Telecommunication Systems (2023) 82:345–362
https://doi.org/10.1007/s11235-022-00988-x

Distributed content placement in cache-enabled small cell networks in the presence of malicious users

Zahra Rashidi[1] · **Vesal Hakami**[2]

Accepted: 25 December 2022 / Published online: 21 January 2023

Abstract
Caching popular files at the edge of wireless networks has been proved to be an effective strategy to reduce the content delivery delay and alleviate the backhaul congestion. However, due to the vulnerability of the edge networks to malicious threats, it is essential to investigate security-aware caching strategies which can combat mischievous content requesting processes. These processes may defeat the whole purpose of edge caching by deliberately issuing requests incompatible with the known popularity distribution. In this paper, we investigate the content placement problem in cache-enabled Small Base Stations (SBSs) to minimize the downloading delay of contents. We consider that some users' requesting behavior may become compromised by an adversary and thus may not follow the known statistics of the content popularity . We formulate the problem using the notion of a multi-leader single-follower Stackelberg game between the SBSs and the adversary. The objective of each SBS is to minimize the congestion duration of the backhaul links and the average delay of the users within its coverage range, whereas the adversary aims at maximizing the congestion duration of the backhaul links by generating fictitious requests for non-cached files. Using the standard notion of a *potential function* in game theory, we propose an iterative algorithm that is provably convergent toward the Stackelberg equilibrium of the formulated game. Simulation experiments are conducted to validate the convergence of the proposed algorithm as well as to evaluate its performance under different mixed populations of malicious/non-malicious users.

Keywords Cache security · Content placement · Edge caching · Misbehavior · Stackelberg game

1 Introduction

1.1 Research background

The idea of content caching at the edge of wireless networks dates back to 3G and 4G cellular networks [1–5]. With the growth of data traffic over cellular networks, the current capacity levels cannot support this surge of traffic, even by allocating a new cellular spectrum [2]. Thus, novel approaches are required to more effectively utilize the existing limited communication resources. One such approach is to utilize the storage capacity of Small Base Stations (SBS) to cache popular contents that may be potentially requested repeatedly by different users. Instead of downloading popular content multiple times from remote servers via backhaul links, we can cache them at SBSs to bring the contents closer to Mobile Users (MUs). Thus, by locally responding to user requests for content, edge caching can effectively improve network performance by reducing the backhaul burden and alleviating access delay of content [6–8].

In this paper, our concentration is on the problem of content placement in Small Cell Networks (SCNs), while giving special attention to the security concerns alongside the performance issues. In general, security in the cache can be considered from different perspectives; for example, denial of service and jamming may disrupt the cache system performance and compromise the quality of service [27–35]. It is also essential to provide a service of secrecy and confidentiality to protect against eavesdropping [9–14] and to ensure

✉ Vesal Hakami
vhakami@iust.ac.ir

Zahra Rashidi
z_rashidi96@comp.iust.ac.ir

[1] School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran

[2] Center of Excellence in Future Networks, School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran

data integrity in the presence of attacks like content tampering [15–20]. In addition, in highly decentralized systems such as Device-to-Device (D2D) communications, it is vital to combat selfish behavior and to encourage the participation of the users in the caching process to reap the benefits of the network edge caching [21–26]. In the sequel, we identify the research gap and state our motivations to propose a novel scheme for maintaining the performance of edge caching in the presence of malicious threats.

1.2 Research gap and motivation

Among the possible security threats to content placement at the wireless edge, there is an important threat that has been less studied to date: *the fictitious content request generation attack*. Consider the case where some compromised edge devices are exploited by a malicious entity (e.g., attacker) to generate content requests in an incompatible way with the statistics of the known content popularity distribution. If the system's content placement strategy is computed only based on some statistical popularity model which is oblivious to such malicious behavior, it would result in a higher cache miss ratio, and can thus lead to increased congestion in backhaul links as a higher number of requests get redirected to the back-end origin servers.

To the best of our knowledge, few studies exist that particularly address the threat of fictitious content request generation. A related line of research contains those studies that provide a remedy against cache pollution attacks where attackers aim to occupy the limited caching space with unpopular contents. The main difference between our scheme and those addressing the pollution attacks is that we have more realistically assumed that the fictitious content requests may be generated strategically and adaptively to the caching decisions of the SBSs. For example, in the edge caching scenario of [35], the attacker is less sophisticated in the sense that it only learns the statistical content popularity and forge unpopular requests that deviate from the statistics. The attacker does not "strategically" react to the defender.

The only study that accounts for strategic behavior on the part of the attacker has been conducted by Gabry et al. in [27]. The authors have modeled the interaction of a caching system with malicious users whose request generation behavior may deviate from the established content popularity model. However, the authors have only assumed a small-scale system in which a centralized entity (e.g., a Macro Base Station (MBS)) makes the decisions as to which content should be placed in which SBS cache. They have formulated the conflict between the MBS and the malicious users as a standard Stackelberg game, and have computed the Subgame Perfect Nash Equilibrium (SPNE) [36]. In contrast, the caching decisions in our proposed system are made in a distributed fashion by the SBSs themselves without engaging the MBS.

Hence, our motivation is twofold: firstly, to account for more sophisticated misbehavior in the request fabrication process; secondly, to combat malicious behavior in distributed setups where the MBS can delegate the caching decisions on the SBSs themselves.

1.3 Contributions

In this paper, we consider a SCN in which the MUs issue content requests to multiple SBSs. The users may exhibit both legitimate/malicious requesting behavior. Unlike [27], we do not assume that the SBSs have the luxury of centralized computation of their caching strategies. Instead, we assume a larger setup where the SBSs themselves need to coordinate their content placement decisions with each other to minimize the delay experienced by the users located within their coverage region. At the same time, the SBSs also need to combat the adversarial requests that are issued to defeat the whole purpose of edge caching by congesting the backhaul links. The decentralized interplay between the SBSs on the one hand, and their confrontation with the malicious party on the other, call for a more complex problem formulation, which distinguishes our work from the existing research. In particular, we come up with the following contributions:

- We use a game-theoretic approach and formulate the content placement problem as a Stackelberg game between the SBSs and an attacker. To formally capture the inter-SBS interactions as well as the SBS-attacker conflicts, we define the Stackelberg game as a two-stage sequential play with a multi-leader single-follower structure. The SBSs as the leaders (or first movers in the parlance of the Stackelberg game) engage in a coordination game with each other to come up with an equilibrium-based content placement strategy. In the meantime, they should also optimize their joint play against the attacker (as the game's follower or second mover) to mitigate its adversarial impact on the efficacy of the overall cache system.
- To compute the equilibrium of the game, we first draw on the notion of *potential function* in classical game theory [37] to prove that the two-stage game between the SBS leaders and the malicious follower is equivalent to a one-shot potential game, and possesses a pure strategy equilibrium. We then exploit the *finite improvement* property of potential games to propose an iterative algorithm based on *best response dynamics* [38] which is provably convergent toward the equilibrium of the caching game.
- We conduct experimental studies to evaluate our proposed algorithm in terms of its convergence properties and measure its performance by varying the content popularity statistics as well as the number of SBSs and the fraction of malicious users.

Table 1 Used main acronyms

Acronym	Explanation
BRD	Best response dynamics
D2D	Device-to-device
MBS	Macro base station
MU	Mobile user
NE	Nash equilibrium
P2P	Peer-to-peer
SBS	Small base station
SE	Stackelberg equilibrium
SINR	Signal to interference plus noise ratio

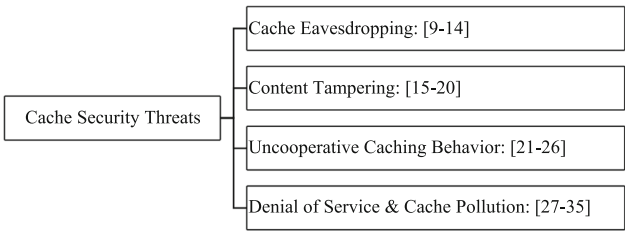

Fig. 1 Taxonomy of cache security threats

Table 1 lists the acronyms used throughout the paper. The remainder of the paper is organized as follows: In Sect. 2, the literature is reviewed. In Sect. 3, we present the system model and key assumptions. In Sect. 4, we present our game-theoretic formulation of the security-aware wireless edge caching problem. Section 5 describes our algorithm for computing the content placement strategies of the SBSs. In Sect. 6, we evaluate the performance of the proposed scheme. Finally, the paper concludes in Sect. 7.

2 Related work

In this section, we review the related studies on security-aware content placement which address several threats to wireless edge caching. Figure 1 shows the taxonomy of the cache security threats:

- *Cache eavesdropping*: Zhou et al. [9] have used the artificial noise and developed a rate splitting-based physical-layer security approach for cache-enabled cloud radio access networks, without costing extra transmit power for artificial noise. Zahed et al. [10] have investigated the allocation of the security services to the offloaded tasks and have formulated this problem as an optimization program to minimize both the energy consumption and probable security damage. They have also designed a heuristic algorithm to approximate the optimal solution in polynomial time. Ochia et al. [11] have used the stochastic geometry framework to analyze the secrecy rate performance of a cellular network. By assuming random file sizes, they have proposed a caching method that is aware of memory and file size to improve the secrecy rate performance in the presence of colluding eavesdroppers.

Authors in [12] have used encryption methods to maintain security. Wang et al. have proposed a coded caching method in which the central server generates random keys to encrypt the broadcast signals for defending against an eavesdropper who may know part of the cached file before the delivery phase. In [13], Xia et al. have adopted Wynar's encoding method to ensure the security of communication. In [14], Fazel et al. have considered an ultra-dense heterogeneous network consisting of cache-enabled devices in the presence of nonlegitimate eavesdroppers. In order to maximize the sum of secure cache throughput by optimizing the cache placement probability of contents, expressions for the signal to interference plus noise ratio, the probability of secure and successful transmission, achievable secrecy rate, and the sum of the secure cache throughput are derived. Then, they presented an iterative algorithm to find a near-optimal solution for the cache throughput of devices.

- *Content tampering attacks*: Xu et al. [15] have constructed the cache pollution attack model based on observations of attacks, considering the number of unpopular contents, malicious users, and the attack intensity. By characterizing the state of the edge node in terms of cache missing rate and request rate, they have used hidden markov models to detect the attack in cache-enabled mobile social networks. They extended their work and used the notion of block-chains for recording the interactions between the MUs and edge nodes, whereby no entities can modify and deny the caching service information [16]. Cui et al. [17] have used a federated learning technique to encourage edge devices to cooperate in training data and have also used blockchain technology to securely transmit the data of internet of things devices.

Sun et al. [18] have proposed a framework based on blockchain to protect the privacy of users and the security of historical data. They have decomposed the optimization problem into two parts to jointly optimize the probability of caching and the rate of redundancy to maximize the probability of secure transmission. Wang et al. [19] have leveraged the blockchain network as a platform for verification between internet of vehicles and edge devices, exploiting off-chain integrity and on-chain hash methods for recording the evidence of interaction between both ends. Additionally, integrated with identity-based blind signature technology, they have designed a security method in which internet of vehicles nodes request services from edge devices in an

anonymous way. Li et al. [20] also used blockchain technology to guarantee security data caching and prevent the tampering of cache data. In order to optimize the data caching placement, they used a quantum particle swarm optimization algorithm to solve the problem with the greatest content caching gain.

- *Device-to-device (D2D) caching and uncooperative behavior*: In [21], Yang et al. proposed a blockchain incentive method to encourage the users to cache contents. They modeled the interaction between the edge computing server and the users as a Stackelberg game in which the server uses the mining profits to reward the users. After the successful implementation of the cache strategy, users are rewarded by the system, which is a new block mining process. In [22] auction mechanisms are exploited to drive edge devices to cooperate in caching contents. Xiong et al. have also adopted a reverse auction in which an intelligent routing relay acts as the purchaser of the wireless services and users are suppliers and consumers. In their strategy, a popularity-based greedy algorithm is applied in the selection of winning bids. In [23], Mobile network operators encourage content providers to store their most popular content in edge caches, with the aim to earn more monetary gain. Alioua et al. modeled the interaction between operators and providers using a multi-leader multi-follower Stackelberg game with two non-cooperative sub-games to model the competition between the operators and the conflict between providers, respectively.

Weifeng et al. [24] proposed a multi-dimensional trust evaluation mechanism between mobile users to select reliable users as partners for caching. In order to motivate users to cache contents for other devices, they introduced a cooperative caching game in which the trust relations and physical distance between two users are considered to formulate the cost function. In [25], a blockchain-based consensus protocol for a D2D network is proposed, where the blockchain acts as a trusted third party to maintain transactions between users. Rocha et al. [26] combined direct and indirect trust for assessing of D2D nodes in content caching. They have designed a blockchain-inspired security framework to coordinate and audit evidence of indirect behavior in a secure way. Their collaborative trust model aims to mitigate the transmission of invalid content, through the collection of indirect and direct observations.

- *Denial of service and cache pollution attack*: Gabry et al. [27] derived the achievable average backhaul rate and investigated the system caching performance from a game-theoretic perspective. They studied the pure-strategy Stackelberg game between the macro-cell base station and the malicious users. Jalalpour et al. [28] have designed a security system to dynamically deploy security service function chains. Their system reacts to suspicious traffic by instantiating and reconfiguring customized security services that are realized by security chains consisting of one or multiple virtual security functions. Natalino et al. [29] have formulated the problem to maximize the robustness of targeted link cuts. The model detects critical links, which if removed from the network, disconnect the greatest number of users from the file. Al-share et al. [30] simulated a collusive interest flooding attack and analyzed its effects on named data networks. They proposed a technique utilizing the non-parametric cumulative sum algorithm to detect malicious abnormalities in behavior and discard attacking interest packets.

Wen et al. in [31] have investigated the impact of the jammers on the secrecy performance of the caching system. They have proposed a secure random caching method to optimize the caching distribution of the contents and to maximize the average reliable transmission probability. Xie et al. [32] have used time-to-live approximation to study the attack resilience of some state-of-the-art policies and derived formulas to investigate the strategy of optimal attack under a constant total attack rate and its effect on the cache performance for legal requests. In [33] the blockchain technology is employed to prevent malicious packets from leaking into the peer-to-peer (P2P) network. In the proposed scheme, the hash values of packets are inserted into the blocks and as each peer has the blocks, they can easily check the authenticity of network packets. Similar ideas have been employed to combat DoS attacks in traditional P2P networks [34]. Wang et al. [35] designed a cache pollution attack detection and defense mechanism based on the change of request pattern is, which helps an edge node to detect cache pollution attacks by analyzing multiple content request indicators and collaboration between edge nodes.

Our addressed threat in Sect. 3 is also within the category of denial of service and cache pollution attack. Table 2 summarizes the most relevant schemes from prior work.

3 System model

In this section, we describe the model of a cache-enabled small cell network and elaborate on the key assumptions for the content placement problem.

3.1 Network model

We consider a small cell network (shown in Fig. 2) in which an MBS and N SBSs, deployed in the coverage area of the MBS to act as relays, serve the requests of M MUs. The sets of SBSs and MUs are respectively indicated by $\mathcal{N} = \{1, 2, 3, \ldots, N\}$ and $\mathcal{M} = \{1, 2, 3, \ldots, M\}$. The

Table 2 Comparison of related work

References	Security threat	Network type	Defence mechanism	Solution approach	Overhead	Complexity
[9]	Cache eavesdropping	Cloud radio access networks	Artificial noise	Centralized	Communication overhead	Polynomial
[10]	Cache eavesdropping	Internet of Things	Cryptographic algorithms	Distributed	Computation overhead	Polynomial
[11]	Cache eavesdropping	Millimeter wave cellular networks	Stochastic geometry	Centralized	Computation overhead	Linear
[12]	Cache eavesdropping	Content-centric wireless network	Encryption methods	Centralized	Computation overhead	Exponential
[13]	Cache eavesdropping	Fog computing networks	Wynar's encoding	Centralized	Computation overhead	Exponential
[14]	Cache eavesdropping	Unmanned aerial vehicles	Optimization methods	Distributed	Computation overhead	Polynomial
[15]	Content tampering	Mobile social networks	Analysis of attacking behavior	Distributed	Computation overhead	Exponential
[16]	Content tampering	Mobile cyber physical system	Blockchain technology	Distributed	Signalling overhead	Polynomial
[17]	Content tampering	Internet of Things	Federated learning	Distributed	Communication and computation overhead	Polynomial
[18]	Content tampering	6G of wireless cellular networks	Blockchain and optimization	Centralized	Computation overhead	Exponential
[19]	Content tampering	6G and Internet of Vehicles	Blockchain and hash methods	Centralized	Signalling overhead	Polynomial
[20]	Content tampering	Edge cloud environment	Blockchain technology	Centralized	Computation overhead	Polynomial
[21]	Uncooperative behavior	Device-to-Device communication	Stackelberg game with mining profit	Distributed	Computation overhead	Polynomial
[22]	Uncooperative behavior	Device-to-Device networks	Auction mechanisms	Distributed	Computation overhead	Linear
[23]	Uncooperative behavior	5G-enabled IoT network	Earning monetary gain to users	Distributed	Computation overhead	Polynomial
[24]	Uncooperative behavior	Device-to-Device communications	Physical layer mechanisms	Distributed	Communication overhead	Exponential
[25]	Uncooperative behavior	Device-to-Device communications	Blockchain and deep learning	Centralized	Computation overhead	Polynomial
[26]	Uncooperative behavior	Device-to-Device communications	Dempster Shafer Theory	Distributed	Computation overhead	Polynomial
[27]	Denial of Service	Small cell networks	Stackelberg game	Centralized	Computation overhead	Linear
[28]	Denial of Service	Content delivery network	Virtualized security chains	Centralized	Computation overhead	Polynomial
[29]	Denial of Service	Content delivery networks	Heuristic method	Centralized	Computation overhead	Exponential
[30]	Denial of Service	Named Data Networking	Analysis of attacking behavior	Centralized	Computation overhead	Polynomial
[31]	Denial of Service	Wireless networks	Optimization methods	Centralized	Computation overhead	Polynomial
[32]	Denial of Service	Software defined networking	Analysis of attacking behavior	Centralized	Computation overhead	Exponential

Table 2 (continued)

References	Security threat	Network type	Defence mechanism	Solution approach	Overhead	Complexity
[33]	Denial of Service	Device-to-Device communications	Blockchain technology	Distributed	Communication overhead	Polynomial
[34]	Denial of Service	Device-to-Device communications	Analysis of Chord network	Distributed	Communication overhead	Logarithmic
[35]	Cache Pollution	Cloud computing networks	Analysis of request pattern	Centralized	Communication and computation overhead	Linear
Proposed algorithm	Fictitious Request Generation	Small cell networks	Stackelberg game	Distributed	Limited Signalling overhead	Polynomial

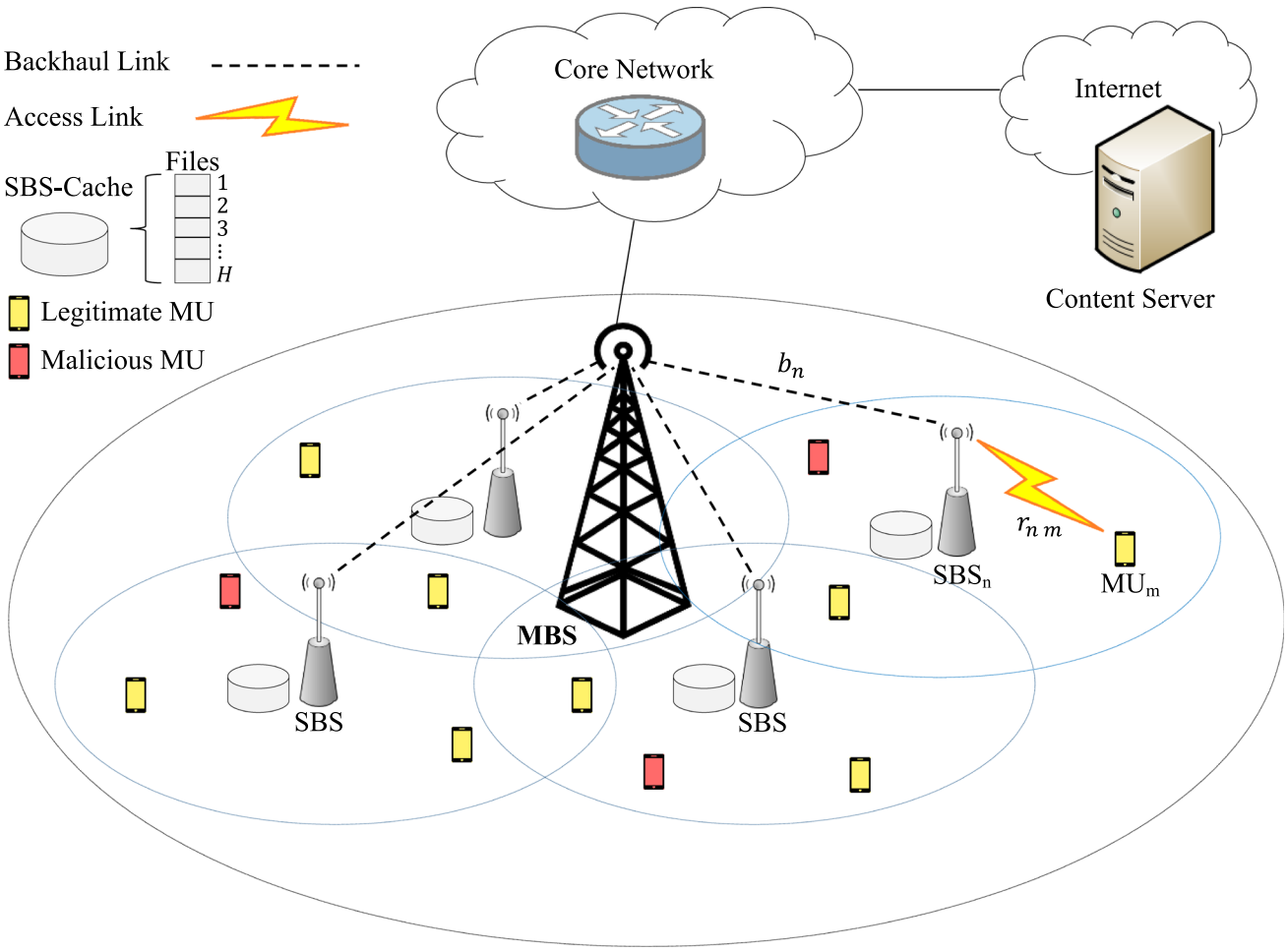

Fig. 2 System model

SBSs are cache-enabled and connected to the MBS with backhaul links with limited capacity b_n for SBS n. In general, SBSs have overlapping coverage areas, and hence one of many potential SBSs can serve MUs. Each timeslot is split into two intervals: an interval for receiving users' requests and the other interval for cache replacement. In the user request interval, users located within the range of an SBS send their content request. If the content exists in the SBS cache, it will be readily downloaded without inflicting any load on the backhaul links; otherwise, the request will be redirected to the MBS. On the other hand, during the cache replacement interval (which is assumed to be of negligible duration), the cache controller module installed in each SBS

may refresh the cached content according to the history of content requests.

3.2 Malicious users' model

It is assumed that there exist $M_{ml} = \beta M$ malicious MUs in the network, with $\beta \in [0, 1]$. The MUs in $\mathcal{M}$ are divided into two sets, $\mathcal{M}_{lg}$ and $\mathcal{M}_{ml}$ representing the set of legitimate and malicious MUs, respectively. The $M_{lg} = (1-\beta)M$ legitimate MUs in $\mathcal{M}_{lg}$ request contents following a known statistical content popularity distribution. On the other hand, the $M_{ml} = \beta M$ MUs in $\mathcal{M}_{ml}$, have been successfully exploited by an attacker (e.g., malware) to request contents according to some adversarial strategy. In particular, their objective is to maximize the congestion of backhaul links by requesting as many non-cached contents as possible. In Fig. 2, the legitimate MUs are shown in yellow and the malicious MUs are represented in red.

3.3 Wireless channel model

The wireless channel capacity between MU m and SBS n is shown as $r_{n,m}$ and can be calculated according to the standard Shannon formula:

$$r_{n,m} = W \log\left(1 + \frac{p_n g_{n,m} d_{n,m}^{-\alpha}}{\sum_{i\in\mathcal{N}, i\neq n} p_i g_{i,m} d_{n,m}^{-\alpha} + \sigma^2}\right) \tag{1}$$

in which W is the spectrum bandwidth and p_n is the transmission power of SBS n. α is the path-loss exponent and σ^2 is the noise power at each MU. $g_{n,m}$ and $d_{n,m}$ are respectively the channel gain and distance between SBS n and MU m.

3.4 Statistical content popularity model

The set of contents is indicated by $\mathcal{F} = \{1, 2, 3, \ldots, F\}$ where F is the number of contents. Each content is assumed to be split into equal size chunks. Also, we divide the contents in $\mathcal{F}$ into groups of H contents. SBSs cache contents based on these groups, and every SBS can cache only one content group (i.e., for simplicity, we assumed that the capacity of cache for every SBS is H contents). Additionally, the total number of content-groups $K = F/H$ without loss of generality is assumed to be an integer, and the set of content groups is denoted by $\mathcal{K} = \{1, 2, 3, \ldots, K\}$.

The Legitimate MUs request contents independently based on their popularity which follows the standard Zipf's law with parameter γ. In particular, for file $f \in \mathcal{F}$, we have:

$$p_f = \frac{1/f^{\gamma}}{\sum_{\mathrm{f}\in\mathcal{F}} 1/\mathrm{f}^{\gamma}} \tag{2}$$

where p_f is content popularity following the well-known Zipfian rank-frequency distribution. In other words, p_f is the fraction of time the f-th most popular file is requested. Consequently, lower indices in the file ranking are dedicated to files having larger popularity. As an additional comment, the simplest case of Zipf's law is a $1/f$ function. In fact, given a set of Zipfian distributed frequencies, sorted from most popular to least popular, the *second* most popular file will happen *half* as many times as the first, the *third* most common frequency will happen *third* as many times as the first, and the f-th most common frequency will happen $1/f$ as many times as the first. As for the role of the exponent γ, larger values of γ cause a steeper distribution because the requests of users are focused on a smaller set of contents (i.e., more queries are concentrated on a set of *hot* contents). Thus, by just caching these hot contents, the SBSs can increase their hit ratio, which reduces the total delay of MUs.

Hence, the popularity of content-group k can be obtained by summing the popularities of contents in this group:

$$P_k = \sum_{f=(k-1)H+1}^{f=kH} p_f \tag{3}$$

3.5 User association criteria

The SBS content placement is denoted by a $N \times K$ matrix I in which $I_{n,k}$ is the element at the n-th row and k-th column getting value from $\{0, 1\}$. $I_{n,k} = 1$ if SBS n caches the content-group k, and 0 otherwise. Every SBS can cache only one content group, thus there is just one non-zero number in every row of I. From SBS n, MU m receives a file in group k with the maximum data rate of $R_{n,m,k}$. It is equal to the wireless capacity between MU m and SBS n if the content is in the cache of SBS n. Otherwise, the backhaul capacity of SBS n also limits this rate. In general, the impressive data rate $R_{n,m,k}$ can be computed as:

$$R_{n,m,k} = r_{n,m} I_{n,k} + \min\{r_{n,m}, b_n\}(1 - I_{n,k}) \tag{4}$$

Given Signal to Interference plus Noise Ratio (SINR) threshold θ, SBSs that can provide an SINR above this threshold to MU m can serve its requests. In other words, the SBSs that can offer an SINR above the aforementioned threshold for serving an MU requests are the neighbor set of it. Also, this MU is a neighbor of those SBSs. The neighboring MUs of SBS n and the neighboring SBSs of MU m are respectively indicated by $\mathcal{M}(n)$ and $\mathcal{N}(m)$. When MU m requests for a file from group k, it connects with the SBS offering the highest rate to it. Hence, the criterion of user association is

stated by:

$$n^* = \arg\max_{n \in \mathcal{N}(m)} R_{n,m,k} \tag{5}$$

3.6 Users' delay model

MU m downloads content from group k with a delay $D_{m,k}$ calculated by:

$$D_{m,k} = L / R_{n^*,m,k} \tag{6}$$

where L is the size of a content. Therefore, the expected value of $D_{m,k}$ (w.r.t. content popularity distribution) is given by:

$$D_m = \mathop{\mathbb{E}}_{k}\big[D_{m,k}\big] = \sum_{k \in \mathcal{K}} P_k \times D_{m,k} \tag{7}$$

3.7 Backhaul congestion model

The attacker has control of the malicious MUs and determines for them from which content group they have to request. The strategy of the attacker is denoted by a vector $\boldsymbol{s}$ of size M_{ml}, as shown in (8) in which s_m is the determined content-group for malicious MU m to request from it.

$$\mathbf{s} = \big(s_1, s_2, s_3, \ldots, s_{M_{\text{ml}}}\big), \quad s_m \in \mathcal{K}, \tag{8}$$

The backhaul congestion is the amount of time that the backhaul links are occupied for transmitting the requested non-cached contents to the associated SBSs. Non-cached contents can be requested by legitimate or malicious MUs. The overall congestion in the backhaul links of the network-wide caching system is denoted by T and defined as:

$$T = \sum_{n \in \mathcal{N}} \Bigg[\sum_{m \in \mathcal{M}_{lg}(n)} \sum_{k \in \mathcal{K}} P_k\big(1 - I_{n,k}\big) \times L/b_n + \sum_{m \in \mathcal{M}_{ml}(n)} \sum_{k \in \mathcal{K}} \delta(s_m, k)\big(1 - I_{n,k}\big) \times L/b_n \Bigg] \tag{9}$$

in which $\delta(x, y)$ is the discrete delta function defined by:

$$\delta(x, y) = \begin{cases} 1, & x = y \\ 0, & x \neq y \end{cases} \tag{10}$$

In (9), the first term in the square bracket is the expected duration of time in which the backhaul link associated with SBS n becomes congested due to the cache misses by all legitimate MUs' requests. These requests are made according to the known content popularity P_k [c.f., Eq. (3)]. On the other hand, the second term is the expected congestion caused by malicious MUs' content requests. These requests are fictitiously made according to some adversarial strategy $\boldsymbol{s}$ (which is determined by the attacker).

Table 3 summarizes the notations used in our system model.

4 Problem formulation

In this section, we model the competitive interaction between the SBSs and the attacker using a game-theoretic framework. Every SBS wants to minimize the backhaul congestion as well as the downloading delay of contents for its neighboring users, while the attacker wants to maximize the congestion in backhaul links. We use the notion of the Stackelberg game to describe this setting:

Definition (*Two-stage multi-leader single-follower stackelberg game*) [39] A Stackelberg game between the SBSs and the attacker is defined by the triplet $\mathcal{G}_s = \Big\{\{\mathcal{N}, J\}, \{\mathcal{A}, \mathcal{S}\}, \Big\{\{c_n\}_{n \in \mathcal{N}}, c_0^{'}\Big\}\Big\}$, which can be elaborated as follows:

- *Leaders*: the set of SBSs $\mathcal{N}$ are the leaders of the game $\mathcal{G}_s$. Leaders make the first move in the sequential play.
- *Leader's strategy set*: each SBS has the strategy space $\mathcal{K}$. That is, each SBS can choose among the set of content groups to cache in its storage. A choice made by SBS n is denoted by: $a_n \in \mathcal{K}$. We use the standard notation $\mathcal{A} = \times_{n \in \mathcal{N}} \mathcal{K}$ to represent the entire space of the joint strategies of the SBSs.
- *Follower*: THE game's follower is the attacker (or the malicious entity), which makes the second move in the sequential play.
- *Follower's strategy set*: the attacker chooses the malicious MUs' requesting profile from the space $\mathcal{S} = \times_{m \in \mathcal{M}_{ml}} \mathcal{K}$. In fact, $\boldsymbol{s} \in \mathcal{S}$ shows the actions of MUs for requesting files of content groups from the set $\mathcal{K}$.
- *Leader's cost function*: The objective of each SBS is to minimize the congestion duration of the backhaul links and the average delay of the users within its coverage range. Associated with each SBS is a cost function of the form: $c_n : \mathcal{A} \times \mathcal{S} \to \mathbb{R}$, which is defined as:

$$c_n(a_n, \boldsymbol{a}_{-n}, \boldsymbol{s}) = T + \sum_{m \in \mathcal{M}(n)} D_m, \ \forall n \in \mathcal{N} \tag{11}$$

in which $a_n \in \mathcal{K}$ is the strategy selected by SBS n, $\boldsymbol{a}_{-n}$ is the strategy vector of all SBSs other than n, and $\boldsymbol{s} \in \mathcal{S}$ is the attacker's strategy. T is the amount of time that the backhaul links are occupied for transmitting the requested non-cached contents to the associated SBSs, as defined in (9). D_m is the

Table 3 Summary of notations used in the system model

Description	Notation
Set of mobile users (MUs)	$\mathcal{M}$
Set of legitimate MUs	$\mathcal{M}_{lg}$
Set of malicious MUs	$\mathcal{M}_{ml}$
The number of MUs	M
The number of legitimate MUs	M_{lg}
The number of malicious MUs	M_{ml}
Set of small base stations (SBSs)	$\mathcal{N}$
The number of SBSs	N
The capacity of cache in every SBS	H
Wireless capacity between SBS n and MU m	$r_{n,m}$
Bandwidth	W
Transmit power of SBS n	p_n
Path-loss exponent	α
Noise power	σ^2
Channel gain between SBS n and MU m	$g_{n,m}$
Distance between SBS n and MU m	$d_{n,m}$
Backhaul capacity of SBS n	b_n
Set of contents	$\mathcal{F}$
Matrix of SBS-content placement	I
The number of contents	F
Set of content-group numbers	$\mathcal{K}$
The number of content-groups	K
The popularity of content f	p_f
The popularity of content-group k	P_k
SINR threshold	θ
Set of neighboring SBSs of MU m	$\mathcal{N}(m)$
Set of neighboring MUs of SBS n	$\mathcal{M}(n)$
Fraction of malicious MUs	β
Set of neighboring legitimate MUs of SBS n	$\mathcal{M}_{lg}(n)$
Set of neighboring malicious MUs of SBS n	$\mathcal{M}_{ml}(n)$
Content-group cached by SBS n	a_n
Maximum data rate between SBS n and MU m when requesting content from group k	$R_{n,m,k}$
Discrete delta function	δ
Delay of MU m for downloading content from group k	$D_{m,k}$
Size of a content	L
Expected delay for MU m	D_m
Vector of content-group numbers determined by the attacker for malicious MUs	$\boldsymbol{s}$
Content-group number determined by the attacker for malicious MU m	s_m
Congestion of backhaul links	T

expected delay of downloading contents by MU m as defined in (7). The second term in the right side of (11) is the sum of expected delays for neighboring MUs of SBS n, $\mathcal{M}(n)$.

- *Follower's cost function*: The objective of the attacker is to maximize the congestion in backhaul links by requesting as many non-cached contents as possible. The attacker J has a cost function of the form $c_0^{'}: \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ which is defined as:

$$c_0^{'}(\boldsymbol{a}, \boldsymbol{s}) = -\sum_{n\in\mathcal{N}} \sum_{m\in\mathcal{M}_{ml}(n)} \sum_{k\in\mathcal{K}} \delta(s_m, k) \times \left(1 - I_{n,k}\right) \times L/b_n \tag{12}$$

where $\boldsymbol{a} = (a_n, \boldsymbol{a}_{-n}) \in \mathcal{A}$ is a strategy profile of all SBSs, and $\boldsymbol{s} \in \mathcal{S}$ is the strategy profile of all malicious MUs (as determined by the attacker). $\delta(s_m, k)$, defined in (10), is the standard Kronecker delta function which is 1 if its variables are equal, and 0 otherwise. It basically indicates whether content k is requested as part of the strategy s_m of the attacker determined for malicious user m. When the term $\left(1 - I_{n,k}\right)$ is 1, it indicates that the requested content k is not cached at SBS n. L and b_n are respectively the content size and the backhaul capacity of SBS n, so division of them gives the duration of congestion in the backhaul link n. Therefore, the summations in (12) calculates the amount of congestion duration in backhaul links arising from transmitting non-cached contents requested by neighboring malicious users. The negative sign in the attacker's cost $c_0^{'}$ is to transform the maximization problem into minimization. More specifically, the objectives of the SBSs and the attacker are as follows:

$$\min_{a_n\in\mathcal{K}} c_n(a_n, \boldsymbol{a}_{-n}, \boldsymbol{s}),\ n \in \mathcal{N}$$

$$\min_{\boldsymbol{s}\in\mathcal{S}} c_0^{'}(\boldsymbol{a}, \boldsymbol{s}) \tag{13}$$

Now, the Stackelberg Equilibrium (SE) is a pair of strategy profiles $\{\boldsymbol{a}^*, \boldsymbol{s}^*\}$ that satisfy the following conditions:

$$c_n\left(a_n^*, \boldsymbol{a}_{-n}^*, \boldsymbol{s}^*\right) \leq c_n\left(a_n^{'}, \boldsymbol{a}_{-n}^*, \boldsymbol{s}^{\prime}\right),\ \forall n \in \mathcal{N}$$

$$c_0^{'}\left(\boldsymbol{a}^*, \boldsymbol{s}^*\right) \leq c_0^{'}\left(\boldsymbol{a}^*, \boldsymbol{s}\right) \tag{14}$$

where $\boldsymbol{s}^*$ is the best reaction of the follower to the strategy $\boldsymbol{a}^*$ of the leaders. When SBS n's strategy changes to $a_n^{'}$, the attacker's strategy also changes to the corresponding $\boldsymbol{s}^{\prime}$. The SE indicates that, at the equilibrium, neither the SBSs nor the attacker will unilaterally deviate due to the resulting increment in cost. Unlike the traditional single leader model, the interaction among multiple leaders affects the Stackelberg equilibrium. Therefore, we need to model the

interaction among the leaders and consider the Nash equilibrium between them.

To prove the existence of SE, we first show that the game between the leaders (SBSs) is an exact potential game:

Definition (*Potential game*) [37] A game $\mathcal{G}_p = \{\mathcal{N}, \mathcal{K}, \{c_n\}_{n\in\mathcal{N}}\}$ with player set $\mathcal{N}$, action set $\mathcal{K}$, and cost functions $\{c_n\}_{n\in\mathcal{N}}$ is called a potential game if there exists a potential function $\Phi : \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ ($\mathcal{A} = \times_{n\in\mathcal{N}}\mathcal{K}$) such that for all $\boldsymbol{a} \in \mathcal{A}$ and $\boldsymbol{s} \in \mathcal{S}$, it holds that $\forall n \in \mathcal{N}$:

$$\begin{aligned}&\Phi\left(a_n^{'}, \boldsymbol{a}_{-n}, \boldsymbol{s'}\right) - \Phi(a_n, \boldsymbol{a}_{-n}, \boldsymbol{s})\\ &= c_n\left(a_n^{'}, \boldsymbol{a}_{-n}, \boldsymbol{s'}\right) - c_n(a_n, \boldsymbol{a}_{-n}, \boldsymbol{s})\end{aligned} \tag{15}$$

Theorem 1 Game $\mathcal{G}_p$, with cost function defined in (11) and potential function (16) (below), is a potential game:

$$\Phi(a_n, \boldsymbol{a}_{-n}, \boldsymbol{s}) = T + \sum_{\mathrm{m}\in\mathcal{M}} D_m \tag{16}$$

Proof. For game $\mathcal{G}_p$, relation (15) can be verified as follows:

$$\begin{aligned}
&\Phi\left(a_n^{'}, \boldsymbol{a}_{-n}, \boldsymbol{s'}\right) - \Phi(a_n, \boldsymbol{a}_{-n}, \boldsymbol{s})\\
&= \left[T\left(a_n^{'}, \boldsymbol{a}_{-n}, \boldsymbol{s'}\right) + \sum_{m\in\mathcal{M}} D_m\left(a_n^{'}, \boldsymbol{a}_{-n}\right)\right] - \left[T(a_n, \boldsymbol{a}_{-n}, \boldsymbol{s}) + \sum_{m\in\mathcal{M}} D_m(a_n, \boldsymbol{a}_{-n})\right]\\
&= \left[T\left(a_n^{'}, \boldsymbol{a}_{-n}, \boldsymbol{s'}\right) + \sum_{m\in\mathcal{M}(n)} D_m\left(a_n^{'}, \boldsymbol{a}_{-n}\right) + \sum_{m\in(\mathcal{M}-\mathcal{M}(n))} D_m\left(a_n^{'}, \boldsymbol{a}_{-n}\right)\right]\\
&\quad - \left[T(a_n, \boldsymbol{a}_{-n}, \boldsymbol{s}) + \sum_{m\in\mathcal{M}(n)} D_m(a_n, \boldsymbol{a}_{-n}) + \sum_{m\in(\mathcal{M}-\mathcal{M}(n))} D_m(a_n, \boldsymbol{a}_{-n})\right]\\
&= \left[T\left(a_n^{'}, \boldsymbol{a}_{-n}, \boldsymbol{s'}\right) + \sum_{m\in\mathcal{M}(n)} D_m\left(a_n^{'}, \boldsymbol{a}_{-n}\right)\right] - \left[T(a_n, \boldsymbol{a}_{-n}, \boldsymbol{s}) + \sum_{m\in\mathcal{M}(n)} D_m(a_n, \boldsymbol{a}_{-n})\right]\\
&= c_n\left(a_n^{'}, \boldsymbol{a}_{-n}, \boldsymbol{s'}\right) - c_n(a_n, \boldsymbol{a}_{-n}, \boldsymbol{s}).
\end{aligned} \tag{17}$$

■

It is noticeable that for the changed strategy $a_n^{'}$, the attacker also changes its strategy from $\boldsymbol{s}$ to $\boldsymbol{s'}$ (to minimize its own cost), but this change does not affect the reasoning. Hence, the above problem is an instance of a potential game and there exists at least one pure NE. As the attacker's every strategy minimizes its own cost when the SBSs reach the NE point, the attacker would also stick to its strategy, i.e., the attacker cannot reduce its cost any further (through unilaterally changing its strategy). Thus, the system's Stackelberg equilibrium is also guaranteed.

The attacker's cost function $c_0^{'}$ as defined in (12) can be easily decomposed into M_{ml} sub-functions $c_m^{'}$, $m \in \mathcal{M}_{ml}$:

$$c_m^{'}(a_n, s_m) = -\sum_{k\in\mathcal{K}} \delta(s_m, k) \times \left(1 - I_{n,k}\right) \times L/b_n \tag{18}$$

Hence, the attacker cost function $c_0^{'}$ is the sum of the malicious users' costs:

$$c_0^{'}(\boldsymbol{a}, \boldsymbol{s}) = \sum_{n\in\mathcal{N}} \sum_{m\in\mathcal{M}_{ml}(n)} c_m^{'}(a_n, s_m) \tag{19}$$

Therefore, when every malicious user minimizes its cost, then the attacker's cost is also minimized.

Note that each malicious user m needs to know a_n to minimize $c_m^{'}$ i.e., every malicious user needs to know whether its requested content has been cached by the associated SBS in the previous iteration. In practice, the malicious user can check the response header (e.g., x-cache in HTTP protocol meta-data) to see if its request has been served from the SBS cache or not.

5 The proposed content placement algorithm

Based on the proof given for Theorem 1, we may employ a modified version of the well-known Best Response Dynamics (BRD) algorithm [38] to compute the Stackelberg equilibrium of the content caching game $\mathcal{G}_p$. Since the content-group selection game among SBSs is proved to be an exact potential game, it also follows the finite improvement property, which guarantees that a best response approach will converge to the NE. As the existence of SE was guaranteed based on the

existence of NE, the content-group selection algorithm also converges to the SE points. Therefore, a content-group selection algorithm based on best response dynamics has been proposed to compute the game's equilibirum.

The pseudo-code is shown in Algorithm 1. BRD is an iterative sequential process in which at every iteration, one SBS optimizes its own strategy; that is, the acting player chooses the strategy minimizing its cost given the most recent play by other players. Within the same iteration, the malicious entity may also react immediately to update its request generation strategy. This procedure is repeated until the strategy profile does not change anymore (See line 12). This BRD process will converge to NE due to standard arguments in the theory of potential games [37]. The formation of NE between the SBS players together with the best response of the attacker results in the emergence of SE in the entire game.

Algorithm 1 Pseudo-code of finding the equilibrium point of the content placement game

Input: Set of contents $\mathcal{F}$, Set of content groups $\mathcal{K}$, and list of MUs under coverage $\mathcal{M}(n)$ for each SBS n
Output: $\boldsymbol{a}$ // the profile of content groups for caching

// **Initialization**
1: $t = 1$ (Initial iteration)
2: Initialize the SBS decision profile $\boldsymbol{a}(1)$ arbitrarily;
3: $\boldsymbol{s}(1) \leftarrow \arg\min_{\boldsymbol{s}\in\mathcal{S}} c_0'(\boldsymbol{a}(1), \boldsymbol{s})$; // the malicious entity reacts to initial SBS decisions
4: Compute $c_n(a_n, \boldsymbol{a}_{-n}, \boldsymbol{s}^*)$ using (11) for each SBS $n \in \mathcal{N}$;

Begin
5: **repeat**
6: SBS n is chosen by a scheduler to update its action.
7: // SBS n selects the best content-group that minimizes its cost function
8: $a_n(t+1) \leftarrow \arg\min_{a_n'\in\mathcal{K}} c_n(a_n', \boldsymbol{a}_{-n}(t), \boldsymbol{s}')$;
9: // The attacker selects the best content groups for the malicious users to forge a request
10: $\boldsymbol{s}(t+1) \leftarrow \arg\min_{\boldsymbol{s}\in\mathcal{S}} c_0'(\boldsymbol{a}(t+1), \boldsymbol{s})$ where $\boldsymbol{a}(t+1) = (a_n(t+1), \boldsymbol{a}_{-n}(t))$;
11: $t \leftarrow t+1$;
12: **until** $\boldsymbol{a}(t-1) \neq \boldsymbol{a}(t)$ **and** $\boldsymbol{s}(t+1) \neq \boldsymbol{s}(t)$
13: **return** $\boldsymbol{a}$;
End

To discuss the computational complexity of the algorithm, we note that Algorithm 1 can be executed as a distributed routine by the SBSs in the system. At each iteration, some SBS n is chosen by the sequential scheduler to update its action. The only costly operation is in line 8 where the SBS has to search for the best content group (among the K available groups) that minimizes its cost function. Hence, the per-iteration computational complexity of each SBS is linear in terms of the number of content groups, i.e., $O(K)$. As for the attacker, it should determine the action profile of all N malicious users under its control. Hence, in line 10, the attacker has to search over an exponential space of $O(K^N)$ to maximize the backhaul congestion. However, given that the attacker's cost function c_0' is additive and separable [c.f., Eqs. (18) and (19)], line 10 can also be executed in a distributed and parallel fashion by each malicious user, i.e., we may replace line 10 by $s_m^* \leftarrow \arg\min_{s_m\in\mathcal{K}} c_m'(s_m, a_n)$ for each malicious user $m \in \mathcal{M}_{ml}$. This makes the computational complexity of the attack linear as well, i.e., $O(K \times |\mathcal{M}_{ml}|)$.

6 Numerical results

In this section, we implement the proposed algorithm in a simulation environment corresponding to the scenario described in Fig. 2. First, we describe the simulation setup which includes the simulation parameters and experiment settings in Sect. 6.1. Then, in Sect. 6.2, we introduce the previous schemes used for comparison. In Sect. 6.3, we show the convergence of the proposed algorithm. Furthermore, in Sect. 6.4, we studied the impact of the some parameters including the fraction of malicious users, content popularities, the number of SBSs, the number of MUs, the number of contents, and backhaul capacities on the average downloading delay of the legitimate users as well as on the congestion of the backhaul links. However, in large scale scenarios in which the number of contents and malicious users increase, we have limitation in servicing MUs. When the number of malicious users increases, the strategy space of the attacker grows exponentially and serving MUs is out of the network capacity.

6.1 Simulation parameters

We consider a scenario in that MUs and SBSs are uniformly distributed in a 500 × 500 m area. The channel gains are set as identically and independently distributed exponential variables with mean 1. The minimum capacity of a wireless link is denoted by $r_{min} = W \log(1+\theta)$, so we set the backhaul

Table 4 Simulation parameters

Description	Parameter	Value
Number of SBSs	N	Default: 10 (varies between 4 and 16 in the experiments)
Number of MUs	M	Default: 20 (varies between 10 and 40 in the experiments)
Number of contents	F	Default: 100 (varies between 100 and 220 in the experiments)
Cache size	H	20 contents
Number of content-groups	K	5
Content size	L	10^9 bits
Bandwidth	W	10^7 Hz
The transmission power of SBS n	p_n	30 dBm
Path-loss exponent	α	4
Noise power	σ^2	10^{-10} W
SINR threshold	θ	0.1
Zipf parameter	γ	0.5
Parameter of backhaul capacity range	ω	0.4
Fraction of malicious MUs	β	0.3

capacities b_n as random variables uniformly distributed in $[\omega \times r_{min}, r_{min}]$. The other simulation parameters are given in Table 4. Similar values for the number of MUs, SBSs as well as the content files have been assumed in other studies on content caching in small cell networks (e.g., [42–44]).

6.2 Comparison with previous schemes

In this section, we show the performance of the proposed algorithm by comparing its results against [40] in which there is no mechanism to combat the attacker as well as [35] exploting the Attack-aware Cache Defense (ACD) algorithm. In [40], Yang et al. have not considered any malicious users in the system and modeled the interaction among SBSs as a potential game. Each SBS responds to the content requests from its neighboring MUs with the objective of minimizing their downloading delay. The authors have proposed a best response dynamic algorithm in which at each iteration, every SBS caches the content that minimizes the average delay of neighboring users, whether legitimate or malicious.

In [35], the authors have proposed a proactive mechanism called ACD algorithm that gracefully reduces the cache probability of unpopular contents. They have utilized the idea of a seminal article (CacheShield [41]) for content centric networking routers which makes the caching decisions robust against cache pollution attacks. CacheShield employs a so-called "shielding function" to identify relatively popular contents and filter out unpopular ones. The basic idea is to use a parameterized logistic function to form a caching probability for each requested content object. If the function returns true, the corresponding content object is cached. If the function returns false, only the name of the returned content object and a counter are stored in the cache as a placeholder. If the same (still not cached) content object is requested again, the corresponding counter is increased and the shielding function is re-evaluated on the updated counter. Before the attack occurs, the algorithm detects the average number of content requests in the first T timeslots. Then, when the attack occurs, the algorithm reduces the cache probabilities of unpopular contents.

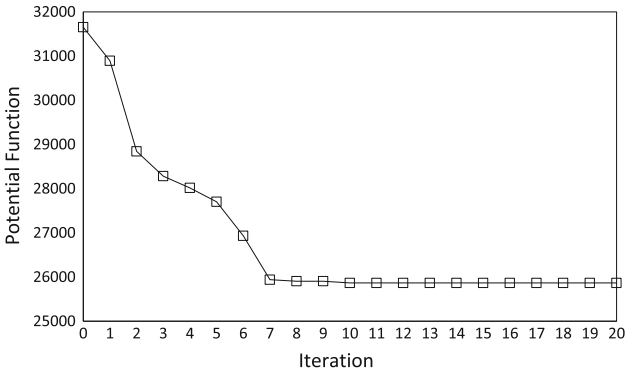

Fig. 3 Convergence of the potential function

6.3 Investigating the convergence property

First, we investigate the convergence property of the proposed algorithm. Figure 3 shows the evolution of the potential function of the content caching game across the iterations of Algorithm 1. As can be seen, the play converges relatively fast towards equilibrium. In Fig. 4, we have also plotted the moving average of the duration of congestion in the backhaul links. In this figure, we compare our result with the performance of a content placement algorithm proposed in [40] that does nothing to counteract the impact of fictitious requests issued by compromised MUs. More specifically, in [40], a game-theoretic model has been proposed in which each SBS minimizes only the average delay of its neighboring MUs assuming simplistically that no foul play is suspected. The standard BRD algorithm is simply employed by the SBSs to find the equilibrium of the game. As expected, the backhaul delay of the proposed algorithm is lower than that of [40]. In this figure, we can see that at first, the backhaul congestion

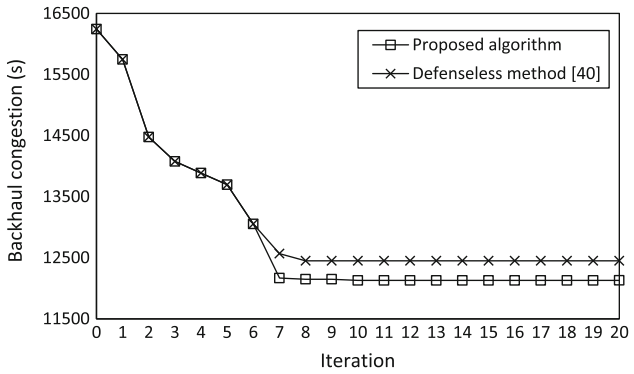

Fig. 4 Convergence of the backhaul congestion time

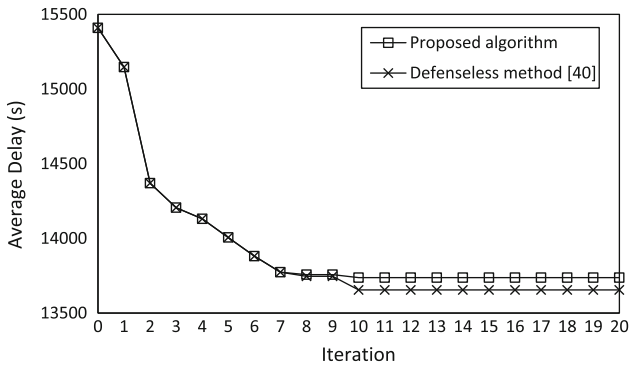

Fig. 5 Convergence of the average delay of MUs

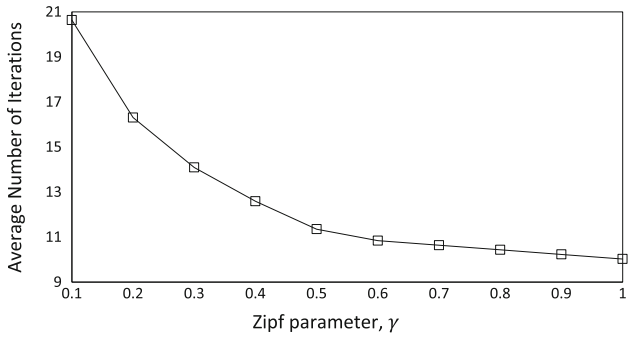

Fig. 6 The average number of iterations versus Zipf parameter

is high, but as SBSs cache the content groups strategically, the congestion of the backhaul links also decreases.

In Fig. 5, we plot the average delay experienced by legitimate users for downloading content. As the proposed algorithm accounts for malicious play and also aims for reducing congestion, we notice a slight increase in the delay compared to [40]. In fact, sometimes the SBSs have to cache less popular content to accommodate the fictitious requests made by the attacker. However, this increase in the delay is less noticeable compared to the reduction in congestion achieved by our algorithm.

In Fig. 6, the average number of iterations for the proposed algorithm is displayed under different values for the Zipf

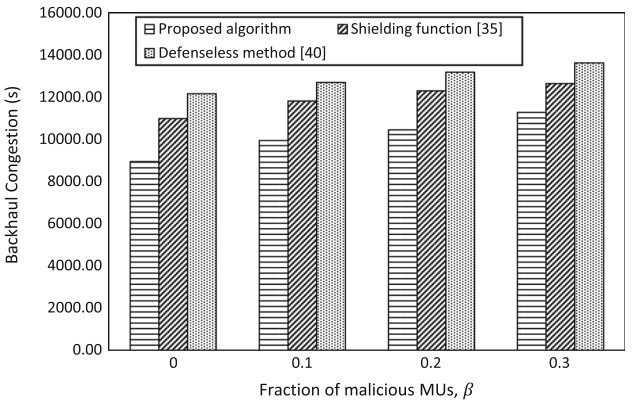

Fig. 7 The backhaul congestion time versus fraction of malicious MUs

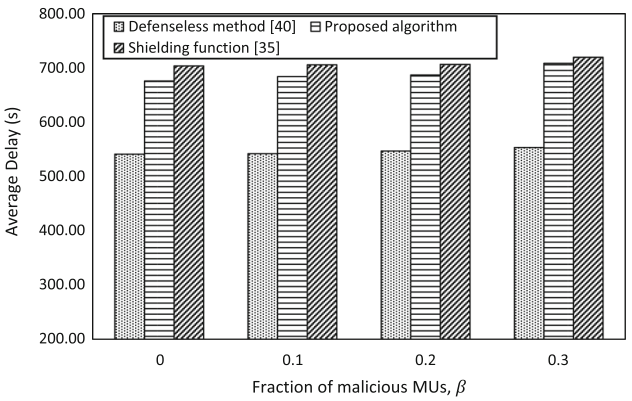

Fig. 8 The average delay of MUs versus fraction of malicious MUs

distribution parameter. In general, a larger Zipf parameter is associated with a steeper distribution such that requests of users are focused on a smaller set of files (i.e., more queries are concentrated on a set of *hot* contents). It can be seen that when content popularity gets more uniform, it takes much longer to converge.

6.4 Performance measurement

6.4.1 The impact of fraction of malicious users

In Fig. 7, we compare the congestion time for the different fractions of the malicious MUs. As expected, the minimum amount of congestion in the backhaul links is for $\beta = 0$, when there are no malicious users in the network. For the greater number of malicious users, the cache miss ratio increases, resulting in more congestion. Consistently across all the results, we notice that the backhaul links will become less congested under our algorithm compared to [35] and [40].

In Fig. 8, we conduct a similar experiment to compare the average delay of the users versus different fractions of the malicious MUs. Similar to Fig. 7, the minimum amount of average delay is for $\beta = 0$, when there are no malicious

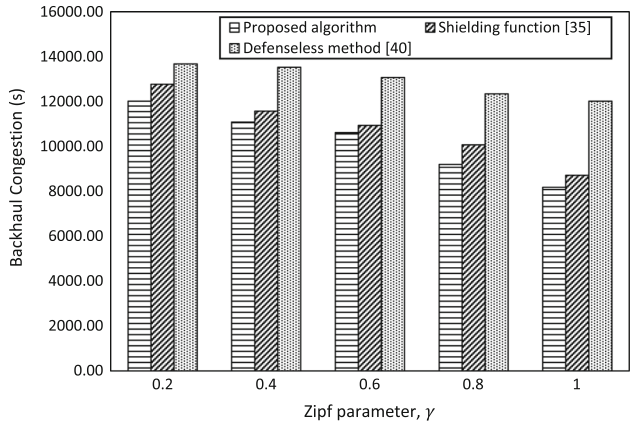

Fig. 9 The backhaul congestion time versus Zipf parameter

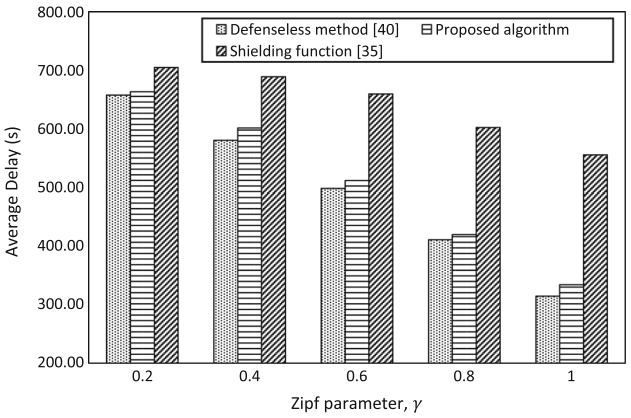

Fig. 10 The average delay of MUs versus Zipf parameter

users and with increase in the fraction, the average delay also increases. For all fraction of malicious users, the average delay for our proposed algorithm is less than that of [35] using shielding function. In Fig. 8, we notice that the delay in our algorithm is more than that in [40], because the SBSs have to sometimes cache less popular content to accommodate the requests made by the attacker. However, this increase in the delay is less noticeable compared to the congestion reduction achieved by our algorithm.

Although for different fraction of malicious users, the proposed algorithm causes at most 21% increase in the average delay compared to the defenseless method [40], its performance in reducing the backhaul congestion is better by at least 26%. In comparison with [35], our proposed algorithm has 15 and 2.8% improvement in the backhaul congestion and the average delay, respectively.

6.4.2 The impact of content popularity

Figure 9 displays the average backhaul congestion time under different Zipf parameter values exploited in content popularity distribution. Under a larger Zipf parameter (and thus a smaller set of popular contents), the SBSs can reduce their miss ratio by just caching hot contents in their limited memory, thereby decreasing the backhaul congestion. In Fig. 10, we conduct a similar experiment to compare the average delay of the users versus different Zipf parameter values. Similar to Fig. 9, with an increase in the Zipf parameter, a fewer number of contents need to be cached to meet the users' requests; hence, the average delay decreases. As the SBSs have to sometimes cache less popular content to accommodate the requests made by the attacker, in Fig. 10 we see that the delay in our algorithm is more than that in [40]. However, this increase in the delay is less noticeable compared to the congestion reduction achieved by our algorithm. The results show that with increase in the Zipf parameter, our proposed algorithm reduces the backhaul congestion by 21% with the

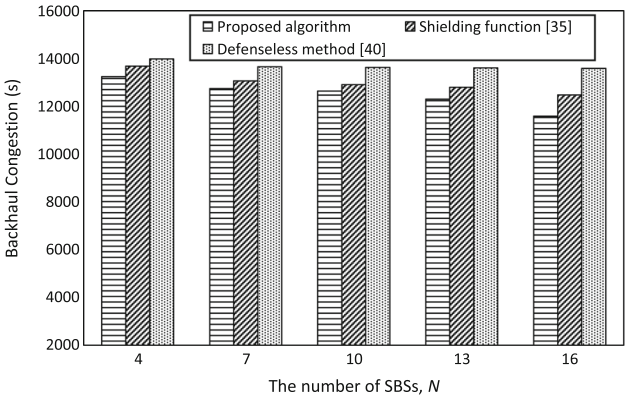

Fig. 11 The backhaul congestion time versus the number of SBSs

cost of a 3% increase in the average delay in comparison with the defenseless method [40]. Compared with the shielding function [35], our performance is 5% better in the backhaul congestion and around 20% better in the average delay.

6.4.3 The impact of the number of SBSs

In Figs. 11 and 12, we investigate how the number of SBSs affects the backhaul congestion as well as the average delay of the legitimate MUs, respectively. In the case of more SBSs, every SBS answers to fewer requests. Furthermore, more files can be cached by all the SBSs, increasing the chance of cache hit. Accordingly, the time of congestion in backhaul links as well as the average delay of content download decreases when using a higher number of SBSs. In Fig. 11, it is noticed that the backhaul links are less congested under our algorithm compared to [40], not reacting to the malicious strategy of the attacker. In Fig. 12, we notice that the delay in our algorithm is a bit more than that in [40], because the SBSs occasionally cache unpopular content to accommodate the fictitious requests made by the attacker. However, this increase in the delay is less noticeable compared to the

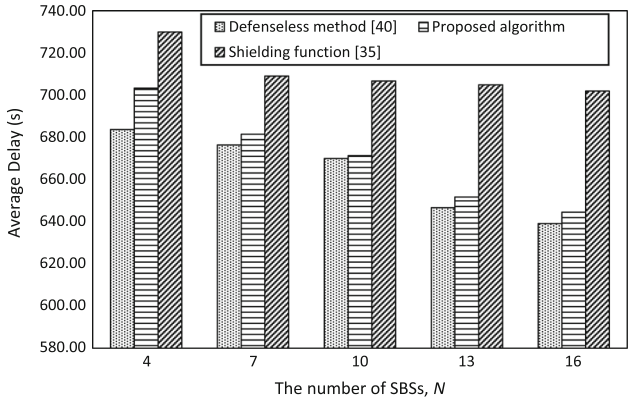

Fig. 12 The average delay of MUs versus the number of SBSs

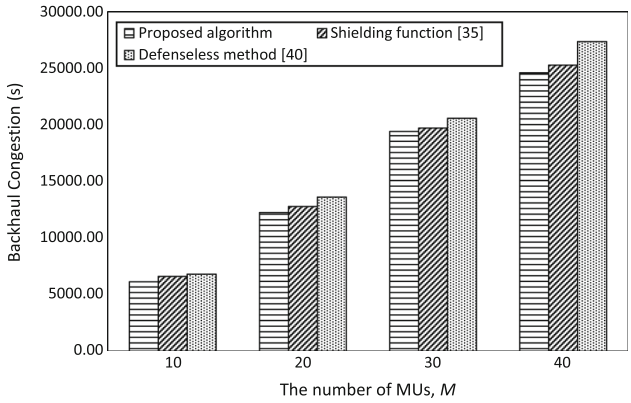

Fig. 13 The backhaul congestion time versus the number of Mus

congestion reduction achieved by our algorithm. Compared with the defenseless method [40], when the number of SBSs increases, the proposed algorithm performs better by 9% in the backhaul congestion, however, it makes the average delay worse by 1%. In comparison with the shielding function [35], we have improvement in the backhaul congestion and average delay by 4 and 6%, respectively. The main reason behind the superiority of our scheme over [35] is that our game-based approach explicitly accounts for the strategic and adaptive misbehavior on the part of the attacker. This is while in [35], the attacker has been assumed to be less sophisticated in the sense that it only learns the statistics of the content popularity and fabricate requests for unpopular contents. In other words, the attacker does not mutually react to the caching decisions made by the system.

6.4.4 The impact of the number of MUs

Figures 13 and 14 respectively show the backhaul congestion time and the average downloading delay versus the number of MUs. In these two figures, it can be observed that there is an upward trend in the backhaul congestion and the

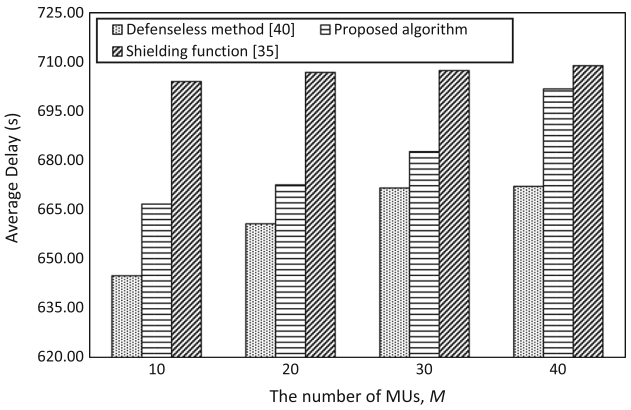

Fig. 14 The average delay of MUs versus the number of MUs

average delay of users when the number of MUs neighboring the SBSs increases. This is mostly because it is difficult to provide the minimum rate requirements for all the MUs who exploit the limited resources. Additionally, with a higher population of MUs, more files are requested, decreasing the probability of cache hit and increasing the backhaul congestion. Congestion of the system slowly increases the load of backhaul traffic deteriorating the achievable impressive data rate (as can be noticeable for more than 20 MUs).

Since the algorithm in [40] does not react to the malicious role of the attacker, we notice less congestion in the backhaul links for our proposed algorithm, as shown in Fig. 13. In Fig. 14, there is a slight increase in the delay compared to [40]. As the proposed algorithm accounts for the attacker role and also aims for reducing congestion, sometimes the SBSs have to cache less popular content in order to combat the attacker. However, this increase in the delay is less important compared to the decrease in congestion achieved by our proposed algorithm. For more MUs, the proposed algorithm has a 9% improvement in the backhaul congestion compared with the defenseless method [40], though it makes the average delay worse by 3%. In comparison with the shielding function [35], our algorithm performs 4% and 3% better in reducing the backhaul congestion and the average delay, respectively.

6.4.5 The impact of network settings

In Figs. 15 and 16, we change the parameter of backhaul capacity from $\omega = 0.2$ to $\omega = 1$ whereas other simulation settings are the same as those given in Table 4. As shown in these two figures, with the increase in ω, the backhaul congestion time and the average downloading delay of contents are reduced. In Figs. 17 and 18, we vary the number of files from 100 to 220 to study its effect on the backhaul congestion and the downloading delay for MUs. With the increase in the number of contents, the number of groups increases as well, but when the capacity of cache at every SBS is limited,

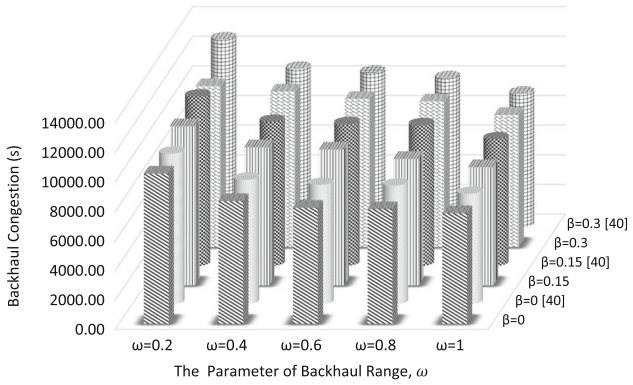

Fig. 15 The backhaul congestion time under different ranges of backhaul capacity

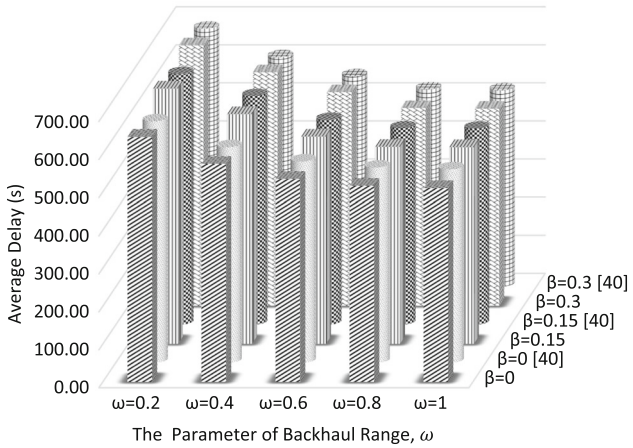

Fig. 16 The average delay of MUs under different ranges of backhaul capacity

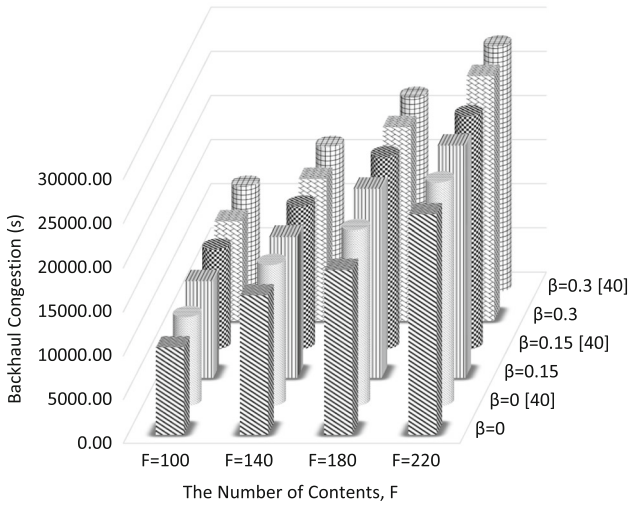

Fig. 17 The backhaul congestion time for a different number of files

it causes to larger cache miss ratio and thus more delay for downloading contents by MUs.

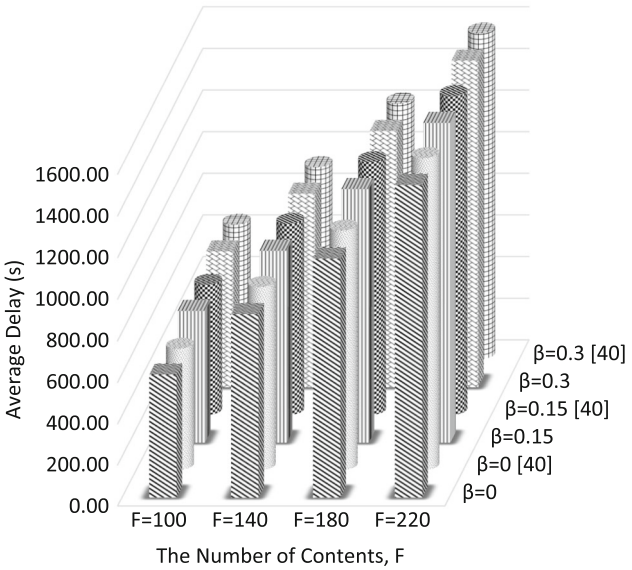

Fig. 18 The average delay of MUs for a different number of files

In Figs. 15 and 17, the congestion in backhaul links for our algorithm is less than that for [40], which does not account for the malicious actions of the attacker. In Figs. 16 and 18, as the proposed algorithm accounts for adversarial play and also aims for reducing congestion, we notice a slight increase in the delay compared to [40].

7 Conclusion

In this paper, we have addressed the problem of content placement in the cache-equipped SBSs at the wireless edge in the presence of an attacker having the control of some users. The compromised users may generate fictitious requests not necessarily in line with the assumed standard popularity distribution. We have modeled the competitive interaction between the SBSs and the attacker using a multi-leader single-follower Stackelberg game formulation. We have then formulated the cooperation among the SBSs as a potential game in which they jointly decide what content to cache so that the average delay of the MUs located within their coverage range, as well as the overall congestion in the backhaul links, is reduced. We have proved that a Stackelberg equilibrium exists for this caching game. Also, to compute the equilibrium, we have proposed a specialized best response dynamics algorithm in which each SBS finds its best action independently. To evaluate the performance of the proposed procedure, we have conducted several experiments measuring how content popularity, the number of SBSs, and the number of MUs affect the caching performance. The obtained results have demonstrated the superiority of the proposed scheme in comparison with another game-theoretic content placement strategy which is inadvertent to the presence of malicious requests. In conclusion, our findings have

highlighted the crucial importance of optimizing content placement in adaptation to adversarial content requests. For future work, we intend to generalize the proposed scheme to a setting where the SBSs have no prior knowledge of the wireless communications environment (e.g., channel quality distribution, content popularity distribution, backhaul bandwidth). In such cases, the utility function of the SBSs would be unknown and need to be estimated through learning and adaptation. Hierarchical reinforcement learning techniques need to be proposed to learn the SBS strategies in the Stackelberg equilibrium.

Funding No funding was received to assist with the preparation of this manuscript.

Availability of data and materials Data sharing is not applicable – no new data generated.

Code availability Not applicable.

Declarations

Conflicts of interest All authors declare that they have no conflict of interest that are relevant to the content of this article.

References

1. Golrezaei, N., Molisch, A. F., Dimakis, A. G., & Caire, G. (2013). Femtocaching and device-to-device collaboration: A new architecture for wireless video distribution. *IEEE Communications Magazine, 51*(4), 142–149. https://doi.org/10.1109/MCOM.2013.6495773
2. Shanmugam, K., Golrezaei, N., Dimakis, A. G., Molisch, A. F., & Caire, G. (2013). Femtocaching: Wireless content delivery through distributed caching helpers. *IEEE Transactions on Information Theory, 59*(12), 8402–8413. https://doi.org/10.1109/TIT.2013.2281606
3. Ahlehagh, H., & Dey, S. (2014). Video-aware scheduling and caching in the radio access network. *IEEE/ACM Transactions on Networking, 22*(5), 1444–1462. https://doi.org/10.1109/TNET.2013.2294111
4. Wang, X., Chen, M., Taleb, T., Ksentini, A., & Leung, V. C. (2014). Cache in the air: Exploiting content caching and delivery techniques for 5G systems. *IEEE Communications Magazine, 52*(2), 131–139. https://doi.org/10.1109/MCOM.2014.6736753
5. Al-Turjman, F. (2018). Fog-Based caching in software-defined information-centric networks. *Computers & Electrical Engineering, 69*, 54–67. https://doi.org/10.1016/j.compeleceng.2018.05.018
6. Liu, D., Chen, B., Yang, C., & Molisch, A. F. (2016). Caching at the wireless edge: Design aspects, challenges, and future directions. *IEEE Communications Magazine, 54*(9), 22–28. https://doi.org/10.1109/MCOM.2016.7565183
7. Yao, J., Han, T., & Ansari, N. (2019). On mobile edge caching. *IEEE Communications Surveys & Tutorials, 21*(3), 2525–2553. https://doi.org/10.1109/COMST.2019.2908280
8. Jeong, M. W., Ryu, J. Y., Kim, S. H., Lee, W., & Ban, T. W. (2020). A completely distributed transmission algorithm for mobile device-to-device caching networks. *Computers & Electrical Engineering, 87*, 1–10. https://doi.org/10.1016/j.compeleceng.2020.106803
9. Zhou, J., Sun, Y., & Tellambura, C. (2022). Physical-layer security for cache-enabled C-RANs via rate splitting. *IEEE Communications Letters, 26*(5), 984–988. https://doi.org/10.1109/LCOMM.2022.3154422
10. Zahed, M. I. A., Ahmad, I., Habibi, D., & Phung, Q. V. (2020). Green and secure computation offloading for cache-enabled IoT networks. *IEEE Access, 8*, 63840–63855. https://doi.org/10.1109/ACCESS.2020.2982669
11. Ochia, O. E., Darwesh, A. F., & Fapojuwo, A. O. (2020). Secrecy rate performance of cache-enabled millimeter wave cellular networks. In *IEEE 92nd Vehicular Technology Conference (VTC 2020-Fall)* (pp. 1–6). https://doi.org/10.1109/VTC2020-Fall49728.2020.9348739
12. Wang, N., Zhao, H., Jin, H., & Hai, L. (2020). Weakly secure coded caching scheme for an eavesdropper having prior knowledge. *IEEE Access, 8*, 15565–15575. https://doi.org/10.1109/ACCESS.2020.2967427
13. Xia, H., Zhou, X., & Li, C. (2020). Security aware caching placement optimization strategy in cooperative networks. *Mobile Networks and Applications, 25*(5), 1729–1735. https://doi.org/10.1007/s11036-020-01583-7
14. Fazel, F., Abouei, J., Jaseemuddin, M., Anpalagan, A., & Plataniotis, K. N. (2021). Secure throughput optimization for cache-enabled multi-UAVs networks. *IEEE Internet of Things Journal, 9*(10), 7783–7801. https://doi.org/10.1109/JIOT.2021.3114086
15. Xu, Q., Su, Z., Zhang, K., & Li, P. (2020). Intelligent cache pollution attacks detection for edge computing enabled mobile social networks. *IEEE Transactions on Emerging Topics in Computational Intelligence, 4*(3), 241–252. https://doi.org/10.1109/TETCI.2019.2918573
16. Xu, Q., Su, Z., & Yang, Q. (2020). Blockchain-based trustworthy edge caching scheme for mobile cyber-physical system. *IEEE Internet of Things Journal, 7*(2), 1098–1110. https://doi.org/10.1109/JIOT.2019.2951007
17. Cui, L., Su, X., Ming, Z., Chen, Z., Yang, S., Zhou, Y., & Xiao, W. (2020). CREAT: Blockchain-assisted compression algorithm of federated learning for content caching in edge computing. *IEEE Internet of Things Journal*. https://doi.org/10.1109/JIOT.2020.3014370
18. Sun, W., Li, S., & Zhang, Y. (2021). Edge Caching in Blockchain Empowered 6G. *China Communications, 18*(1), 1–17. https://doi.org/10.23919/JCC.2021.01.001
19. Wang, Y., Tian, Y., Hei, X., Zhu, L., & Ji, W. (2021). A novel IoV block-streaming service awareness and trusted verification scheme in 6G. *IEEE Transactions on Vehicular Technology, 70*(6), 5197–5210. https://doi.org/10.1109/TVT.2021.3063783
20. Li, C., Liang, S., Zhang, J., Wang, Q. E., & Luo, Y. (2022). Blockchain-based data trading in edge-cloud computing environment. *Information Processing & Management, 59*(1), 102786. https://doi.org/10.1016/j.ipm.2021.102786
21. Yang, Y., Liu, Z., Liu, Z., Chan, K. Y., & Guan, X. (2022). Joint optimization of edge computing resource pricing and wireless caching for blockchain-driven networks. *IEEE Transactions on Vehicular Technology, 71*(6), 6661–6670. https://doi.org/10.1109/TVT.2022.3162075
22. Xiong, J., Xie, H., Liu, B., Li, B., & Gui, L. (2021). Cooperative caching services on high-speed train by reverse auction. *IEEE Transactions on Vehicular Technology, 70*(9), 9437–9449. https://doi.org/10.1109/TVT.2021.3099525
23. Alioua, A., Hamiroune, R., Amiri, O., Khelifi, M., Senouci, S. M., Gidlund, M., & Abedin, S. F. (2022). Incentive mechanism for competitive edge caching in 5G-enabled internet of things. *Computer Networks, 213*, 109096. https://doi.org/10.1016/j.comnet.2022.109096
24. Weifeng, L., Mingqi, Z., Jia, X., Siguang, C., Lijun, Y., & Jian, X. (2020). Cooperative caching game based on social trust for D2D

communication networks. *International Journal of Communication Systems, 33*(9), e4380. https://doi.org/10.1002/dac.4380
25. Li, Q., Sun, Y., Tian, T., Yang, R., Meng, L., Zhang, Y., & Yu, F. R. (2020). Research on security of D2D resource sharing based on blockchain in mobile edge network. In *12th international conference on communication software and networks (ICCSN 2020)* (pp. 202–206). https://doi.org/10.1109/ICCSN49894.2020.9139065
26. Rocha, A. S., Pinheiro, B. A., & Borges, V. C. (2021). Secure D2D caching framework inspired on trust management and blockchain for mobile edge caching. *Pervasive and Mobile Computing, 77*, 101481. https://doi.org/10.1016/j.pmcj.2021.101481
27. Gabry, F., Bioglio, V., & Land, I. (2016). On edge caching in the presence of malicious users. In *IEEE international conference on communications workshops (ICC 2016)* (pp. 278–283). https://doi.org/10.1109/ICCW.2016.7503800
28. Jalalpour, E., Ghaznavi, M., Migault, D., Preda, S., Pourzandi, M., & Boutaba, R. (2018). A security orchestration system for CDN edge servers. In *4th IEEE conference on network softwarization and workshops (NetSoft 2018)* (pp. 46–54). https://doi.org/10.1109/NETSOFT.2018.8459910
29. Natalino, C., de Sousa, A., Wosinska, L., & Furdek, M. (2020). Content placement in 5G-enabled edge/core data center networks resilient to link cut attacks. *Networks, 75*(4), 392–404. https://doi.org/10.1002/net.21930
30. Al-Share, R. A., Shatnawi, A. S., & Al-Duwairi, B. (2022). Detecting and mitigating collusive interest flooding attacks in named data networking. *IEEE Access, 10*, 65996–66017. https://doi.org/10.1109/ACCESS.2022.3184304
31. Wen, W., Liu, C., Fu, Y., Quek, T. Q., Zheng, F. C., & Jin, S. (2020). Enhancing physical layer security of random caching in large-scale multi-antenna heterogeneous wireless networks. *IEEE Transactions on Information Forensics and Security, 15*, 2840–2855. https://doi.org/10.1109/TIFS.2020.2976961
32. Xie, T., Nambiar, N., He, T., & McDaniel, P. (2022). Attack resilience of cache replacement policies: A study based on TTL approximation. *IEEE/ACM Transactions on Networking*. https://doi.org/10.1109/TNET.2022.3171720
33. Guzey, S., Karabulut-Kurt, G., Mhaish, A., Ozdemir, E., & Tavakkoli, N. (2022). Secure device-to-device caching with blockchain. *IEEE Internet of Things Journal, 9*(20), 20750–20762. https://doi.org/10.1109/JIOT.2022.3177574
34. Beitollahi, H., & Deconinck, G. (2008). Analyzing the chord peer-to-peer network for power grid applications. In *4th IEEE young researchers symposium in electrical power engineering* (p. 5).
35. Wang, J., Wei, X., Fan, J., Duan, Q., Liu, J., & Wang, Y. (2022). Request pattern change-based cache pollution attack detection and defense in edge computing. *Digital Communications and Networks*. https://doi.org/10.1016/j.dcan.2022.03.019
36. Nash, J. (1951). Non-cooperative games. *Annals of Mathematics*. https://doi.org/10.2307/1969529
37. Monderer, D., & Shapley, L. S. (1996). Potential games. *Games and Economic Behavior, 14*(1), 124–143. https://doi.org/10.1006/game.1996.0044
38. Fudenberg, D., & Tirole, J. (1991). *Game theory*. MIT Press.
39. Von Stackelberg, H. (2011). *Market structure and eequilibrium*. Springer.
40. Yang, Z., Tian, H., Fan, S., & Chen, G. (2017). Distributed cooperative caching in backhaul-limited small cell networks. *Electronics Letters, 53*(3), 158–160. https://doi.org/10.1049/el.2016.3221
41. Xie, M., Widjaja, I., & Wang, H. (2012). Enhancing cache robustness for content-centric networking. In *2012 proceedings IEEE INFOCOM* (pp. 2426–2434). https://doi.org/10.1109/INFCOM.2012.6195632
42. Liao, J., Wong, K. K., Khandaker, M. R., & Zheng, Z. (2016). Optimizing cache placement for heterogeneous small cell networks. *IEEE Communications Letters, 21*(1), 120–123. https://doi.org/10.1109/LCOMM.2016.2612197
43. Xie, R., Li, Z., Huang, T., & Liu, Y. (2017). Energy-efficient joint content caching and small base station activation mechanism design in heterogeneous cellular networks. *China Communications, 14*(10), 70–83. https://doi.org/10.1109/CC.2017.8107633
44. Tran, T. D., Hoang, T. D., & Le, L. B. (2018). Caching for heterogeneous small-cell networks with bandwidth allocation and caching-aware BS association. *IEEE Wireless Communications Letters, 8*(1), 49–52. https://doi.org/10.1109/LWC.2018.2851202

Zahra Rashidi Zahra Rashidi received her B.S. degree in Information Technology Engineering from Amirkabir University of Technology, Tehran, Iran, in 2014 and her M.S. degree in Information Technology Engineering (Computer Networks) from Sharif University of Technology, Tehran, Iran, in 2017. She is currently pursuing the Ph.D. degree in Computer Engineering in Iran University of Science and Technology, Tehran, Iran. In her Ph.D. thesis, she studies the application of game-theoretic learning for caching contents in small-cell base stations.

Vesal Hakami received his B.Sc. degree in Computer Engineering (Software) and his M.Sc. and Ph.D. degrees in Information Technology (Computer Networks), all from Amirkabir University of Technology (AUT), Tehran, Iran, in 2004, 2008 and 2015, respectively. Following graduation, he has served as a research consultant in Iran Telecommunications Research Center (ITRC), working on standardization issues for future wireless networks. In 2016, he joined as an Assistant Professor to the School of Computer Engineering, Iran University of Science and Technology (IUST), Tehran, Iran. His current research mainly focuses on cognitive control of computer networks using stochastic control theory, and game-theoretic learning.