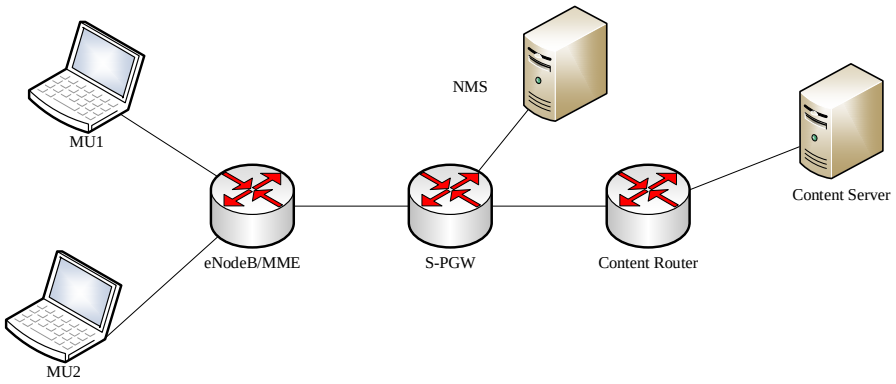


Fig. 2. Prototype topology of LTE-based information centric mobile Internet

Table 1. Experimental Parameters

Cache size	5GB
Content object size	5MB
Request rate of legitimate user MU1	10^3/s
Request process of legitimate user	Poisson process
Content request pattern of MU1	Zipf-Like(μ=1.2)
Content request distribution of malicious user MU2	Uniformly selected between CoN index 1500 to 1600
Cache replacement strategy	LRU

The details of the experimental parameters are shown in **Table 1**. The cache size for each router is set to be 5GB. The content size has fixed value as 5MB. Without loss of generality, we assume that the content request distribution for legitimate user follows the Zipf-like distribution with μ=1.2 over a given content object space $\{C_1, \ldots\ldots C_k, \ldots\ldots C_K\}$ of size $K=2^{12}$. The average request rate of legitimate user UE1 is 10^3content objects/s, and the content request process obeys Poisson process. The malicious user UE2 is requiring content objects at a constant rate, the value of which changes for different scenarios. Smaller CoN index value indicates higher popularity. And the legitimate user will require unpopular content with CoN index uniformly distributed between 1500 and 1600.

The cache replacement strategy is least recently used(LRU), where least recently used cache item is discarded first when cache is full. The duration for one time cycle is set to be 10 seconds. The experiments last for 30 minutes. During the first 5 minutes, all requests are sent by legitimate user. The malicious user starts sending malicious requests after 10 minutes. We use and change the ccnping-master tool to simulate both legitimate and malicious requests. By varying the request content name and patterns, the legitimate and malicious requests can be simulated. The threshold value ε in experiments in Section 6.2 and 6.3 is set to be 0.5.

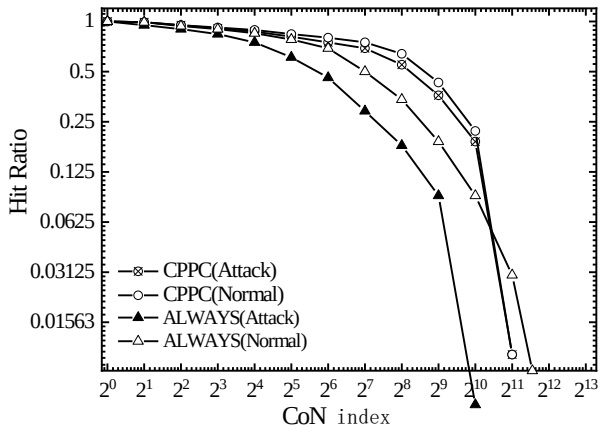
We compare our results with a baseline results ALWAYS mechanism and CacheShield [15]. In ALWAYs mechanism, no particular defending method is adopted. The network node simply caches any content that is received. In CacheShield, the received data object will be cached with a probability obtained from a shielding function, which is in inverse relationship with the cache visiting time counts.

Based on the above prototype platform, extensive numerical results are obtained in terms of content hit ratio and round trip time (RTT). The cache hit ratio is the ratio that the content is obtained from eNodeB for legitimate user. It is calculated as the ratio of the number of requests which can be satisfied by the eNodeB to the total number of request. The RTT is the average delay between sending request and data retrieval over the test period of 30 minutes. We also analyze the impact of the value of threshold on the detection performance including the detection rate and FAR.

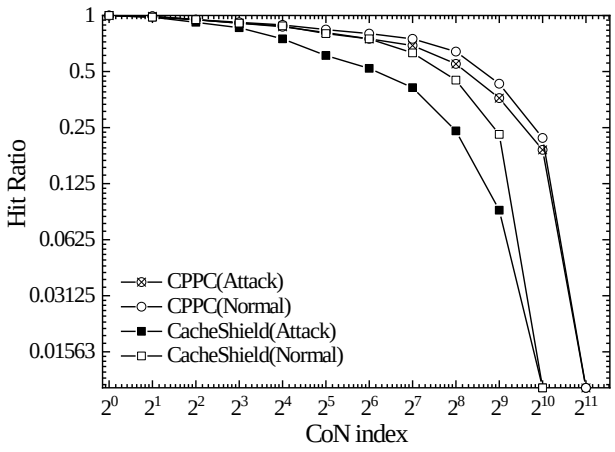
6.2 Cache Hit Ratio

The experimental results for hit ratio on eNodeB with respect to CoN index are plotted in **Fig. 3**. The malicious user is requesting content at a rate of $2*10^3$/s. **Fig. 3(a)** compares the hit ratio variations before and after the attack for the proposed CPPC and ALWAYS method. It can be observed that in the ALWAYS method, the cache hit ratio after the attack (shown in solid triangles) decreases obviously comparing with the hit ratio without attack (shown in hollow triangles). While for the proposed CPPC algorithm, the cache hit ratio after the attack (shown in crossed circles) only reduces slightly comparing with the cache hit ratio before the attack (shown in hollow circles). For example, for the CoN index 2^7, the difference of the hit ratio variation is around 25% for ALWAYS method, which is much larger than that of CPPC (around 5%).This illustrates that the cache hit ratio remains very stable after attack for the proposed CPPC algorithm.

In addition, in **Fig. 3(b)**, we compare the hit ratio variation results between the proposed CPPC algorithm with ALWAYS and CacheShield method. Similarly, it can be observed that an obvious reduction of cache hit ratio for CacheShield method exists. The cache hit ratio of MU1 before the attack (presented in hollow squares) reduces obviously comparing with the results after the attack (presented in solid squares). For example, for CoN index 2^7, the cache hit ratio reduces by around *25%* for CacheShield method. This further demonstrates the cache robustness of the proposed algorithm.



(a) Comparison between CPPC and ALWAYS



(b) Comparison between CPPC and CacheShield

Fig. 3. Impact of cache pollution attacks on the hit ratio of legitimate user MU1

In **Fig. 4**, the average hit ratio for CoN index from 1 to 20 (representing popular contents) of legitimate user MU1 is plotted with respect to different attack rate ratios for various approaches. The attack rate ratio is the ratio of content request rate of the legitimate users to that of the malicious users. With the increasing of request rate of the abnormal user, the hit ratio of legal user decreases obviously for ALWAYS approach represented by squares. That is because cache is quickly occupied by unpopular data objects. For CacheShield approach presented by triangles, higher attack rate increases the caching probability of the CoN. This leads to the abnormal data objects to be cached and reduces the hit ratio of legitimate user. However, in CPPC strategy denoted by circles, the hit rate remains almost the same for larger attack rate. This indicates that the proposed approach is more accurate for false locality based attack, since it is easier to identify the abnormal behavior.

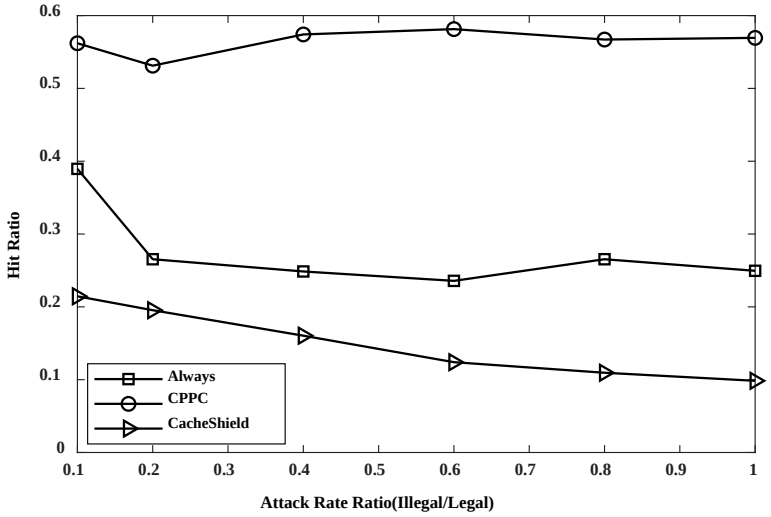


Fig. 4. Impact of attack rate on hit ratio of legitimate user MU1

Fig. 5 depicts the impact of the cache size on legitimate user hit ratio for various approaches. The attack rate is set to be $2*10^3$/s. Similar with **Fig. 4**, the CoN index ranges from 1 to 20 representing relative popular contents. It is observed that with the increasing of the cache size, the cache hit ratio increases. It is obvious since larger cache size will cause more data objects to be cached in the network, thus increasing the hit ratio.

In addition, it is observed that the proposed CPPC approach (presented by circles) achieves higher cache hit ratio than the CacheShield approach (presented by triangles) and ALWAYS approach (denoted by squares). For example, for cache size equals to 6GB, the hit ratio of the proposed CPPC approach is about 2.5% more than that of CacheShield approach and about 10% more than that of ALWAYS approach. This again verifies the effectiveness of CPPC approach.

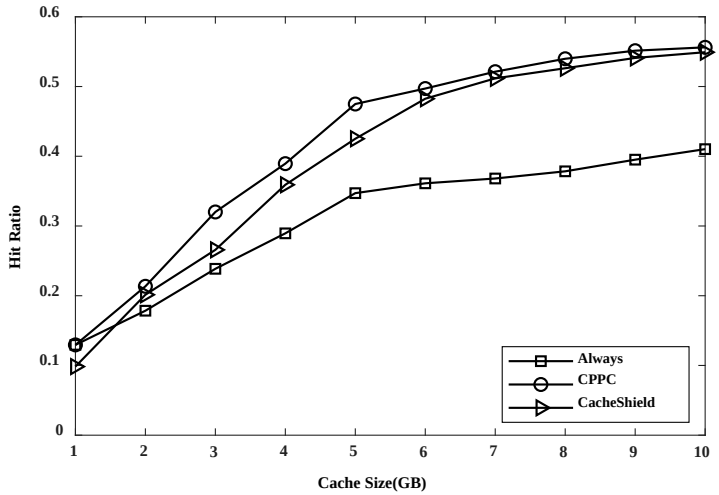


Fig. 5. Impact of cache size on hit ratio of legitimate user MU1.

6.3 RTT

In this section, the RTT results of legitimate user MU1 are plotted with respect to CoN index in **Fig. 6**. Similarly, the attack rate is set to be $2*10^3$/s. It is observed that with the increasing of CoN index, the RTT increases obviously because obtaining unpopular contents usually requires the packets to transmit over longer distance. In addition, we can observe that the RTT for the proposed CPPC approach (denoted as circles) shows the smallest RTT results when comparing with that of ALWAYS approach (represented by squares) and the CacheShield approach (denoted as triangles). For example, for CoN index 2^9, the RTT of CPPC approach is around *0.5*ms less than that of CacheShield approach. The experimental results further demonstrate the effectiveness of the proposed CPPC approach.

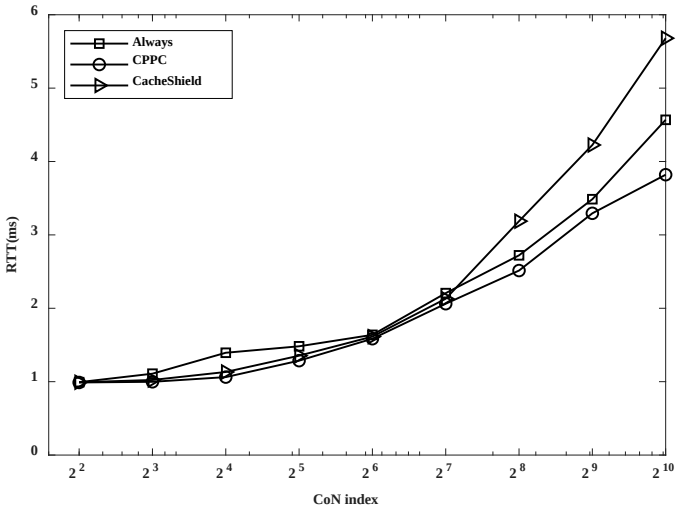


Fig. 6. RTT versus CoN index of legitimate user MU1

6.4 Impact of Threshold ε

In this section, we evaluate the impact of threshold ε on the content popularity detection rate and false alarm rate (FAR). In our experiment, detection rate is the fraction that the malicious content request from MU2 is detected to the total number of CoN requests from MU2. FAR is the fraction of the number that the abnormal behavior is determined for legitimate user MU1 to the overall number of requests from MU1. The attack rate is set to be more than 1%, 5%, 10%, 15% and 20% than the legitimate user MU1 respectively, which is denoted by Incremental Attack Rate Ratio (IARR). The detection rate versus the IARR is shown in **Table 2** for different threshold values ε=0.4, 0.6, 0.8.

Table 2. Detection Rate for Malicious User MU2

IARR / ε	1%	5%	10%	15%	20%
0.4	0.89	0.92	0.93	0.97	0.97
0.6	0.85	0.87	0.92	0.94	0.95
0.8	0.72	0.79	0.84	0.83	0.88

It can be seen from the results that with the increasing of threshold ε, the detection rate of CPPC decreases for a fixed value of IARR. That is because the detection of the attack in the CPPC algorithm mainly relies on the difference of the predicted content popularity and historic content popularity. And larger value of threshold ε indicates that the detection is less sensitive for attack detection. On the other hand, with the increasing of IARR, the detection rate increases for a fixed value of ε. This implies that increasing the attack rate will improve the attack detection accuracy in CPPC algorithm. For the threshold value ε=0.4, the detection rate for different IARRs can achieve at more than 90%.

Table 4. FAR for Legitimate User MU1

IARR / ε	1%	5%	10%	15%	20%
0.4	0.14	0.13	0.14	0.18	0.12
0.6	0.08	0.08	0.12	0.07	0.10
0.8	0.02	0.03	0.03	0.02	0.02

Table 4 shows the experiment results for FAR for legitimate user MU1. From the results, we can observe that increasing the threshold will cause the increase of FAR for legitimate user. While changing the value of IARR will have little impact on the FAR. This is because with the increasing of ε, the detection method is less sensitive, which will lead to decreasing value of FAR for legitimate user. Therefore, we need to consider both detection rate and FAR when choose the appropriate value of threshold ε.

7. Summary and Future Work

In this paper, we have designed CPPC for mitigating cache pollution attacks in information centric mobile Internet. Particularly, the content popularity statistic is calculated and predicted aiming to detect abnormal behavior. And a probabilistic caching strategy based on abnormal behavior has been proposed to dynamically maintain the steady-state distribution for content visiting probability and achieve the purpose of defense. We have built an LTE-based information centric Internet and implemented the proposed algorithm. The experimental results have been conducted and compared with CacheShield approach and the baseline approach ALWAYS. Extensive results show that the proposed scheme can yield higher request hit ratio and smaller RTT than the CacheShield approach and the baseline approach. The results have also shown that varying threshold values will lead to different detection rates for malicious users and different FARs for legitimate users.

To summarize, the experimental results have demonstrated the effectiveness and superiority of the proposed CPPC approach in mitigating cache pollution attack comparing with the previous approaches. The results also indicate that the value of threshold needs to be selected appropriately by leveraging both the detection rate and FAR. Future work includes extending the proposed approach considering cooperation among neighboring nodes.

References

[1] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. Katsaros, and G. Polyzos, "A survey of information-centric networking research," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 2, pp. 1024-1049, 2014. Article (CrossRef Link)

[2] D. R. Cheriton and M. Gritter, "Triad: A scalable deployable NAT-based internet architecture," *Technical Report, Stanford University*, January, 2000.

[3] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," in *Proc. of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, pp.181-192, August 27-31, 2007. Article (CrossRef Link)

[4] PURSUIT. http://www.fp7-pursuit.eu.

[5] Named Data Networking (NDN). http://www.named-data.org.

[6] Hongke Zhang, Wei Quan, Han-chieh Chao and Chunming Qiao, "Smart Identifier Network: A collaborative architecture for the future Internet," *IEEE Network*, vol. 30, no. 3, pp. 46-51, 2016. Article (CrossRef Link)
[7] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," *Communications of the ACM*, vol. 55, no. 1, pp. 117-124, January, 2012. Article (CrossRef Link)
[8] N. L. van Adrichem and F. A. Kuipers, "Globally accessible names in named data networking," in *Proc. of 2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 1–6, April 14-19, 2013. Article (CrossRef Link)
[9] A. Afanasyev, X. Jiang, Y. Yu, J. Tan, Y. Xia, A. Mankin, and L. Zhang, "NDNS: A DNS-like name service for NDN," in *Proc. of IEEE Conference on Computer Communications and Networks* (*ICCCN*), pp.1-9, 31 July-3 August, 2017.Article (CrossRef Link)
[10] Jia Chen, Bo Tong and Hongke Zhang, "Dynamic Interest Transmission Approach for Improving Link Failure Resiliency in Content Centric Network," *IEEE Transactions on Network and Service Management*, vol. 15, no. 2,pp. 665-678, June, 2018. Article (CrossRef Link)
[11] Xiaofei Wang, Min Chen, T. Taleb, A. Ksentini, and V. Leung, "Cache in the air: exploiting content caching and delivery techniques for 5G systems," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 131-139, February, 2014. Article (CrossRef Link)
[12] G. Paschos, E. Bastug, I. Land, G. Caire and M. Debbah, "Wireless caching: technical misconceptions and business barriers," *IEEE Communications Magazine*, vol. 54, no. 8, pp. 16-22, August, 2016. Article (CrossRef Link)
[13] H. Park, I. Widjaja, and H. Lee, "Detection of cache pollution attacks using randomness checks," In *Proc. of IEEE International Conference on Communications* (*ICC*), pp. 1096-1100, 10-15 June, 2012. Article (CrossRef Link)
[14] R. Tourani, S. Misra, T. Mick and G. Panwar, "Security, Privacy, and Access Control in Information-Centric Networking: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 1, pp. 566-600, First quarter, 2018. Article (CrossRef Link)
[15] M. Xie, I. Widjaja, and H. Wang, "Enhancing cache robustness for content-centric networking," in *Proc. of IEEE INFOCOM*, pp. 2426-2434, 25-30 March, 2012. Article (CrossRef Link)
[16] A. Karami and M. Guerrero-Zapata, "An anfis-based cache replacement method for mitigating cache pollution attacks in named data," *Networking Computer Networks*, vol. 80, pp. 51-65, 7 April, 2015. Article (CrossRef Link)
[17] M. Conti, P. Gasti, and M. Teoli, "A lightweight mechanism for detection of cache pollution attacks in named data networking," *Computer Networks*, vol. 57, no. 16, pp. 3178-3191, 13 November, 2013. Article (CrossRef Link)
[18] G. Mauri, R. Raspadori, M. Gerlay, and G. Verticale, "Exploiting information centric networking to build an attacker-controlled content delivery network," in *Proc. of 14th Annual Mediterranean Ad Hoc Networking Workshop*, pp. 1-6, 17-18 June, 2015. Article (CrossRef Link)
[19] W. Chai, D. He, I. Psaras, and G. Pavlou, "Cache 'less for more' in information-centric networks," in *Proc. of the 11th International IFIP TC 6 Conference on Networking*, pp. 27-40, May 2012. Article (CrossRef Link)
[20] K. Cho, M. Lee, K. Park, T. Kwon, Y. Choi, and S. Pack, "Wave: Popularity-based and collaborative in-network caching for content oriented networks," in *Proc. of IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 316–321, 2012. Article (CrossRef Link)
[21] S. Lim, B. Ko, and G. Jung, "Inter-Chunk Popularity-Based Edge-First Caching in Content-Centric Networking," *IEEE Communications Letters*, vol. 18, no. 8, pp. 1331-1334, 2014. Article (CrossRef Link)
[22] L. Breslau, P. Cue, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and zipf-like distributions: Evidence and implications," in *Proc. of IEEE Conference on Computer Communications (INFOCOM)*, pp. 126–134, 1999. Article (CrossRef Link)
[23] A. Mahanti, C. Williamson, and D. Eager, "Traffic analysis of a web proxy caching hierarchy," *IEEE Network*, vol. 14, no. 3, pp. 16–23, 2000. Article (CrossRef Lin)

[24] Hisashi Kobayashi, Brian L. Mark and William Turin, *Probability, Random Processes and Statistical Analysis*, Cambridge University Press, New York, USA, 2012. Article (CrossRef Link)
[25] Source code for nwEPC-EPC SAE Gateway. https://sourceforge.net/projects/nwepc
[26] Source code for CCNx. www.ccnx.org.

Jia Chen is an Associate Professor with the National Laboratory of Next Generation Internet Interconnection Device, Department of Electronic and Information Engineering, Beijing Jiaotong University. She received BEng degree in Communication Engineering in 2005 from Beijing University of Posts and Telecommunications. She received Master and PhD degree in 2006 and 2010 respectively from Department of Electrical and Electronic Engineering, University College London. Her current research interests include architecture and protocol design for the future Internet.

Liang Yue is a Network Engineer with China Unicom Network Technology Research Center, Beijing, China. She received BEng degree in Communication Engineering in Department of Electronic and Information Engineering in 2014 from Beijing Jiaotong University. She received Master degree in the National Laboratory of Next Generation Internet Interconnection Device from Department of Electronic and Information Engineering, Beijing Jiaotong University in 2018.

Jing Chen is a PhD student with the National Laboratory of Next Generation Internet Interconnection Device, Department of Electronic and Information Engineering, Beijing Jiaotong University. She received Bachelor degree in computer application technology in 2014 from Zhoukou Normal University (Henan, PRC), and received master degree in computer application technology in 2017 from Henan University of Science and Technology (Henan, PRC). Her research interests include applying artificial intelligence (AI) in the future Internet design.