Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 150512, 12 pages
http://dx.doi.org/10.1155/2015/150512

Research Article

Bloom Filter-Based Secure Data Forwarding in Large-Scale Cyber-Physical Systems

Siyu Lin[1,2] and Hao Wu[2]

[1] *School of Electronic and Information Engineering, Beijing Jiaotong University, No. 3 Shang Yuan Cun, Haidian District, Beijing 100044, China*
[2] *State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, No. 3 Shang Yuan Cun, Haidian District, Beijing 100044, China*

Correspondence should be addressed to Siyu Lin; sylin@bjtu.edu.cn

Received 24 September 2014; Accepted 19 March 2015

Academic Editor: Yang Tang

Cyber-physical systems (CPSs) connect with the physical world via communication networks, which significantly increases security risks of CPSs. To secure the sensitive data, secure forwarding is an essential component of CPSs. However, CPSs require high dimensional multiattribute and multilevel security requirements due to the significantly increased system scale and diversity, and hence impose high demand on the secure forwarding information query and storage. To tackle these challenges, we propose a practical secure data forwarding scheme for CPSs. Considering the limited storage capability and computational power of entities, we adopt bloom filter to store the secure forwarding information for each entity, which can achieve well balance between the storage consumption and query delay. Furthermore, a novel link-based bloom filter construction method is designed to reduce false positive rate during bloom filter construction. Finally, the effects of false positive rate on the performance of bloom filter-based secure forwarding with different routing policies are discussed.

1. Introduction

Over the past few years, the cyber-physical systems (CPSs) have attracted considerable attention due to their wide applications in various areas such as power grids [1], intelligent transportation systems [2], and aerospace systems [3]. Communication networks are the connecting links between CPSs and physical world. To defend the attacks from the external and internal of networks destroying the data exchange or stealing data, the CPSs need a secure data forwarding scheme to achieve an end-to-end protection for sensitive data [4].

The forwarders can alter, drop, and delay existing packets, thus sensitive data packets should only be forwarded by the entities that have secure forwarding capabilities. On one hand, CPSs could be large-scale systems including many different types of devices and data, such as smart grid systems; they require different security levels (e.g., high, medium, and low) on different security attributes (e.g., integrity, confidentiality, and availability) [5–7]. The high dimensional multiattribute and multilevel security requirements require efficient secure forwarding capability management and evaluations. On the other hand, the ever increased system scales of CPSs and diverse secure forwarding policies significantly increase the amount of secure forwarding information, which introduces the secure information storage and query problem to the entities which have limited storage capability and computational power (such as embedded devices in CPSs).

Bloom filters can achieve space-efficient storage with constant query delay, which have been applied in many applications [8]. In this paper, we propose a novel secure data forwarding scheme for large-scale CPSs that achieves well balance between the space-efficiency and query delay based on the bloom filters. Compared with traditional secure data forwarding schemes, the proposed scheme improves the space-efficiency while achieving a comparable query delay. However, bloom filters suffer from the *false positive* issues, which are especially undesirable for the secure forwarding information management and evaluation. A link-based bloom filter construction method is further proposed to reduce the possibility of secure forwarding policies violation caused by the false positive issue.

Specifically, our main contributions in this work include the following.

(i) To the best of our knowledge, our work is the first attempt to design a generic and practical secure data forwarding scheme based on the high dimensional multiattribute and multilevel security requirements for large-scale CPSs.

(ii) We propose a bloom filter-based secure forwarding capabilities representation method, which improves the storage efficiency while achieving constant query delay. Furthermore, we derive the optimal secure forwarding information storage size.

(iii) To simplify the construction process of bloom filters and minimize the violation of security policies caused by the potential false positive issue, we design a novel link-based bloom filter construction method, with which the offspring entities construct their own bloom filters based on the incoming links from their respective parent entities.

(iv) Finally, we evaluate the performance of proposed secure data forwarding scheme with two classical routing protocols, for example, Open Shortest Path First (OSPF) and Greedy Perimeter Stateless Routing (GPSR) routing protocol. The impacts of false positive problem on the average length of routing path are investigated.

2. Preliminaries

2.1. System and Security Models. The CPS is consistent with amount of entities. Within the CPS, there are M security attributes such as confidentiality, integrity, availability, and authentication [5, 9]. Each security attribute has L security levels, such as high, medium, and low when $L = 3$. The security capability of an entity is described by its *security label*, which is security attributes and corresponding security levels combination. The security label is denoted as $\langle l_1, \dots, l_m, \dots, l_M \rangle$, where l_m specifies the security level of the mth security attribute that the entity can offer. If $l_m = 1$, this means that the entity can not provide the mth security attribute. Similarly, the secure forwarding requirements of a packet are also denoted as a security label. If the packet does not require the mth security attribute, then $l_m = 1$ for the security label of packet. For example, in a smart grid system, it may include six main security attributes: integrity, confidentiality, availability, privacy, authentication, and nonrepudiation. Then, for smart meter data reports, they may require integrity, confidentiality, and privacy at high levels. In contrast, the control commands may require high integrity, availability, and authentication [5, 10].

The secure forwarding policy of CPS is expressed in terms of security labels attached to the entities and packets [11]:

(i) The entity A satisfies the security requirement of the mth security attribute of a packet p if and only if $l_m(A) \geq l_m(p)$.

(ii) We define that the security label of an entity A *dominates* the security label of a packet p if and only if $l_m(A) \geq l_m(p)$ $(1 \leq m \leq M)$ for all security attributes of packet p, denoted as

$$\langle l_1, \dots, l_M \rangle_A \succeq \langle l_1, \dots, l_M \rangle_p . \tag{1}$$

The entity A has secure capability to forward the packet p if and only if the entity A dominates the security label of the packet p.

Here we use an example to illustrate the secure forwarding policy. Supposing the CPS has two security attributes and there are three security levels on each security attribute, the security labels of entity A and entity B are $\langle 3, 2 \rangle_A$ and $\langle 2, 1 \rangle_B$, respectively, and the security label of a packet p is $\langle 2, 2 \rangle_p$. Since $\langle 3, 2 \rangle_A$ dominates $\langle 2, 2 \rangle_d$ while $\langle 2, 1 \rangle_B$ cannot dominate $\langle 2, 2 \rangle_p$, only entity A can securely forward packet p.

2.2. Secure Forwarding Capability. To deliver a packet to its destination(s), we need to evaluate the forwarding capability of each potential forwarder for the given security label of the packet. However, the forwarding capability of an entity can not only be described by its security label, but also be jointly determined based on the security labels of entities along the forwarding path. Next, we use an example to clearly illustrate it. Considering a CPS with four entities, one packet with security label $\langle 3, 2 \rangle$ needs to be forwarded to destination D, which is shown in Figure 1. In this case, although the security label of entity $A\langle 3, 2 \rangle_A$ dominates the security label of the packet, A still cannot forward the packet to its destination, because neither of the potential forwarders B and C has the capability to deliver the packet. This example shows that an entity has forwarding capability for a given packet if and only if there is a path between the entity and the destination along which all the entities have the capabilities to deliver the packet. Therefore, the secure forwarding capability of an entity (*E2E deliverable set*) is determined by a set of security labels that an entity can receive (*receivable set*) and a set of security labels that its neighbors can forward to the destination (*E2E deliverable sets of neighbors*).

The receivable set of an entity includes all security labels dominated by the entity's security label, which is derived only based on its security label. For example, the receivable set for entity A in Figure 1 is $\{\langle 1, 1 \rangle, \langle 1, 2 \rangle, \langle 2, 1 \rangle, \langle 2, 2 \rangle, \langle 3, 1 \rangle, \langle 3, 2 \rangle\}$. On the other hand, according to the definition of forwarding capability, the E2E deliverable set of an entity is the intersection between its receivable set and the union of E2E deliverable sets of its neighboring entities. For instance, for the destination entity D, the E2E deliverable sets for entities B and C are $\{\langle 1, 1 \rangle, \langle 1, 2 \rangle, \langle 2, 1 \rangle, \langle 2, 2 \rangle\}$ and $\{\langle 1, 1 \rangle, \langle 1, 2 \rangle, \langle 1, 3 \rangle, \langle 2, 1 \rangle, \langle 2, 2 \rangle, \langle 2, 3 \rangle\}$, respectively. As the receivable set for entity A is $\{\langle 1, 1 \rangle, \langle 1, 2 \rangle, \langle 2, 1 \rangle, \langle 2, 2 \rangle, \langle 3, 1 \rangle, \langle 3, 2 \rangle\}$, the E2E deliverable set of A should be $\{\langle 1, 1 \rangle, \langle 1, 2 \rangle, \langle 2, 1 \rangle, \langle 2, 2 \rangle\}$. For a given

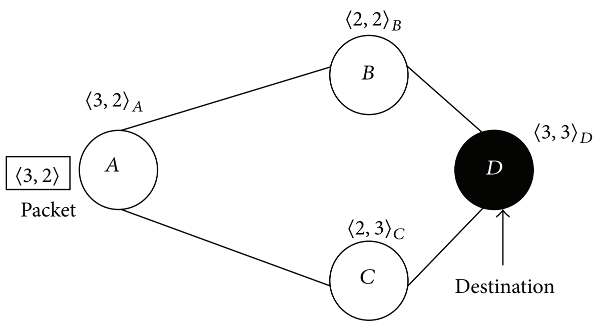


Figure 1: Entities and packets with their security labels.

destination, we adopt the distributed Bellman Ford algorithm to calculate the forwarding capability of each entity [12].

Considering an entity with security label $\langle l_1,\ldots,l_m,\ldots,l_M\rangle$, its E2E deliverable set potentially contains $l_1\times\cdots\times l_m\times\cdots\times l_M$ security labels at most, so the size of E2E deliverable set may grow dramatically with the increasing number of security attributes and security levels. However, many entities in CPSs, such as embedded computers and sensors, have limited computational power and storage space [2, 13]. To achieve effective secure forwarding, how to effectively store and query the forwarding capability information with low computation and storage overhead needs to be addressed. In this work, we propose a bloom filter-based secure forwarding scheme to tackle these challenges.

3. Bloom Filter-Based E2E Deliverable Set

In this section, we first introduce the fundamentals of the bloom filter. Next, we analyze the optimal size of bloom filter-based E2E deliverable set. At last, we propose a link-based deliverable set construction approach to reduce the negative impact on security policies caused by the false positive problems.

3.1. Bloom Filter Fundamentals. A bloom filter (BF) is a bit vector with v bits, which is used to test whether an element is a member of a set. Initially, the bloom filter is an empty vector; all v bits are set to 0. Considering a set $S=\{s_1,s_2,\ldots,s_N\}$ with N elements and K independent hash functions, to represent the set S to a v-bit bloom filter, each element s_n is mapped by the K independent hash functions to K positions $h_1(s_i),h_2(s_i),\ldots,h_K(s_i)$ in the v-bit vector, all set to 1 (each bit might be set to 1 multiple times), where $h_k()$ is the kth hash function, $k\in\{1,2,\ldots,K\}$. After all the elements of set S are mapped to the v-bit vector, the v-bit vector is the bloom filter expression of S, denoted as BF(S).

To query whether an element x is in the set S, map x to K bit positions $h_1(x),h_2(x),\ldots,h_K(x)$ of BF(S) via the K independent hash functions. If any of these bit positions is 0, x is definitely not in the set S. Otherwise, we conjecture that x is in the set [8]. The query delay of bloom filter is a constant, described by $O(K)$, which is only determined by the number of hash functions and independent from the number of elements in the set.

However, bloom filter achieves the above advantages with cost of false positive. False positive problem results from the fact that several positions of BF are shared with multiple elements. If the positions of BF(S) indicated by hash function results of an element y, $h_1(y),h_2(y),\ldots,h_k(y)$ have been set to 1 by several other elements of set S, but actually y is not in the S, then the query result will be a false positive when it queries y in the BF(S). Therefore, the false positive problem should be paid more attention in the various application with bloom filter. The probability of such false positive for a given BF can be expressed as [8]

$$f=\left(1-\left(1-\frac{1}{v}\right)^{NK}\right)^K\approx\left(1-\exp\left(-\frac{NK}{v}\right)\right)^K, \qquad (2)$$

where v is the size of BF, N is the number of elements in the BF, and K is the number of independent hash functions.

3.2. BF-Based E2E Deliverable Set Size Optimization. Since bloom filter is a space-efficiency data structure with constant query delay, which is suitable to represent the E2E deliverable set at individual entities, in this section, we analyze the minimal size of BF-based E2E deliverable set.

Based on (2), the size of the E2E deliverable set can be calculated as

$$v=\frac{-K}{\ln\left(1-\sqrt[K]{f}\right)}N, \qquad (3)$$

where N is the number of deliverable security labels inserted into the E2E deliverable set and f is the false positive rate of the deliverable set. Considering that many entities have limited storage space and computational power, we aim to minimize the sizes of the E2E deliverable sets with the false positive rate bound. Then, we can formulate our objective as an optimization problem as follows:

$$\min \quad v=\frac{-K}{\ln\left(1-\sqrt[K]{f}\right)}N, \qquad (4)$$

$$\text{s.t.} \quad 0\le f\le\varepsilon, \qquad (5)$$

$$N=N_m,\quad K\in\mathbb{N}, \qquad (6)$$

$$\text{given} \quad N_m,\varepsilon. \qquad (7)$$

Constraint (5) is the bound on the false positive rate. Constraint (6) is the maximum number of elements in the E2E deliverable set of an entity.

According to (4), the size of the E2E deliverable set increases linearly with the number of security labels. Since $-K/\ln(1-\sqrt[K]{f})>0$, $N>0$, and they are independent. To minimize the size of deliverable set, we define $-K/\ln(1-\sqrt[K]{f})$ as the size factor, which can be optimized independently. The above optimization problem can be expressed as

$$\min \quad \frac{-K}{\ln\left(1-\sqrt[K]{\varepsilon}\right)}, \qquad (8)$$

$$\text{s.t.} \quad K\in\mathbb{N}. \qquad (9)$$

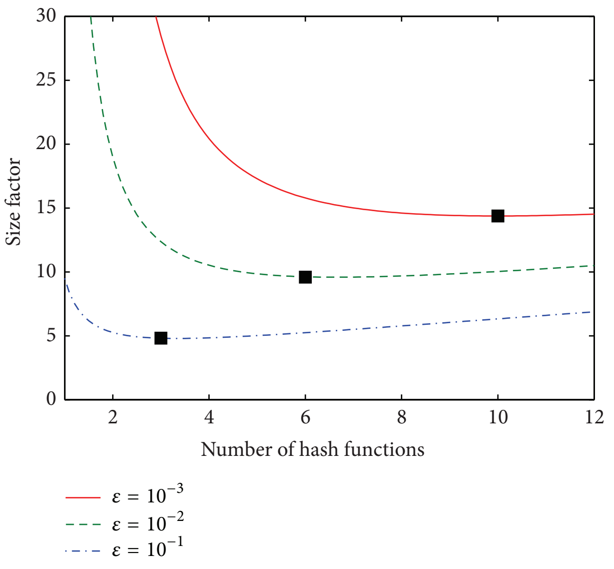


FIGURE 2: The size factor as a function of number of hash functions with different false positive rate bounds.

The optimal number of hash functions for (8) can be obtained with interior point methods [14]. For a given false positive bound ε with the optimal number of hash functions K^*, the size of the deliverable set is $\lceil(-K^*/\ln(1-\sqrt[K^*]{\varepsilon}))N_m\rceil$ bits.

Figure 2 shows the impacts of number of hash functions on the size factor, that is, (8), with different false positive bounds, where the optimal number of hash functions with different false positive bounds is marked as black square. From the figure we can see that, with the decrease of false positive rate bounds, the size factor and optimal number of hash functions are increased. Since the size of bloom filter-based deliverable set and query delay are proportional to the size factor and optimal number of hash functions, respectively, then the decrease of false positive rate bounds also leads to the increased size of bloom filter-based deliverable set and query delay. For example, the false positive rate bound decreases from 10^{-1} to 10^{-3}, the optimal size of deliverable set and corresponding number of hash functions increases from $4.83N_m$ bits and 3 to $14.3776N_m$ and 10, respectively.

3.3. *E2E Deliverable Set Construction.*

Adopting the bloom filter to represent the E2E deliverable set can achieve efficient storage and forwarding capability query delay, but the false positive problem is introduced in forwarding capability query. To reduce the negative impact of security policies caused by the false positive problems; in this section, we design a link-based method to construct the bloom filter-based E2E deliverable set, with which each entity constructs its E2E deliverable set based on the incoming links of its neighboring entities.

3.3.1. *Set Operations of Bloom Filter.*

Before designing the E2E deliverable set construction method, we introduce two common set operations of bloom filter such as union and intersection, because they can simplify the E2E deliverable set construction process in this work.

Considering two sets S_A and S_B, as well as their corresponding bloom filters $\mathrm{BF}(S_A)$ and $\mathrm{BF}(S_B)$, the properties of union and intersection operations can be presented by the following two theorems [15].

The union of $\mathrm{BF}(S_A)$ and $\mathrm{BF}(S_B)$ can be represented by a logical *or* operation between their bloom filters.

Theorem 1. *If $BF(S_A \cup S_B)$, $BF(S_A)$, and $BF(S_B)$ have the same length and hash functions, then $BF(S_A \cup S_B) = BF(S_A) \cup BF(S_B)$.*

The intersection of $\mathrm{BF}(S_A)$ and $\mathrm{BF}(S_B)$ can be represented by a logical *and* operation between their bloom filters.

Theorem 2. *If $BF(S_A \cap S_B)$, $BF(S_A)$, and $BF(S_B)$ have the same length and hash functions, then $BF(S_A \cap S_B) = BF(S_A) \cap BF(S_B)$ with probability $(1-1/m)^{K^2 \times |S_A - S_A \cap S_B| \times |S_B - S_A \cap S_B|}$, where K is the number of hash functions.*

According to the above two theorems, we can observe that using the intersection operations of bloom filter cannot always get the accurate bloom filter of two corresponding sets intersection, but using the union operation can get the accurate bloom filter of two corresponding sets union. Therefore, to minimize potential false positive rate, we should only utilize the union operation of bloom filters for E2E deliverable set construction.

3.3.2. *Bloom Filter Construction.*

The bloom filter is used to store all the security labels of E2E deliverable set. As mentioned in Section 2.2, the E2E deliverable set of an entity is the intersection set between its receivable set and the union of E2E deliverable sets of its neighboring entities. Intuitively, each entity do union operation among all the bloom filter-based E2E deliverable sets of its neighbors firstly, then do the intersect operation between its own bloom filter-based receivable set and the union result to obtain its bloom filter-based deliverable set. However, due to the potential false positives for intersection operations in Theorem 2, this intuitive solution cannot work well in practice. For example, assume there are two sets $S_A = \{s_1, s_2\}$ and $S_B = \{s_3, s_4\}$, with their bloom filter representations as $\mathrm{BF}(S_A) = [1010011]$ and $\mathrm{BF}(S_B) = [1100110]$, respectively. Although $S_A \cap S_B = \emptyset$, $\mathrm{BF}(S_A) \cap \mathrm{BF}(S_B) = [1000010]$ is not an empty bloom filter due to the false positive caused by bloom filter intersection operations.

To reduce the impact of intersection operation on false positive rate, we should *only* perform the union operation to construct the bloom filter-based E2E deliverable set. Therefore, we introduce a link-based bloom filter construction scheme to construct the bloom filter-based E2E deliverable set. The scheme consists of three phases: link security label calculation, link bloom filter construction, and bloom filter-based E2E deliverable set construction. Next, we elaborate each of the three phases in detail with a walkthrough example in Figure 3. In the example, assuming that entities A and B have constructed their deliverable sets, we mainly focus

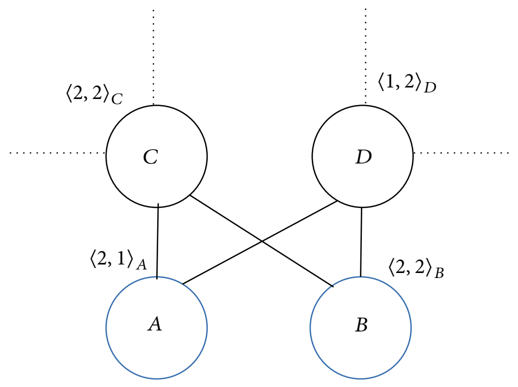


Figure 3: Example network.

on investigating how to construct the bloom filters for intermediate nodes, such as C and D.

Link Security Label Calculation. The link security label is used to describe the maximum forwarding capability between two entities, which is defined as the lower level of each attribute for these two entities; that is, $\langle l_1, \dots, l_m, \dots, l_M\rangle_{AB} = \langle \min\{l_1(A), l_1(B)\}, \dots, \min\{l_M(A), l_M(B)\}\rangle$. The link security label is used to filter the deliverable security labels of this link. For example, as shown in Figure 4, $\langle 2,1\rangle_A$ and $\langle 2,2\rangle_C$ are security labels of entities A and C. Then the security label of link AC is $\langle \min\{2,2\}, \min\{1,2\}\rangle = \langle 2,1\rangle_{AC}$.

Link Bloom Filter Construction. The link bloom filter contains the security labels which can be delivered by either end of entities. To insert a security label into the link bloom filter, an entity which has already constructed its bloom filter extracts one of the security labels within its bloom filter and then compares it with the link security labels of all associated links. If the extracted label is dominated by the link security label, then it will be inserted into to the corresponding link bloom filter.

For example, since A has constructed its bloom filter, the security label of A is $\langle 2,1\rangle_A$ and its bloom filter is [1010110]. Firstly, entity A extracts each of its security labels $\langle 1,1\rangle$ and $\langle 2,1\rangle$ from its constructed bloom filter, which is shown in Figure 5(a). Then, entity A compares each of them with link security labels of AC and AD. Security label $\langle 1,1\rangle$ is dominated by both link security label of AC and AD, $\langle 2,1\rangle_{AC}$, and $\langle 1,1\rangle_{AD}$; then $\langle 1,1\rangle$ is inserted into the bloom filters of link AC and AD. Similarly, $\langle 2,1\rangle$ is only dominated by $\langle 2,1\rangle_{AC}$; then $\langle 2,1\rangle$ is inserted into the bloom filter of link AC. The above processes are shown in Figure 5(b).

Bloom Filter-Based E2E Deliverable Set Construction. After the link bloom filters are constructed for an entity, the entity simply performs the union operation among all its link bloom filters to obtain its bloom filter-based E2E deliverable set. For example, assuming the bloom filters of link AC and BC are [1010110] and [1110111], entity C performs the union operation between the two link bloom filters of link AC and BC to obtain its bloom filter-based E2E deliverable set [1110111], which is shown in Figure 6.

4. Secure Forwarding

After all the E2E deliverable sets are constructed, the forwarding capability of each entity is expressed as a bloom filter with its security label. In this section, we design the bloom filter-based secure data forwarding scheme. To deliver a packet to its destination(s), we need to evaluate the forwarding capability of each potential forwarder for the given security label of the packet; the forwarding capability evaluation is implemented via bloom filter query. The objective of forwarding capability query is that the source entity or a forwarder judges which neighbors have the capability to forward the packet to destination(s). This is a forwarder candidates selection process, whose outputs are the feasible forwarders satisfied the secure forwarding policy. The final forwarder is decided by routing policy, such as shortest path routing protocols and geographic routing protocols. The bloom filter-based forwarding capability query can be added to any security aware routing protocols as a middleware.

The forwarding capability query process has two steps. Firstly, the forwarder compares the security label of the packet with the security labels of its neighbors; secondly, if the security label of the packet is dominated by the neighbor's security label, then, query whether the security label of packet is in the bloom filter-based deliverable set of this neighbor. If the security label can be extracted from the E2E deliverable set, this entity is selected as a forwarder candidate. In particular, the false positive problem will not lead to the fact that packets are forwarded to the entities which have no capability to receive the packets; this is because the fact that the forwarder compares the security label of the packet with the security labels firstly.

After all the forwarder candidates are selected from the neighbors by relay entities or source entity; the forwarder selects the best candidate as the final forwarder according to the routing policy. In this work, we select two classic routing protocols, such as shortest path routing protocol and geographic routing protocol, which are combined with bloom filter-based forwarding capability query to implement the secure forwarding.

Shortest path routing is the most widely used routing protocol; Open Shortest Path First (OSPF) routing protocol is selected as it is the most typical shortest path routing. OSPF collects link state information from available entity and constructs a topology map of the network. The topology is presented as a routing table, and it computes the shortest path tree for each route using a method based on Dijkstra's algorithm [16]. Bloom filter-based E2E deliverable set introduces the false positives into the forwarding capability query, which may lead to the wrong routing decision; that is, an entity is selected as the forwarder that cannot forward the data to the destination(s). When a false positive occurs, the packet will be forwarded continuously until a forwarder who cannot find the next forwarder to the destination(s) without routing loop. In this case, the false positive can be detected. When a node detects a false positive, the false positive recovery mechanism will be implemented [17]. When a false positive is detected, the forwarder marks the packet as the recovery packet and unicasts the data back to the previous forwarder.

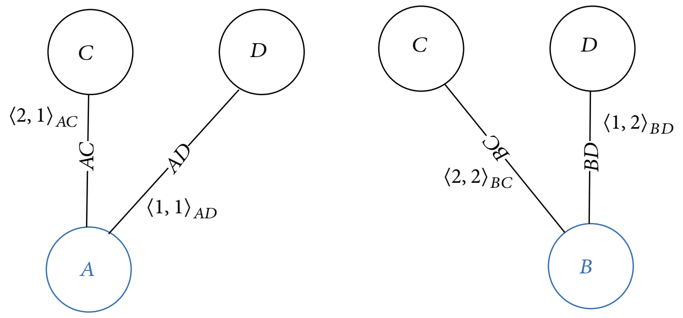
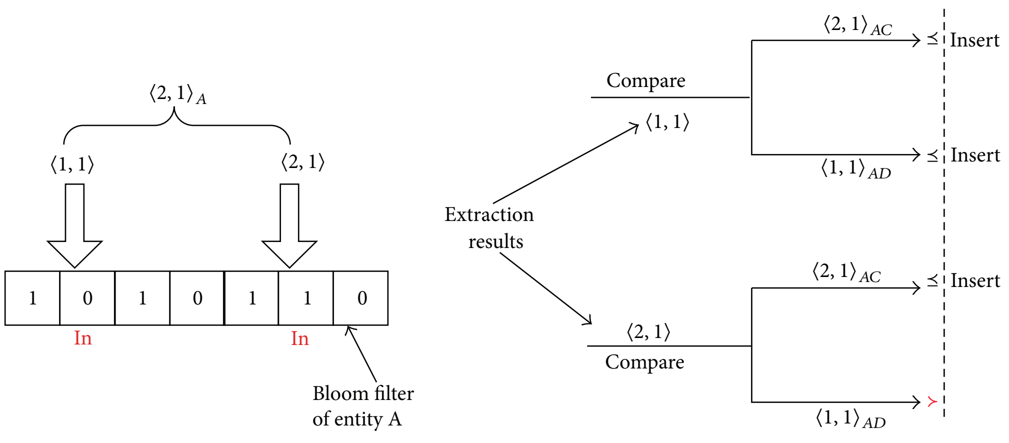


FIGURE 4: Calculate the link security label.



(a) Extraction

(b) Comparison

FIGURE 5: Link bloom filter construction.

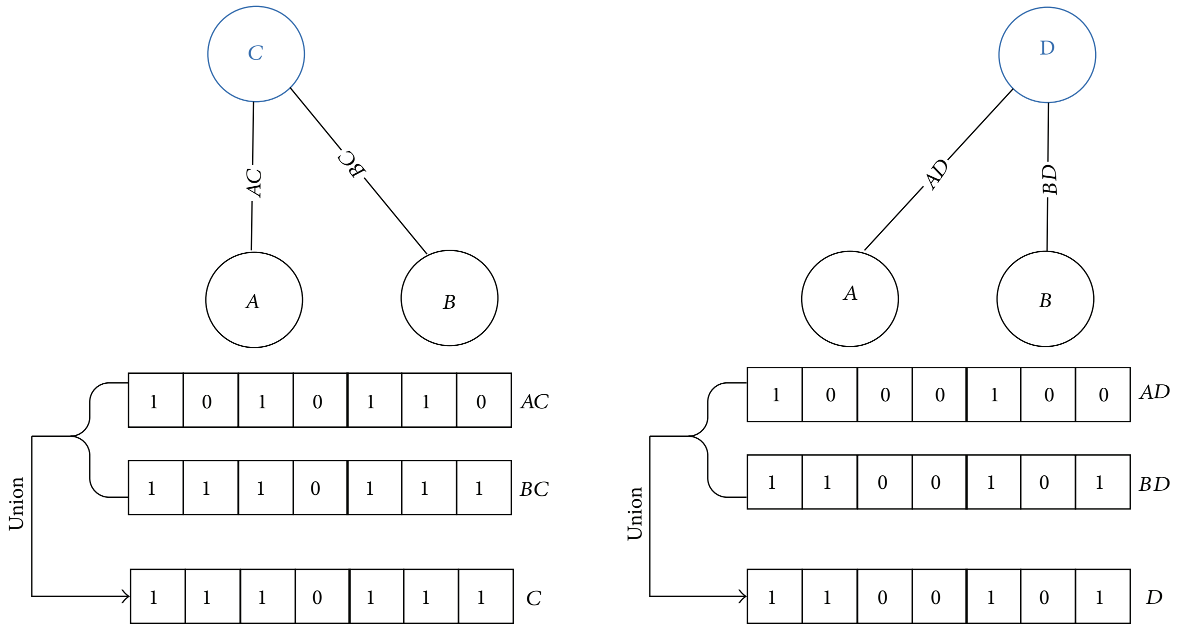


FIGURE 6: Bloom filter-based E2E deliverable set construction.

Once an entity receives the recovery data, it will select another forwarder to forward this data. If it fails again, the recovery mechanism will be implemented recursively. The worst case is a full exploration of the entities which have the ability to forward this packet. But, in practice, for the given reasonable desired false positive rate, the recovery mechanism is rarely implemented.

We select Greedy Perimeter Stateless Routing (GPSR) as another routing policy which is a typical geographic routing [18]. The packets are forwarded to neighboring entity which is closer to the destination based on a local point of view. The above greedy forwarding mechanism may lead into a dead end, where there is no neighbor closer to the destination, which means there is a void. If a void is detected, GPSR will activate the Planar Perimeters mode and adopt right hand rule to bypass void. The Planar Perimeters mode of GPSR can provide good insight about the effect of false positive on the forwarding routing path of routing protocols.

5. Performance Evaluation

In this section, we evaluate the performance of bloom filter-based secure forwarding scheme through extensive simulations.

5.1. Simulation Setting. We simulate a large-scale CPS with 1000–1500 entities randomly deployed nodes. The average number of neighbors of an entity varies from 20 to 30, and a sink deployed in the center of the field. The default number of attributes of CPS is set to 10. The default number of security levels is set as $L = 4$; each security attribute is generally classified into four levels, such as high, medium, low, and not applicable [5]. The default false positive rate bound ε is set as 10^{-2}. The presented results are average over 30 runs. We select the bitmap [17] scheme as the baseline, in which the deliverable set is represented to a bit vector. Each bit corresponds to a unique deliverable security label. Therefore, the bitmap can provide accurate forwarding capability query results without false rate.

5.2. Performance Evaluation. In this section, we compare the E2E deliverable set size and query delay between bitmap and bloom filter scheme. As mentioned in Section 2.1, considering the diversity of various entities, each kind of entity only supports a subset of attributes [10, 19], and packets normally also only require a subset of attributes of the system according to their functionalities, so we define the maximum number of attributes that a packet requires as the number of effective security attributes, whose default number is set as 6 in this work.

5.2.1. E2E Deliverable Set Size. In this section, we investigate the optimal sizes of the two different schemes under different false positive rate bounds, different number of security levels, and different number of effective security attributes. Since packets normally also only require a subset of attributes of the system, the total number of effective security labels of the packets that the system supports can be calculated as $\sum_{i=0}^{M_p} \binom{M}{i} (L-1)^i$, where M_p is the number of effective security attributes, which is also the size of bitmap-based E2E deliverable set. Similarly, the maximum number of effective security labels of each entity, N_m, is $\sum_{i=0}^{M_p} \binom{M_e}{i} (L-1)^i$, where M_e is the maximum number of security attributes that entities can support. The security level on each attribute is randomly assigned. The optimal size of BF-based E2E deliverable set, $\lceil (-K^* / \ln(1 - \sqrt[K^*]{\varepsilon})) N_m \rceil$ bits, is derived based on the optimization function (8) in Section 3.2.

Impact of False Positive Rate Bound. The bloom filter achieves the space-efficiency with the cost of false positive rate, so we first study the effect of false positive rate bound on the size of E2E deliverable set. Figure 7 shows the sizes of two schemes when false positive rate bound varies from 10^{-4} to 10^{-1}. The BF-based scheme requires smaller size when the false positive rate bound becomes larger, which is intuitive. We can see that the size of BF-based deliverable set outperforms the bitmap-based scheme with any given false positive rate bounds. For example, even in the case when false positive rate is 10^{-4}, the size of BF-based deliverable set only achieves 32% of the size of bitmap-based deliverable set. This is because the fact that the BF-based deliverable set only needs to store effective security labels of each entity, but all the effective security labels of the packets that the system supports need to be stored in bitmap-based deliverable set.

Impact of the Number of Security Levels. Since the maximum number of effective security labels in the E2E deliverable set of each entity is determined by the number of security levels, we study the effect of the number of security levels on the size of E2E deliverable set. Figure 8 shows the sizes of two schemes under different number of security levels. When the number of security levels varies from 2 to 6, the advantage on space-efficiency of BF becomes more obvious. For example, when $L = 2$, comparing with the bitmap, BF-based deliverable set size is a little large compared to that of bitmap scheme; however, when the number of security levels increases to 6, the size improvement of BF achieves about 90%. This is because that the gap between the maximum number of effective security labels of system and that of entities becomes larger with the increase of security levels.

Impact of the Effective Security Attributes. The number of effective security attributes also impacts the maximum number of effective security labels in the E2E deliverable set; then it further affects the storage consumption of entities. Here, the number of effective security attributes varies from 1 to 10; the maximum number of attributes that an entity can support equals the number of effective security attributes. Figure 9 shows the sizes of two schemes under different percentage of effective security attributes. With the percentage of effective security attributes increasing, the sizes of BF-based deliverable set increase as well. When the number of effective security attributes of entities is less than 80% of system attributes, comparing with bitmap, the BF scheme has smaller size. While the number of effective security attributes of entities exceeds 80% of system attributes, the maximum

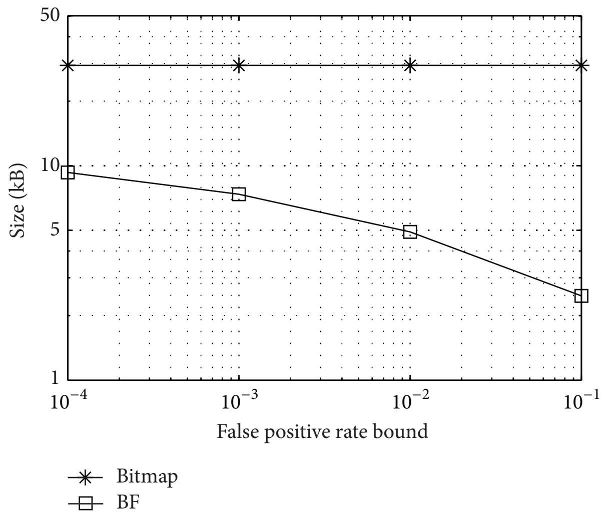


Figure 7: E2E deliverable set size comparison with varying false positive rate bounds.

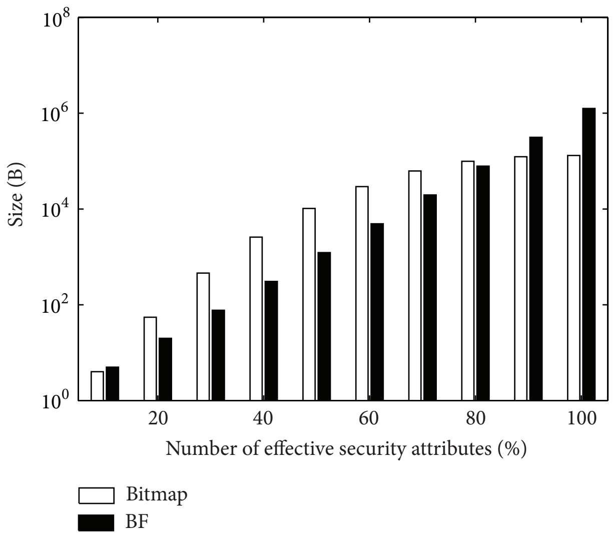


Figure 9: Impact of the number of effective security attributes.

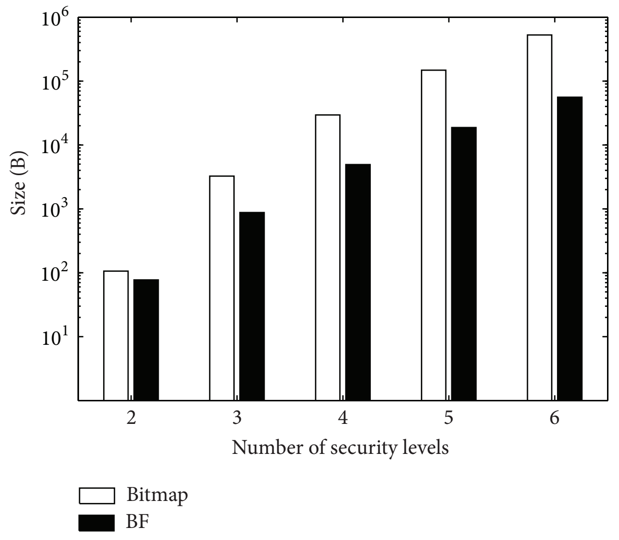


Figure 8: Impact of the number of security levels.

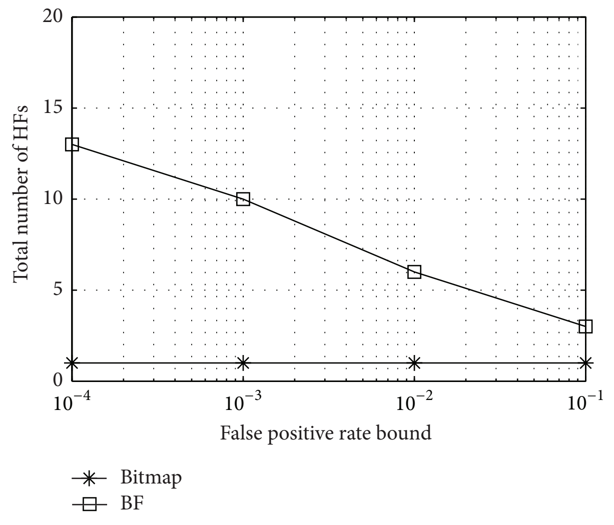


Figure 10: Total number of hash operations with varying false positive rate bound.

number of security labels of E2E deliverable set is comparable to the number of security labels that the system supports, so the sizes of BF scheme exceed the bitmap. When the percentage of effective security attributes achieves 40%, the sizes of BF and bitmap are 307 Bytes and 2586 Bytes; the size improvement of BF achieves more than 88%.

5.2.2. Query Delay. In this section, we examine the query delay of two different schemes under different false positive rate bounds. The query delay of bitmap is $O(1)$, as it is a one-to-one mapping. For a given false positive rate bound, the query delay of BF scheme can be evaluated by the optimal number of hash operations $O(K^*)$, which are derived based on the optimization function (8) in Section 3.2.

Figure 10 shows the optimal number of hash operations under different false positive rate bounds of BF scheme. As false positive rate bound decreases, the number of hash operations of BF increases, which means that the query delay is increased. This is because the optimal number of hash functions is increased with smaller false positive rate. For example, when the false positive rate bound decreases from 10^{-1} to 10^{-4}, the optimal number of hash operations of BF increases from 3 to 13.

5.2.3. Average Length of Routing Path. Utilize the bloom filter to represent the E2E deliverable set of entities; the false positive problem may lead the forwarder to make wrong routing decision; then the length of routing path may be stretched. In this section, we investigate the impact of false positive rate bound of E2E deliverable set on the average length of routing path of OSPF routing with recovery mechanism and GPSR routing. We select the OSPF with

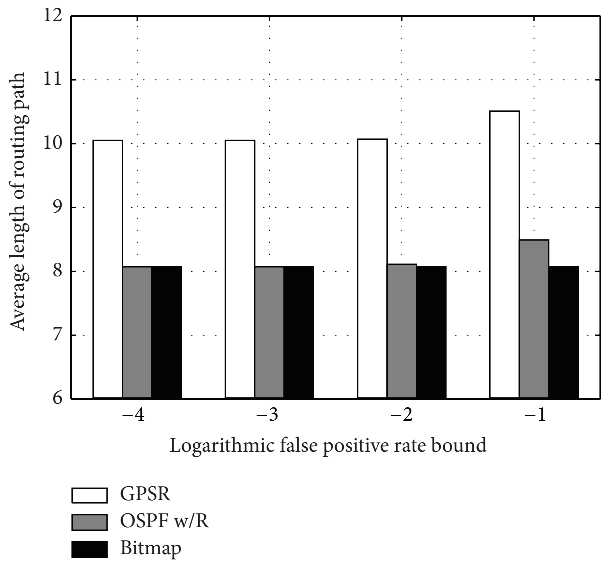


Figure 11: Impact of false positive rate bounds on average forwarding hops.

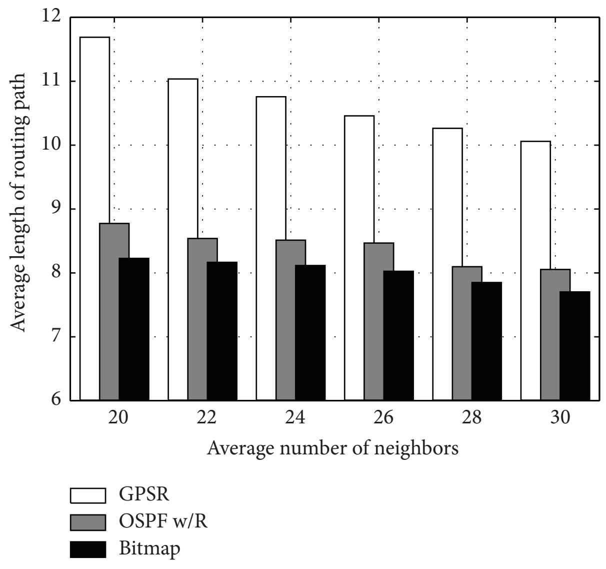


Figure 12: Impact of average of number of neighbors on average forwarding hops.

bitmap-based E2E deliverable set as the baseline; this is because that this scheme can achieve the shortest routing path. All the results are obtained by averaging 1000 source-to-sink communications.

Impact of the False Positive Rate Bounds. Figure 11 shows the average length of routing path of three different secure forwarding schemes under different false positive rate bounds. When the false positive rate bound is less than 10^{-3}, the average length of routing path of OSPF with recovery mechanism and GPSR routing almost keep the same; this is because the incorrect forwarding capability query results rarely occur when the false positive rate bound is less than 10^{-3}. With the increase of false positive rate bound, the average length of routing path also increases, which resulted from the increased possibility of the routing protocol triggering a recovery mechanism [17] of OSPF and Planar Perimeters mode [18] of GPSR to reach the destination via longer path. For example, comparing the cases of false positive rate bound of 10^{-4} and 10^{-1}, the average length of routing path of OSPF and GPSR is stretched about 5% and 7%. The above results demonstrate that the routing path of bloom filter-based secure forwarding scheme is not stretched obviously when the appropriate false positive rate bound is selected. Furthermore, the average length of GPSR routing is still larger than that of OSPF routing, even when the recovery mechanism of OSPF is usually activated; this is because the GPSR routing wastes more routing path on detouring the void.

Impact of the Average Number of Neighbors. The average length of routing path of different schemes is also impacted by the average number of neighbors of an entity. Figure 12 shows the average length of routing path of different schemes with different number of neighbors, where the false positive rate bound is 10^{-1}. When the average number of neighbors increases from 20 to 30, the average length of routing path of the three schemes is also decreased. The decreasing rate of GPSR is faster than OSPF with bloom filter-based E2E deliverable set; this is because the fact that probability of void is decreased quickly with increase of number of neighbors.

6. Related Work

Two important research topics are involved in this work: secure aware routing and bloom filter application in computer systems. We briefly survey the related works of these two topics in this section.

6.1. Secure Aware Routing. Existing research efforts on security aware routing can be classified into two categories: security reactive routing and security proactive routing [20].

The security reactive routing protocols obtain the necessary security path when required, by using a connection establishment process, which are typical for mobile networks. Many of the existing security reactive routing protocols have been extended from Ad Hoc On-Demand Distance Vector (AODV) routing and Dynamic Source Routing protocol (DSR) routing, such as SAR [9], TQOS [21], Castor [22], Usor [23], THMR [24], and RMR [25]. Security Aware Ad Hoc Routing (SAR) protocol [9] is the most typical one, which adopts trust levels (a security attribute assigned to nodes) to make secure routing decisions. SAR can discover a path consisted of the node that can provide the required security level. TQOS [21] and RMR [25] select the trustworthiness-based quality of service and space reliability as the security metrics to build the security path for defending the attack launched by malicious nodes. THMR [24] focuses on solving the topology-exposure problem to defend against the attacks with a low overhead and short routing convergent time.

The security proactive routing protocols maintain the network topology in the form of routing tables by periodically exchanging security routing information. SEAD [26] and SLSP [27] are two typical security proactive routing protocols. SEAD uses a one-way hash function for authenticating to distinguish update messages that are received from nonmalicious nodes and malicious nodes. SLSP is a secure link state protocol, which adopts one-way hash function and a public key cryptosystem to ensure the security of link state updates.

Our work is the first attempt to efficiently represent amount of secure state information to achieve storage efficiency, quick query, and easy to maintain requirements, which is compatible with existing secure aware routing protocols.

6.2. Bloom Filter Application in Computer Network. Bloom filter is a space-efficient data structure that supports set membership queries with constant delay; therefore, bloom filters and its variants have been widely used in computer network and distribution system [28, 29], such as packet forwarding, information-guided routing, and supporting security operations.

Bloom filters are very useful tools for fast packet forwarding. A bloom filter-based longest-prefix matching algorithm is introduced in [30]. The basic idea is to determine the longest matching prefix membership in sets of prefixes sorted by prefix length via parallel queries of bloom filters and then look up the routing results in a large hash table directly. Yu et al. propose the BUFFALO architecture, which uses a small SRAM to store one bloom filter of the addresses to perform the flat address lookup in the data-center network [31]. Duquennoy et al. adopt the bloom filter to store the knowledge of the subtree of the nodes for routing protocol for low-power and lossy networks with tree topology [17]. Different from these works, we focus on using one bloom filter to store secure forwarding information in CPS.

Bloom filters have also been used for information-guided routing [32–34], which adopt bloom filter-based probabilistic algorithms to locate the objects. Kumar et al. introduce exponentially decaying bloom filter in [35], which is a representation of a set of objects of nodes. The basic idea is that the nodes propagate its object information in bloom filter with exponentially decay within the given range. Meanwhile, each node can obtain the incomplete object information of the other nodes via the union of all received bloom filters. Then, the query results of bloom filter of nodes can be used to guide the random walk to locate the objects. Using the similar design, Acer et al. present the weak state routing for large and dynamic networks [33]. Our "link-based bloom filter construction" scheme is similar to their basic idea of utilizing the union operation of bloom filter to obtain the object information of neighboring nodes. Different from decay model, we utilize the link security label to filter the elements to be inserted into the link bloom filter, which can avoid the entity bloom filter becoming a vector of ones.

Bloom filters have also been used for supporting security operations, such as flow admission [36] and attack detection [36–38]. To achieve immediate broadcast authentication and avoid the security vulnerability, Ren et al. propose an efficient public-key-based schemes integrating bloom filters with partial message recovery signatures and Merkle hash trees [36]. To detect the bogus sensing reports in wireless sensor networks, Ye et al. propose a bloom filter-based statistical en-route filtering approach to detect and drop such false reports [37]. In [38], Shieh et al. adopt bloom filter to detect the replay attacks in the Trickles stateless network stack and transport protocol. Different from these works, we adopt bloom filter to represent the forwarding capability in a space-efficient way, which can assist entities to find an appropriate secure data forwarding path with a constant delay. The bloom filter-based deliverable set representation scheme can be added to the other secure routing protocols of CPS as the access control middleware.

7. Conclusions

In this paper, we design a practical secure data forwarding scheme for large-scale cyber-physical systems based on the multiattribute and multilevel security requirements, in which the secure forwarding capability information of each entity is represented as a bloom filter. We derived the optimal size of bloom filter and corresponding query delay. Furthermore, we design a novel link-based bloom filter construction method to control the false positive rate of the constructed bloom filter. Finally, the impact of false positive rate bound of forwarding capability query on the average length of secure routing path is investigated. The space-efficiency and comparable query delay of proposed scheme with bloom filter are verified through simulations.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was partly supported by the Fundamental Research Funds for the Central Universities (no. 2015RC032), the National Natural Science Foundation of China (no. U1334202, no. U1261109, and no. 61201201), the Key Grant Project of Chinese Ministry of Education (no. 313006), the State Key Laboratory of Rail Traffic Control and Safety (no. RCS2015 K011 and no. RCS2014ZT31), and the Program for New Century Excellent Talents in University (no. NCET-13-0657).

References

[1] S. Sridhar, A. Hahn, and M. Govindarasu, "Cyber-physical system security for the electric power grid," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 210–224, 2012.

[2] E. A. Lee, "Cyber physical systems: design challenges," in *Proceedings of the 11th IEEE International Symposium on Object Oriented Real-Time Distributed Computing (ISORC '08)*, pp. 363–369, Orlando, Fla, USA, May 2008.

[3] O. S. Saydjari, "Cyber defense: art to science," *Communications of the ACM*, vol. 47, no. 3, pp. 52–57, 2004.

[4] A. Cardenas, S. Amin, B. Sinopoli, A. Giani, A. Perrig, and S. Sastry, "Challenges for securing cyber physical systems," in

Proceedings of the Workshop on Future Directions in Cyber-Physical Systems Security, 2009.

[5] Y. Mo, T. H.-J. Kim, K. Brancik et al., "Cyber-physical security of a smart grid infrastructure," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 195–209, 2012.

[6] C. Alvaro, A. Saurabh, S. Bruno, G. Annarita, P. Adrian, and S. Shankar, "Challenges for securing cyber physical systems," in *Proceedings of the Workshop on Future Directions in Cyber-physical Systems Security (DHS '09)*, Newark, NJ, USA, July 2009.

[7] B. Panja, S. K. Madria, and B. Bhargava, "A role-based access in a hierarchical sensor network architecture to provide multilevel security," *Computer Communications*, vol. 31, no. 4, pp. 793–806, 2008.

[8] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[9] S. Yi, P. Naldurg, and R. Kravets, "Security-aware ad hoc routing for wireless networks," in *Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc '01)*, pp. 299–302, October 2001.

[10] H. Khurana, M. Hadley, N. Lu, and D. A. Frincke, "Smart-grid security issues," *IEEE Security and Privacy*, vol. 8, no. 1, pp. 81–85, 2010.

[11] R. S. Sandhu, "Lattice-based access control models," *IEEE Computer*, vol. 26, no. 11, pp. 9–19, 1993.

[12] B. Awerbuch, A. Bar-Noy, and M. Gopal, "Approximate distributed bellman-ford algorithms," *IEEE Transactions on Communications*, vol. 42, no. 8, pp. 2515–2517, 1994.

[13] D. Culler, D. Estrin, and M. Srivastava, "Guest Editors'introduction: overview of sensor networks," *Computer*, vol. 37, no. 8, pp. 41–49, 2004.

[14] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

[15] D. Guo, J. Wu, H. Chen, Y. Yuan, and X. Luo, "The dynamic bloom filters," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 1, pp. 120–133, 2010.

[16] D. Sidhu, T. Fu, S. Abdallah, R. Nair, and R. Coltun, "Open shortest path first (ospf) routing protocol simulation," *ACM SIGCOMM Computer Communication Review*, vol. 23, no. 4, pp. 53–62, 1993.

[17] S. Duquennoy, O. Landsiedel, and T. Voigt, "Let the tree bloom: scalable opportunistic routing with ORPL," in *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems (SenSys '13)*, Roma, Italy, November 2013.

[18] B. Karp and H.-T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks," in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MOBICOM '00)*, pp. 243–254, ACM, August 2000.

[19] S. Ravi, A. Raghunathan, P. Kocher, and S. Hattangady, "Security in embedded systems," *ACM Transactions on Embedded Computing Systems*, vol. 3, no. 3, pp. 461–491, 2004.

[20] L. Abusalah, A. Khokhar, and M. Guizani, "A survey of secure mobile Ad hoc routing protocols," *IEEE Communications Surveys and Tutorials*, vol. 10, no. 4, pp. 78–93, 2008.

[21] M. Yu and K. K. Leung, "A Trustworthiness-based qoS routing protocol for wireless ad hoc networks," *IEEE Transactions on Wireless Communications*, vol. 8, no. 4, pp. 1888–1898, 2009.

[22] W. Galuba, P. Papadimitratos, M. Poturalski, K. Aberer, Z. Despotovic, and W. Kellerer, "Castor: scalable secure routing for ad hoc networks," in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM '10)*, pp. 1–9, San Diego, Calif, USA, March 2010.

[23] Z. Wan, K. Ren, and M. Gu, "USOR: an unobservable secure on-demand routing protocol for mobile ad hoc networks," *IEEE Transactions on Wireless Communications*, vol. 11, no. 5, pp. 1922–1932, 2012.

[24] Y. Zhang, G. Wang, Q. Hu, Z. Li, and J. Tian, "Design and performance study of a topology-hiding multipath routing protocol for mobile ad hoc networks," in *Proceedings of the Conference on Computer Communications (INFOCOM '12)*, pp. 10–18, March 2012.

[25] A. A. Gohari, R. Pakbaz, P. M. Melliar-Smith, L. E. Moser, and V. Rodoplu, "RMR: reliability map routing for tactical mobile ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 10, pp. 1935–1947, 2011.

[26] Y.-C. Hu, D. B. Johnson, and A. Perrig, "SEAD: secure efficient distance vector routing for mobile wireless ad hoc networks," *Ad Hoc Networks*, vol. 1, no. 1, pp. 175–192, 2003.

[27] P. Papadimitratos and Z. J. Haas, "Secure link state routing for mobile ad hoc networks," in *Proceedings of the Symposium on Applications and the Internet Workshops*, pp. 379–383, January 2003.

[28] A. Broder and M. Mitzenmacher, "Network applications of bloom filters: a survey," *Internet Mathematics*, vol. 1, no. 4, pp. 485–509, 2004.

[29] S. Tarkoma, C. E. Rothenberg, and E. Lagerspetz, "Theory and practice of bloom filters for distributed systems," *IEEE Communications Surveys & Tutorials*, vol. 14, no. 1, pp. 131–155, 2012.

[30] S. Dharmapurikar, P. Krishnamurthy, and D. E. Taylor, "Longest prefix matching using bloom filters," in *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM '03)*, pp. 201–212, ACM, 2003.

[31] M. Yu, A. Fabrikant, and J. Rexford, "BUFFALO: bloom filter forwarding architecture for large organizations," in *PRoceedings of the 5th ACM International Conference on Emerging Networking Experiments and Technologies (CoNEXT '09)*, pp. 313–324, Rome, Italy, December 2009.

[32] H. Jiang and S. Jin, "Exploiting dynamic querying like flooding techniques in unstructured peer-to-peer networks," in *Proceedings of the 13TH IEEE International Conference on Network Protocols (ICNP '05)*, pp. 122–131, November 2005.

[33] U. G. Acer, S. Kalyanaraman, and A. A. Abouzeid, "Weak state routing for large-scale dynamic networks," *IEEE/ACM Transactions on Networking*, vol. 18, no. 5, pp. 1450–1463, 2010.

[34] D. Guo, Y. He, and Y. Liu, "On the feasibility of gradient-based data-centric routing using bloom filters," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 180–190, 2014.

[35] A. Kumar, J. Xu, and E. W. Zegura, "Efficient and scalable query routing for unstructured peer-to-peer networks," in *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '05)*, vol. 2, pp. 1162–1173, March 2005.

[36] K. Ren, S. Yu, W. Lou, and Y. Zhang, "Multi-user broadcast authentication in wireless sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 8, pp. 4554–4564, 2009.

[37] F. Ye, H. Luo, S. Lu, and L. Zhang, "Statistical en-route filtering of injected false data in sensor networks," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM '04)*, pp. 2446–2457, March 2004.

[38] A. Shieh, A. C. Myers, and E. G. Sirer, “A stateless approach to connection-oriented protocols,” *ACM Transactions on Computer Systems*, vol. 26, no. 3, article 8, 2008.