Computer Science and Information Systems 13(1):93–114 DOI: 10.2298/CSIS150320045G

Publish/Subscribe Scheme For Content Centric MANETs

Xian Guo[1], Tao Feng[1], Cheng Chen[1], and Tong Liu[2]

[1] School of Computer and Communication
Lanzhou University of Technology, Lanzhou 730050, China
iamxguo@aol.com, fengt@lut.cn, 410614506@qq.com
[2] Beijing Computing Center, Beijing 100094, China
liutong@bcc.ac.cn

Abstract. It is allowed that data of the content producer is cached anywhere in Content Centric MANET (CCMANET). This scheme decoupling of data from the data source make traditional end-end authentication transmission unavailable in CCMANET. So, it is a challenge how to ensure content publishing or subscribing only by legitimate users. In this paper, we firstly design a new Yaksha system on Elliptic Curve (EC-Yaksha), and then a secure Content Publish/Subscribe System based on EC-Yaksha (ECY-CPS) for CCMANET is proposed. In ECY-CPS, EC-Yaksha server manages joining or leaving of network users and distributes "license" only to legitimate user for content publishing or subscribing. Such, it is guaranteed that only the legitimate users can publish content to network or access content cached in network. In addition, using of the license can efficiently defend common attacks such as the interest flooding and the content pollution etc. Finally, we prove security properties of ECY-CPS in PCL and make a simple comparison between our system and the related solution.

Keywords: information centric networking, content centric networking/named data networking, content delivery, mobile ad hoc networks.

1. Introduction

According to the recent Cisco Visual Networking Index Global Forecast and Service Adoption for 2013 to 2018 [1]: content delivery networks will carry over half of internet traffic by 2018; Traffic from wireless and mobile devices will exceed traffic from wired devices by 2018. To satisfy these new application requirements such as content delivery and mobile application etc., the research community pays close attention to the discussion of new internet architecture [2, 3]. In this context, Information Centric Networking (ICN) has emerged as a promising candidate for the architecture of the future internet. CCN/NDN (Content Centric Networking [4]/ Named Data Network [5]) is pioneering fully-fledged ICN architecture. The CCN/NDN has rapidly gained consensus, thanks to the simple, robust and effective communication model.

CCN/NDN has two built-in features that are beneficial to mobility: its data-centric design and stateful forwarding plane. First, every packet, either Interest or Content, carriers a content name only, instead of any address. Second, the states of content requests, namely interest packets in the Pending Interest Tables (PITs) of the forwarding plane, enable the reverse path forwarding of content packets, which makes the location of content consumers transparent to the routing plane and content producers. Thus,

CCN/NDN natively supports data delivery to mobile consumers. So, these features make CCN/NDN a particularly attractive solution for wireless ad hoc environments, like Mobile Ad hoc Networks (MANETs) [6-11]. However, in CCN/NDN, decoupling of data from the source allows data to be stored anywhere. This in-network caching makes traditional end-end authentication transmission unavailable in CCN/NDN. It is a challenge how to ensure publishing of network data only by legitimate users and high availability of the cached data only to legitimate users. In [12], Christopher A. Wood proposed a content distribution scheme based on Proxy Re-Encryption [13]. We call this scheme as PRE-SCD. The content delivery doesn't rely on any server in PRE-SCD, but it requires the content producer is always online. Misra et al. [14] developed a secure content delivery scheme based on Broadcast Encryption [15, 16] to preserve content accessibility in the event that producers are taken offline. We call this scheme as BE-SCD. However, BE-SCD needs to rely on the internet infrastructure or CDN. BE-SCD is not fit for a multi-source environment Such as CCMANET in which any legitimate user can publish the produced content.

In addition, CCN/NDN changes the security model from securing the path to securing the content, which is available to all ICN nodes. As a consequence, new attacks have appeared with this new security model in addition to the legacy attacks that may have an impact on ICN. Surveys of security for ICN are investigated in [17-18]. Attacks to ICN are classified into four categories: naming, routing, caching, and other miscellaneous related attacks in [17]. New privacy challenges are studied in [18]. They also think that the interest flooding and the content pollution are two important attacks to ICN architecture. So, it is necessary to design a secure content publish/subscribe scheme for CCN/NDN especially in sensitive network environments, such as the tactical and emergency MANETs.

Yaksha system [19] is a variant of public-key cryptosystems RSA [20], in which the RSA private key is split in two portions. One portion becomes a user's Yaksha private key, and the other the Yaksha server's private key. This scheme can be used to key escrow, joint digital Signature and key exchange et al. cryptographic functions.

In this paper, we firstly design a new Yaksha system on ECC (Elliptic Curve Cryptography [21]) and call it EC-Yaksha system. The principal attraction of ECC, compared to RSA, is that it can offer equal security for a far smaller key size, thereby reducing processing overhead. And then, we propose a secure content publish/subscribe system based on EC-Yaksha. It is abbreviated to ECY-CPS (Content Publish/Subscribe system based on Yaksha on Elliptic Curve) in this paper. In ECY- CPS, Joining and leaving of network users is uniformly managed by EC-Yaksha server. It is particularly important that EC-Yaksha server is responsible for monitoring content publishing and subscribing. That is to say, the EC-Yaksha server only issues the "license" (a joint signature between the content consumer and the EC-Yaksha server) for publishing or subscribing content to legitimate user. So, it can be guaranteed by ECY-CPS that only the legitimate user can publish content to network and access content cached in network. Our scheme is fit for multi-source environment, and doesn't require that the content producer is always online because using of the EC-Yaksha server that is independent of the producer, and our new solution may don't rely on the internet infrastructure or CDN. In addition, our scheme also can efficiently prevent the interest flooding attack and the content pollution attack because of using the joint signature.

Formal security analysis based on logic is an important approach to verify security properties of a network and cryptographic protocol. Protocol Composition Logic (PCL)

[22, 23] is a formal logic for stating and proving security properties of network protocols. The logic was originally proposed in 2001 and has evolved over time through case studies of widely-used protocol such as SSL/TLS, IEEE 802.11i, and variants of Kerberos. In [24], based on observational equivalence theory, we extend PCL to be Anonymity PCL (APCL) to satisfy the special needs of anonymous analysis. In this paper, we model and demonstrate secrecy and authentication properties of ECY-CPS in PCL. Finally, we make a comparison analysis between our system and the related solutions.

The rest of the paper is organized as follows. Section 2 presents the related work. Some backgrounds on ECY-CPS is given in section 3, followed by PCL in section 4. in section 5 an elliptic curve based new Yaksha system is proposed. The simplification model of CCMANET is given in section 6. ECY-CPS is developed in section 7, and security properties are proved in section 8. Comparison analysis is made in section 9 with conclusion in section 10.

2. Related Work

2.1. PRE-SCD Protocol

Proxy Re-Encryption (PRE) [13] enables decryption rights to be selectively delegated. Based on proxy re-encryption with identity-based cryptography, a secure content dissemination scheme is proposed in [12]. We refer to the scheme as PRE-SCD (Proxy Re-Encryption based Secure Content Distribution). In PRE-SCD, the content producer encrypts the published content using the identity of the respective content name. Because all content would be protected with the PRE scheme so as to (1) enable effective use of network caches, (2) prevent key and content leakages (i.e. users exposing decryption keys or the decrypted content), (3) simplify key management, and (4) improve overall security by true end-to-end encryption. This ecosystem is captured in Figure1. With an architecture based entirely on PRE for content protection, each piece of content would be encrypted once by the producer using the identity of the respective content name. Since the producer is the only entity that can generate and store the corresponding secret key for this identity, the content remains secure as it is distributed throughout the network. Note that a content consumer who is interested in some content needs to apply for a corresponding re-encryption key from the producer when it receives the encrypted content. Until it re-encrypts the encrypted content from the producer or the network using the secret key of the respective itself identity, the content consumer can decrypt to obtain the interested content. It is obvious that this scheme allows any user publish content to the network.

However, in PRE-SCD, to guarantee that only a legitimate user can acquire the re-encryption key and further access the content on the network, an end-end authentication process between the content producer and the content consumer must be required even if a content consumer obtains the encrypted content from a neighboring router that cached this content. So, to satisfy this requirement, it must be necessary that the content producer is forever online. However, this can be difficult in a claimed environment that does not rely on trusted CDNs or traditional PKI cryptosystems. In our scheme, the

encrypted/decrypted key for content is managed and distributed by the Yaksha server that is required online forever.

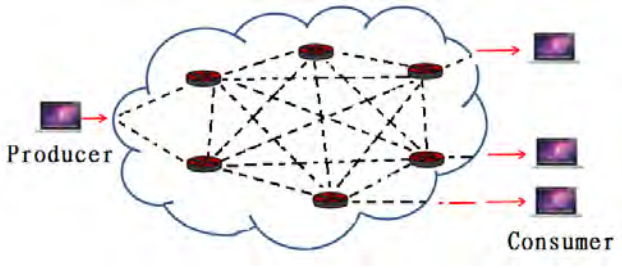

Fig. 1. PRE-SCD Architecture [12]

2.2. BE-SCD Protocol

Broadcast encryption [15, 16] is well known technique for secure content distribution. In the typical scenario, a content producer will encrypt the published content for a subset of users and then publishes the ciphertext on a specific channel. Each content consumer belonging to the original subset and who is also listening on that channel may then use their secret share to decrypt the content. In [14], a secure content delivery for ICN based on broadcast encryption was proposed. We name the scheme as BE-SCD. In BE-SCD, the content producer (such as Netflix etc.) firstly encrypts the published content using a secret key and generates an Enabling Block (EB) that contains the information for a legitimate user to extract the secret key using its' secret share (have been in advance distributed to the legitimate user in registers stage). And then the producer publishes the encrypted content along with EB to the network. In [14], the authors assume a hierarchical set-up as shown in Fig. 2. The encrypted content flows from the content producer, to the Content Delivery Network (CDN) nodes, to the Internet Service Provider (ISP), and finally to the end user who requests the content. It is clear that PRE-SCD relies on the internet infrastructure or CDN.

When a content consumer, who is interested in the received content and possesses the secret share, receives a content packet from a router in its' neighborhood router, it firstly computes the decrypted key for the content from EB using its' secret share and the lagrangian interpolation method. The consumer can get the interested content when it captures the decrypted key. It is the goal of BE-SCD to ensure the content is encrypted and cannot be used by an entity that is not a legitimate user (not even the CDN or the ISPs).

Using of this scheme, which the encrypted content is send along with the information used for calculating the decrypted key by the content consumer, efficiently tackled the problem that the content publisher is always required online in PRE-SCD. However, this scheme requires that the producer (it can be a server or CDN) pre-distributes the secret share to each user in registration stage. So, it isn't fit for our multi-source environment that allows any producer has right to apply for publishing content to the network. And we concern such network that can't rely on CDN and even can't rely on the internet infrastructure.

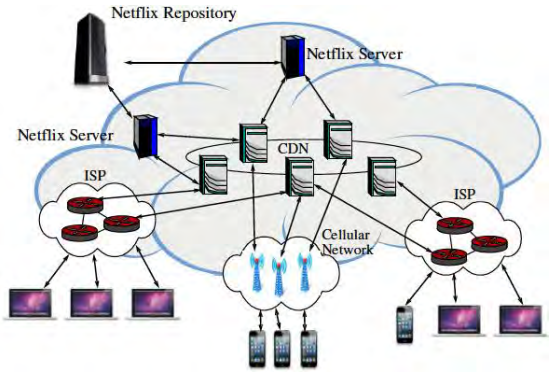

Fig. 2. BE-SCD Architecture [14]

3. Preliminaries

3.1. Overview of CCN/NDN

Information retrieval in CCN/NDN is driven by the consumer, which uses interest packets to request content "by name". Intermediate nodes in select their outgoing network interface(s) for interest forwarding. Finally, upon receiving an interest, a provider, which is the content source or any other network node that temporarily stores the requested content, replies with a named data packet that piggybacks authentication and data-integrity information. The content packet is a self-identify and self-authenticating unit, so in-network data caching is enabled. The data packet follows the "bread crumbs" left by the interest at intermediate nodes back to the consumer. CCN/NDN node is composed of three main components, namely the Forwarding Information Base (FIB), The Pending Interest Table (PIT), and the Content Store (CS). The FIB table stores domains (name prefixes) and their corresponding outgoing face. The PIT table keeps track of outstanding interests; when an intermediate router forwards an interest packet through one of its outgoing faces, a corresponding "pending interest" entry is created in the PIT. The last component, CS, is essentially a content repository. For a more complete discussion of CCN/NDN, please see [4, 5].

3.2. Overview of Yaksha System

The Yaksha system [19] is a variant of the RSA public-key cryptosystem. The system works as follows: as in the RSA system, user Alice has her public key pair (e_A, n_A). Unlike the traditional RSA system, however, the Yaksha system uses two distinct private keys-Alice's private Key, denoted by d_{AA}, and the Yaksha server's corresponding key for Alice, denoted by d_{AY}. These two new private keys are related to the original RSA private key d_A by the mathematical relation $d_{AA} \times d_{AY} = d_A \bmod n_A$.

In the Yaksha system, each user X has his or her own private key d_{XX}, and the Yaksha server maintains a corresponding server private key d_{XY}. The system can be used to perform several security functions such as joint digital signatures, key exchange, key escrow etc.

The joint digital signature is an important technology provided by Yaksha system. Now, we show how the user can interact with the server to sign a message M. Alice calculates $s_1 = M^{d_{AA}} \bmod \varphi(n_A)$ and sends s_1 to the Yaksha server. The Yaksha server uses d_{AY} to calculate the joint signature $s = s_1^{d_{AY}} \bmod \varphi(n_A) = M^{d_{AA} \times d_{AY}} \bmod n_A$. Now, s is Alice's signature on message M and is indistinguishable from a regular RSA signature. The other user can verify the signature using Alice's RSA public key e_A.

4. Protocol Compsition Logic (PCL)

4.1. Syntax of the PCL

Protocol Composition Logic (PCL) is developed in [22, 23]. A simple protocol programming language is used to represent a protocol by a set of roles, such as "initiator" or "server", each specifying a sequence of actions to be executed by an honest participant. Protocol actions include nonce generation, encryption, decryption and communication steps (sending and receiving). Every principal can be executing one or more copies of each role at the same time. PCL use the word thread to refer to a principal executing one particular instance of a role. Each thread X is a pair $(\hat{X}, \eta)$ where $\hat{X}$ is a principal and η is a unique session *id*. A run is a record of all actions executed by honest principals and the attacker during protocol execution. The set of runs of a protocol is determined by the operational semantics of the protocol programming language.

Table 1. Syntax of PCL

Action formulas
$a ::= \text{Send}(X,t) \mid \text{Receive}(X,t) \mid \text{New}(X,t) \mid \text{Sign}(X,t,k) \mid \text{Verify}(X,t,k) \mid$ $\text{SymEnc}(X,t,k) \mid \text{PkEnc}(X,t,k) \mid \text{SymDec}(X,t,k) \mid \text{PkDec}(X,t,k)$
Formulas
$\psi ::= a \mid \text{Has}(X,t) \mid \text{Honest}(\hat{X}) \mid \text{SafeMsg}(M,s,\mathcal{K}) \mid$ $\text{SendsSafeMsg}(X,s,\mathcal{K}) \mid \varphi \wedge \psi \mid \text{SafeNet}(s,\mathcal{K}) \mid \exists V.\varphi$

Table 1 summarizes the syntax of the logic used in this paper. For every protocol action, there is a corresponding action predicate which asserts that the action has occurred in the run. Action predicates are useful for capturing authentication properties of protocols since they can be used to assert which principals sent and received certain messages. For example, predicate Send(X, t) means that the thread X has sent the term t, while New(X, n) means X generates fresh nonce n. Honest($\hat{X}$) means that $\hat{X}$ is acting

honestly, *i.e.*, the actions of every thread of $\hat{X}$ precisely follows some role of the protocol. Interpretations of other principals see [22, 23].

4.2. Proof System of the PCL

Protocol proofs usually use modal formulas of the form $\psi[P]_X\varphi$ in PCL. The modal formulas informally means that if X starts from a state in which ψ holds, and executes the program P, then in the resulting state the security property φ is guaranteed to hold irrespective of the actions of an attacker and other honest principals.

The proof system extends first-order logic with axioms and proof rules for protocol actions, temporal reasoning, knowledge, and a specialized form of invariance rule called the honesty rule. This rule is essential for combining facts about one role with inferred actions of other roles. Intuitively, if Alice receives a response from a message sent to Bob, the honesty rule captures Alice's ability to use properties of Bob's role to reason about how Bob generated his reply. In short, if Alice assumes that Bob is honest, she may use Bob's role to reason from this assumption. PCL's axioms and rules used in this paper see [22, 23].

4.3. Compositional Proof of PCL

Following the modular design of the protocol, the compositional approach is developed in [22, 23]. This method can prove properties of the whole protocol by combining proofs of its parts. In [22, 23], the authors proposed three kinds of composition operation on protocols-parallel, sequential, and staged. One central concept in composition proof methods is the notion of an "invariant". An invariant for a protocol is a logic formula that characterizes the environment in which it retains its security properties. In this paper, we use staged composition in secrecy analysis. The details of staged composition see [23].

5. Yaksha System on Elliptic Curve

Based on Yaksha system [19], EC-Yaksha (Yaksha system on Elliptic Curve) is developed in this paper. We assume that E is elliptic curve group on finite field F_q, $P \in E(F_q)$, and the order of P is a large prime number n. The process creating Yaksha key is descripted as follows.

In EC-Yaksha system, user Alice firstly selects d_{AA}($d_{AA} \in Z_n$) as herself Yaksha private key and computes $Q_{AA}=d_{AA}\times P$, and then she sends Q_{AA} to Yaksha server on a secure channel. When Yaksha server receives Q_{AA} from Alice, it firstly selects d_{AY} as its Yaksha private key corresponding with d_{AA} and computes $Q_A=d_{AY}\times Q_{AA}$, and then sends Q_A to Alice on a secure channel. Now, Alice's public and private key pair is ((d_{AA}, d_{AY}), Q_A), where the private key d_{AA} only is known by Alice and the private key d_{AY} only is

possessed by the Yaksha server. Q_A is Alice's public key corresponding with the private key pair (d_{AA}, d_{AY}).

5.1. Joint Signature on EC-Yaksha

According to ECDSA developed in [21], joint signature of Alice and EC-Yaksha server on message m is developed as follows:

Alice:
(1) Randomly selects $k \in Z_n$.
(2) Computes $k \times P=(x_1,y_1)$ and $r= x_1 \bmod n$;
(3) Computes $k^{-1} \bmod n$;
(4) Computes the signature $s'=d_{AA} \bmod n$, and sends four tuples (m, r, k^{-1}, s') to the Yaksha server on a secure channel.

Yaksha server:
(5) Computes $s=k^{-1} \times (h(m)+d_{AY} \times s') \bmod n$;

Now, (m, r, s) is the joint signature of Alice and the Yaksha server on message m.

5.2. Signature Verification on EC-Yaksha

We assume that Bob receives joint signature (m, r, s), he verifies the signature as follows:
(1) Computes $w=s^{-1} \bmod n$ and hash value $h(m)$;
(2) Computes $u_1=h(m) \times w \bmod n$, $u_2=r \times w \bmod n$;
(3) Computes $u_1 \times P+u_2 \times Q_A =(x_0, y_0)$, $v=x_0 \bmod n$;
(4) If only $v=r$, Bob accepts this signature.

5.3. Correctness Verification

Finally, we prove correctness of this joint signature algorithm.

$$
\begin{aligned}
& u_1 \times P+ u_2 \times Q_A= h(m) \times w \times P+r \times w \times Q_A \\
&= (h(m) \times s^{-1} \times P+ r \times s^{-1} \times d_{AA} \times d_{AY} \times P) \\
&= (h(m) + r \times d_{AA} \times d_{AY}) \times s^{-1} \times P \\
&= (h(m) + r \times d_{AA} \times d_{AY})(k^{-1} \times (h(m)+d_{AY} \times s'))^{-1} \times P \\
&= (h(m) + r \times d_{AA} \times d_{AY}) \times (h(m) + r \times d_{AA} \times d_{AY})^{-1} \times (k^{-1})^{-1} \times P \\
&= k \times P
\end{aligned}
$$

6. System Model

6.1. Network Model

To illustrate our new scheme, simplified CCMANET network model is shown in figure 3. Here, node *S* is a content producer, namely the data source. It may publish content to the network. Node *C* is a user who is interested in some content cached in network, so, we refer to it as a content consumer. The content consumer may subscribe content cached in the network by broadcasting the interest packet. Node *Y* that can be connected with any node is the EC-Yaksha server. The EC-Yaksha server manages all network user information and the relative information of the content in the network. Each user who wants to join the network needs to register to the EC-Yaksha server. Of course, it is deleted from the EC-Yaksha server when a user leaves the network. At the same time, the EC-Yaksha server monitors publishing and subscribing of content. The EC-Yaksha server maintains a database, we refer to the database as Content Information Database (CIDB), to implement the management task. CIDB consists of three main fields: NAME, KEY, SID. Of course, you can append other fields in CIDB to satisfy a concrete application. Here, NAME is the name of some content cached on the network. KEY is the encrypted key generated by the EC-Yaksha server and used to encrypt the published content by *S*, or decrypt the encrypted content by *C*. And SID is the identity of the content producer. The EC-Yaksha server may update CIDB according to the popularity degree of content. This problem will be discussed in the future.

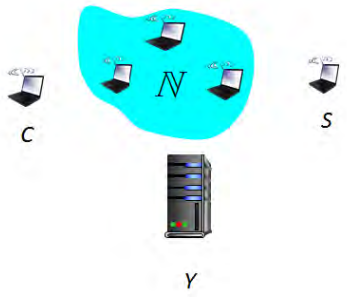

Fig. 3. Simplified CCMANET Network Model

In Figure 3, $\mathbb{N}$ is the network consisted of other nodes except that we consider specific nodes such as *C*, *S*, and *Y*. All of nodes in $\mathbb{N}$ are equipped three components such as CS, PIT and FIB. To cope with wireless links and node mobility, we can select one of forwarding strategies for CCN/NDN packets are proposed in [8-11]. That is to say, $\mathbb{N}$ models the network following requirements of forwarding the interest packet and the content packet in CCN/NDN, and also can cache data.

6.2. Threat model

In our threat model, we consider the consumer *C*, the data source *S*and the EC-Yaksha server are all honest. That is to say, all of them will behave honestly to follow the proposed scheme. We also assume that the network $\mathbb{N}$ will forward the interest packet and the content packet according to our proposed protocol rules. In this paper, we mainly consider the external adversaries. The external adversaries are assumed in our system to eavesdrop the message transmitted on the network. What is more, they could also launch some active attacks, for instance, tampering the message or replaying the old message.

6.3. Security requirements

To prevent attacks from the aforementioned adversaries and ensure the security of content delivery, we require that the proposed scheme should satisfy the security requirements given as follows.

Data confidentiality: It requires that only the legitimate network user can access the content cached on the network, and any other one should be prevented from obtaining the relevant encrypted/decrypted key on the content.

Data authentication: It indicates the authentication for the user identity and message integrity. The former means that only the legitimate network user can broadcast the interest packet to subscribe the interested content from the network or publish content to the network. While the latter guarantees any altered data, during transmission could be detected. These data is often the content name in the interest packet and the content in the content packet.

7. ECY-CPS System

Our new system consists of five stages: EC-Yaksha key generation, content publish, content subscribe, packet forwarding and content fetch. We detailedly illustrate how they work as follows.

7.1. Key Generation

We assume that any user *X* acquires the master secret key K_{XY} shared with the EC-Yaksha server when it registers to the network. And then the EC-Yaksha server and the user *X* collaboratively compute the EC-Yaksha system keys $((d_{XX}, d_{XY}), Q_X)$ as illustrated in section 5. In addition, the EC-Yaksha server also plays the role of CA (Certification Authority) to manage user's public key certification. Finally, we assume that all of users in network use same symmetric cryptography algorithm such as AES or DES.

7.2. Content Publish

We assume that the data source S produces content C_a, and the CCN/NDN name of this content C_a is a, the process of publishing this content is as follows:

1) S firstly applies for a license used to publish content C_a and an encrypted key k_a used to encrypt C_a by sending the following message to the EC-Yaksha server.

$$\{ID_S \| ID_Y \| n_{SY} \| a \| SIG_{d_{SS}}\{H(C_a)\}\} \quad (1)$$

Where n_{SY} in (1) is a nonce, and the subscript SY indicates that the nonce is one that S sends to the EC-Yaksha server. $H(C_a)$ is hash value of the content C_a. $SIG_{d_{SS}}\{H(C_a)\}$ is the signature that S signs on $H(C_a)$ using the user portion d_{SS} of EC-Yaksha private keys.

2) The EC-Yaksha server firstly verifies the identity of S when it receives the message in (1). If this verification is fail, it deletes this message. Otherwise, the EC-Yaksha server generates an encrypted key k_a and records information, such as the identity of S, the name a of content and the encrypted key k_a in CIDB.

3) The EC-Yaksha server signs on $SIG_{d_{SS}}\{H(C_a)\}$ using the server portion d_{SY} of the EC-Yaksha private key further to generate the joint signature $SIG_{d_{SS}\times d_{SY}}\{H(C_a)\}$ of $H(C_a)$. And then it sends the following message to S.

$$\{ID_Y \| ID_S \| SYMENC_{K_{SY}}\{n_{SY} \| SIG_{d_{SS}\times d_{SY}}\{H(C_a)\} \| k_a\} \quad (2)$$

4) When S receives the message in (2), it firstly recovers the joint signature and the encrypted key k_a by decrypting the ciphertext in the message (2) using the shared master key K_{SY}, and then it verifies the joint signature $SIG_{d_{SS}\times d_{SY}}\{H(C_a)\}$ using public key Q_S. If the signature verification successes, it publishes the following content to $\mathbb{N}$ and deletes this key k_a:

$$\{ID_S \| a \| SIG_{d_{SS}\times d_{SY}}\{H(C_a)\} \| SYMENC_{k_a}\{C_a \| H(C_a) \| SIG_{d_{SS}\times d_{SY}}\{H(C_a)\}\}$$

Note that the joint signature $SIG_{d_{SS}\times d_{SY}}\{H(C_a)\}$ is the "license" that S publishes content to the network $\mathbb{N}$.

7.3. Content Subscribe

We assume that a content consumer C is interested in the content C_a (the CCN/NDN name of this content C_a is a), the process that the content consumer C generates and broadcasts the interest packet is as follows:

1) C sends the following message to the EC-Yaksha server, here, n_{CY} is a nonce that C sends to server, and $H(a)$ is hash value of the content name a.

$$\{ID_C \| ID_Y \| a \| n_{CY} \| SIG_{d_{CC}}\{H(a)\}\} \quad (3)$$

2) The EC-Yaksha server firstly verifies the identity of C when it receives the message in (3). If this verification is fail, it deletes this message. Otherwise, the server checks if there exists the content that the name is a on the network by locating its'

CIDB. If the locating is fail, it discards the interest packet, Otherwise, it generates and sends the following messages to C:

$$\{ID_Y \| ID_C \| a \| SYMENC_{K_{CY}}\{n_{CY} \| SIG_{d_{CC} \times d_{CY}}\{H(a)\} \| k_a\}\} \quad (4)$$

3) When C receives the message in (4), it firstly recovers the joint signature and the decrypted key k_a by decrypting the ciphertext in the message (4) using the shared master key K_{CY}, and then it verifies the joint signature $SIG_{d_{CC} \times d_{CY}}\{H(a)\}$ using public key Q_C. If the verification successes, it broadcasts the following interest packet to the network $\mathbb{N}$ and saves this key k_a.

$$\{ID_C \| a \| SIG_{d_{CC} \times d_{CY}}\{H(a)\}\} \quad (5)$$

Note that the joint signature $SIG_{d_{CC} \times d_{CY}}\{H(a)\}$ is the "license" that C subscribes content cached in $\mathbb{N}$.

7.4. Packet Forwarding

Nodes in the network $\mathbb{N}$ will process the interest packet and the content packet according to the CCN/NDN components such as FIB table, PIT table and content store CS, the detailed of this process is shown as follows:

1) When a node n in the network $\mathbb{N}$ receives an interest packet about the content C_a, it firstly verifies the subscribing license using the public key of the subscriber. If this verification is fail, it deletes the interest packet. Otherwise, it checks its' CS and does as follows:
   - If there exists the content C_a in node n's CS, then n deletes the interest packet and directly sends the following content packet to the node who forwarded this interest packet to the node n.

     $$\{ID_S \| a \| SYMENC_{k_a}\{C_a \| H(C_a) \| SIG_{d_{SS} \times d_{SY}}\{H(C_a)\}\}\}$$

   - Otherwise, the node n continues to forward the received interest packet according to its FIB table.
2) When a node n in the network $\mathbb{N}$ receives a content packet, similar to receiving the interest packet, it firstly verifies the publishing license. If this verification is fail, it deletes the content packet, otherwise, it saves this content packet and forwards the content packet according to the information recorded in PIT table.

7.5. Content Fetch

When the content consumer C receives the following content packet,

$$\{ID_S \| a \| SYMENC_{k_a}\{C_a \| H(C_a) \| SIG_{d_{SS} \times d_{SY}}\{H(C_a)\}\}\}$$

1) It firstly decrypts this ciphertext using the decrypted key k_a (obtained from the EC-Yaksha server in the content subscribe stage) to recover the content C_a and the publishing license $SIG_{d_{SS} \times d_{SY}}\{H(C_a)\}$.

2) It verifies the publishing license using the data source S's public key Q_S and calculates the hash value $H(C_a)$. If these verifications success, C accepts the content and deletes the decrypted key k_a. Otherwise, it deletes the received content packet.

Now, we summarize the process of message exchange for ECY-CPS system in table 2 (shown in next page).

Table 2. The Message Exchange of ECY-CPS

$S \to Y : \{ID_S \parallel ID_Y \parallel a \parallel n_{SY} \parallel SIG_{d_{SS}} \{H(C_a)\}\}$
$Y \to S : \{ID_Y \parallel ID_S \parallel a \parallel SYMENC_{k_{SY}} \{n_{SY} \parallel SIG_{d_{SS} \times d_{SY}} \{H(C_a)\} \parallel k_a\}$
$S \to \mathbb{N} : \{ID_S \parallel a \parallel SIG_{d_{SS} \times d_{SY}} \{H(C_a)\} \parallel SYMENC_{k_a} \{C_a \parallel H(C_a) \parallel SIG_{d_{SS} \times d_{SY}} \{H(C_a)\}\}\}$
$C \to Y : \{ID_C \parallel ID_Y \parallel a \parallel n_{CY} \parallel SIG_{d_{CC}} \{H(a)\}\}$
$Y \to C : \{ID_Y \parallel ID_C \parallel a \parallel SYMENC_{k_{CY}} \{n_{CY} \parallel SIG_{d_{CC} \times d_{CY}} \{H(a)\} \parallel k_a\}$
$C \to \mathbb{N} : \{ID_C \parallel a \parallel SIG_{d_{CC} \times d_{CY}} \{H(a)\}\}$
$\mathbb{N} \to C : \{ID_S \parallel a \parallel SIG_{d_{SS} \times d_{SY}} \{H(C_a)\} \parallel SYMENC_{k_a} \{C_a \parallel H(C_a) \parallel SIG_{d_{SS} \times d_{SY}} \{H(C_a)\}\}\}$

8. Security Analysis

8.1. Formal Description of ECY-CPS System

The ECY-CPS system has four roles. A set of roles is {Consumer(), EC-Yaksha(), Source(), Network()}.We use $\hat{C}, \hat{Y}, \hat{S}, \hat{N}$ as names for protocol participants the Consumer, the EC-Yaksha server, the data Source and the Network. C, Y, S, N respectively represents thread of protocol participant. Using protocol programming language of PCL, Roles of ECY-CPS are shown below.

source($S, \hat{Y}, \hat{C}, H(C_a)$){
/* *apply for the key* k_a *used to encrpt the content* C_a */
new n_{SY};
$H(C_a) :=$ hash C_a;
$s_S :=$ sign $H(C_a), d_{SS}$;
send $\hat{S}.\hat{Y}.a.n_{SY}.s_S$;
/* *content publish* */
receive $\hat{Y}.\hat{S}.a.enc_{Y1}$;
$text_{S1} :=$ symdec enc_{Y1}, K_{SY};
match $text_{S1}$ as $n_{SY}.s_{SY}.k_a$;
Verify s_{SY}, Q_S;
$enc_{S1} :=$ symenc $C_a.H(C_a).s_{SY}, k_a$;
send $\hat{S}.a.s_{SY}.enc_{S1}$; }source

consumer($C, \hat{Y}, \hat{S}, \hat{N}, a$){

/* *apply for the key* k_a *used to decrpt the content* */

new n_{CY};

$H(a)$:= hash a;

s_C := sign $H(a), d_{CC}$;

send $\hat{C}.\hat{Y}.a.n_{CY}.s_C$;

/* *content subscribe* */

receive $\hat{Y}.\hat{C}.enc_{Y2}$;

$text_{C1}$:= symdec enc_{Y2}, K_{CY};

match $text_{C1}$ as $n_{CY}.s_{CY}.k_a$;

Verify s_{CY}, Q_C;

send $\hat{C}.a.s_{CY}$;

/* *content fetch* */

receive $\hat{S}.a.s_{SY}.enc_{S1}$;

$text_{C2}$:= symdec enc_{S1}, k_a;

match $text_{C2}$ as $C_a.H(C_a).s_{SY}$;

verify s_{SY}, Q_S;

h_{C1} := Hash(C_a);

match h_{C1} as $H(C_a)$;}consumer

Yaksha($Y, \hat{C}, \hat{S}, H(a), H(C_a)$){

/* *sends the encrpted key to the data source* */

receive $\hat{S}.\hat{Y}.n_{SY}.a.s_S$;

s_{SY} := sign s_S, d_{SY};

enc_{Y1} := symenc $n_{SY}.s_{SY}.k_a, K_{SY}$;

send $\hat{Y}.\hat{S}.a.enc_{Y1}$;

/* *sends the decrpted key to the content consumer* */

receive $\hat{C}.\hat{Y}.n_{CY}.a.s_C$;

s_{CY} := sign s_C, d_{CY};

enc_{Y2} := symenc $n_{CY}.s_{CY}.k_a, K_{CY}$;

send $\hat{Y}.\hat{C}.a.enc_{Y2}$;}Yaksha

Network$(Y,\hat{C},\hat{S},H(a),H(C_a))\{$
/* *forwards the interested packet* */
receive $\hat{C}.a.s_{CY}$;
verify s_{CY},Q_C;
send $\hat{C}.a.s_{CY}$;
/* *forwards the content packet* */
receive $\hat{S}.a.s_{SY}.enc_{S1}$;
verify s_{SY},Q_S;
send $\hat{S}.a.s_{SY}.enc_{S1}$;}Network

8.2. Security Properties

The security objectives of ECY-CPS are of two types: authentication and secrecy. The authentication objectives take the form that a message of a certain format was indeed sent by some thread of the expected principal. The secrecy objectives take the form that a putative secret is known only to certain principals. Using the symbol of PCL, the security properties of ECY-CPS are described in table 3. Where $AUTH_{Yaksha}^{Source}$ states that when the thread S finishes executing the role Source(), some thread Y of principal $\hat{Y}$ indeed sent the expected message that contains the encrypted key k_a. $AUTH_{Consumer}^{Source}$ states that when the thread C of principal $\hat{C}$ finishes executing the role Consumer(), some thread S of principal $\hat{S}$ indeed publishes the content C_a that the content consumer is interested in. $sec_{k_a}^{Consumer}$ states that the key $k_a$is secret after execution of the consumer role by C; the other security properties are analogous.

Table 3. Security Properties of ECY-CPS

$AUTH_{YS}:\exists\eta_1.\text{Send}((\hat{Y},\eta_1),\hat{Y}\|\hat{S}\|SYMENC_{K_{SY}}\{n_{SY}\|SIG_{d_{SS}\times d_{SY}}\{H(C_a)\}\|k_a\})$
$AUTH_{YC}:\exists\eta_2.\text{Send}((\hat{Y},\eta_2),\hat{Y}\|\hat{C}\|SYMENC_{K_{CY}}\{n_{CY}\|SIG_{d_{CC}\times d_{CY}}\{H(a)\}\|k_a\})$
$AUTH_{CS}:\exists\eta_3.\text{Send}((\hat{S},\eta_3),\hat{S}\|a\|SIG_{d_{SS}\times d_{SY}}\{H(C_a)\}\|SYMENC_{k_a}\{C_a\|SIG_{d_{SS}\times d_{SY}}\{H(C_a)\}\})$
$AUTH_{yaksha}^{source}:\ [\text{source}]_S\text{Honest}(\hat{S},\hat{Y})\supset AUTH_{YS}$
$AUTH_{yaksha}^{consumer}:[\text{consumer}]_C\text{Honest}(\hat{C},\hat{Y})\supset AUTH_{YC}$
$AUTH_{consumer}^{source}:[\text{consumer}]_C\text{Honest}(\hat{C},\hat{Y})\supset AUTH_{CS}$
$SEC_{k_a}:\quad \text{Honest}(\hat{C},\hat{Y},\hat{S})\supset(\text{Has}(X,k_a)\supset\hat{X}\in\{\hat{C},\hat{Y},\hat{S}\})$
$SEC_{k_a}^{yaksha}:\ [\text{yaksha}]_Y SEC_{k_a}$, $\quad SEC_{k_a}^{source}:\ [\text{source}]_S SEC_{k_a}$
$SEC_{k_a}^{consumer}:\ [\text{consumer}]_C SEC_{k_a}$, $\quad SEC_{k_a}^{network}:\ [\text{network}]_N SEC_{k_a}$

8.3. Environment Assumptions

We assume that the EC-Yaksha private key of protocol participant only is known by the owner in our ECY-CPS system, namely they don't leave to any other participant. Using the symbol of PCL, this assumption is illustrated as follows:

$$\Gamma_1 : \forall X, Z.\text{Honest}(\hat{X}) \wedge (\text{Has}(Z, d_{XY}) \wedge \neg\text{Has}(Z, d_{XX})) \\ \vee(\text{Has}(Z, d_{XX}) \wedge \neg\text{Has}(Z, d_{XY})) \supset (\hat{Z} = \hat{X})$$

In addition, the master secret key K_{XY} shared between any user X and the EC-Yaksha server is also only known by the owner. This assumption is described as follows:

$$\Gamma_2 : \forall X, Y, Z.\text{Honest}(\hat{X}, \hat{Y}) \wedge (\text{Has}(Z, K_{XY}) \supset (\hat{Z} = \hat{X} \vee \hat{Z} = \hat{Y})$$

8.4. Security Proof

Theorem 1 (The encrypted Key k_a Authentication) On execution of the content publisher and the content consumer roles by the relative principal it is guaranteed that the intended EC-Yaksha server indeed sent expected response under the assumption $\text{Honest}(\hat{C}, \hat{S}, \hat{Y}, \hat{N})$. Formally,

$$\text{ECY - CPS} \succ AUTH_{Yaksha}^{Consumer}, AUTH_{Yaksha}^{Source}$$

Proof: we firstly prove that $\text{ECY - CPS} \succ AUTH_{Yaksha}^{Source}$. We assume that the thread S of the protocol participant $\hat{S}$ completely performs the role Source(), we make induction proof on basic sequence source_2 of Source():

$$\begin{aligned} \text{source}_2 = \{\ & \text{receive}\ \hat{Y}.\hat{S}.a.enc_{Y1}; \\ & text_{S1} := \text{symdec}\ enc_{Y1}, K_{SY}; \\ & \text{match}\ text_{S1}\ \text{as}\ n_{SY}.s_{SY}.k_a;\ \} \end{aligned}$$

According to the axiom ENC1, the fact, that the content publisher S can decrypt enc_{Y1} using the master secret key K_{SY} shared between S and the EC-Yaksha server, indicates that there exists the thread X of the protocol participant $\hat{X}$ generates and sends the message contains the ciphertextenc_{Y1} encrypted with the master secret key K_{SY}. On the environment assumption Γ_2, the assumption $\text{Honest}(\hat{C}, \hat{S}, \hat{Y}, \hat{N})$ and the honest rule HON, however, the protocol participant $\hat{X}$ possessed the master secret key K_{SY} must be the EC-Yaksha server. Such we prove that the equation as below:

$$\text{ECY - CPS} \succ AUTH_{Yaksha}^{Source}$$

Similar to the above proof process, we can conclude that

$$\text{ECY - CPS} \succ AUTH_{Yaksha}^{Consumer}.$$

Theorem 2 (The encrypted Key k_a Secrecy) On execution of the content consumer and the content publisher roles by the relative principal, secrecy of the encrypted key k_a is preserved under the assumption $\text{Honest}(\hat{C}, \hat{S}, \hat{Y}, \hat{N})$. Formally,

$$\text{ECY - CPS} \succ SEC_{k_a}^{Yaksha}, SEC_{k_a}^{Consumer}, SEC_{k_a}^{Source}, SEC_{k_a}^{Network}$$

Proof: under the protection of the set $\mathcal{K}$ of the master secret keys $\mathcal{K}=\{K_{SY}, K_{CY}\}$, the secrecy property of the key k_a can be proved. We firstly prove that the execution of the content consumer, the content publisher, the network or the EC-Yaksha server roles discharges the invariant assumptions Φ, that is:

$$[\text{Consumer}()]_C \quad \text{Honest}(\hat{C},\hat{S},\hat{Y},\hat{N}) \supset \Phi \qquad (1)$$

$$[\text{Source}()]_S \quad \text{Honest}(\hat{C},\hat{S},\hat{Y},\hat{N}) \supset \Phi \qquad (2)$$

$$[\text{Yaksha}()]_Y \quad \text{Honest}(\hat{C},\hat{S},\hat{Y},\hat{N}) \supset \Phi \qquad (3)$$

The invariant Φ of ECY-CPS scheme defines the operations that the thread Y generated the encrypted key k_a can do and can't. The assumed condition Φ is the conjunction of the following formulas Φ_i (i=1, 2, 3).

$$\Phi_1 : \forall X, m.\text{New}(X, k_a) \supset \neg(\text{Send}(X, m) \wedge \text{ContainsOpen}(m, k_a))$$

$$\Phi_2 : \forall X, \hat{S}_0, \hat{Y}_0.\text{New}(X, k_a) \wedge \text{SymEnc}(X, n_{S_0Y_0}.s_{S_0Y_0}.k_a, K_{S_0Y_0}) \supset \hat{X}=\hat{Y} \wedge \hat{S}_0=\hat{S}$$

$$\Phi_3 : \forall X, \hat{C}_0, \hat{Y}_0.\text{New}(X, k_a) \wedge \text{SymEnc}(X, n_{C_0Y_0}.s_{C_0Y_0}.k_a, K_{C_0Y_0}) \supset \hat{X}=\hat{Y} \wedge \hat{C}_0=\hat{C}$$

The predicate ContainsOpen(m,k_a) in Φ_1 asserts that k_a can be obtained from m by a series of un-pairings only no decryption required. In addition, assumes that some thread Y of $\hat{Y}$ generated the key k_a. Therefore, we can conclude that:

$$\text{KOHonest}(k_a, \mathcal{K}) \equiv \text{Honest}(\hat{Y},\hat{C},\hat{S},\hat{N})$$

Clearly, Φ is prefix closed and the set $\mathcal{K}$ of the master secret keys only can be accessed by the protocol participants $\hat{C},\hat{S},\hat{Y}$. These proofs of the predicate Φ_i (i=1, 2, 3) are derived from the authentication properties $AUTH_{Yaksha}^{Consumer}, AUTH_{Yaksha}^{Source}$.

Secondly, we show the proof of the following expression:

$$\text{ECY - CPS} \succ \text{Honest}(\hat{C},\hat{S},\hat{Y},\hat{N}) \wedge \Phi \supset \text{SafeNet}(k_a, \mathcal{K})$$

The above formula indicates that the message sent/received on the network and protected by the key set $\mathcal{K}$ is safe if the assumptions of $\text{Honest}(\hat{C},\hat{S},\hat{Y},\hat{N})$ and Φ is satisfied. The proof of this part uses the staged composition theorem (Theorem 6 in [23]).

Finally, the following conclusion can be inferred by combining the above two proof steps and using the rules NET, POS and POSL:

$$\text{ECY - CPS} \succ SEC_{k_a}^{Consumer}, SEC_{k_a}^{Source}, SEC_{k_a}^{Yaksha}, SEC_{k_a}^{Network}.$$

Theorem 3 (the content C_a authentication) On execution of the consumer role by a principal it is guaranteed that the intended publisher indeed sent the expected content C_a under the assumption $\text{Honest}(\hat{C},\hat{S},\hat{Y},\hat{N})$. Formally,

$$\text{ECY - CPS} \succ AUTH_{consumer}^{Source}$$

Proof: We assume that the thread C of protocol participant $\hat{C}$ completely performs the role Consumer(), we make induction proof on basic sequence consumer_2 of Consumer():

$$\begin{aligned}\text{consumer}_2 = \{\ & \text{receive } \hat{S}.a.s_{SY}.enc_{S1};\\ & text_{C2} := \text{symdec } enc_{S1}, k_a;\\ & \text{match } text_{C2} \text{ as } C_a.H(C_a).s_{SY};\\ & \text{verify } s_{SY}, Q_S;\ \ h_{C1} := \text{Hash}(C_a);\\ & \text{match } h_{C1} \text{ as } H(C_a);\}\end{aligned}$$

The fact, that the consumer can verify the publishing license using the content publisher S's public key Q_S, indicates that there exists the thread X of the protocol participant $\hat{X}$ generates this license (the joint signature) according the PCL axiom VER. On the environment assumption Γ_1 and the assumption $\text{Honest}(\hat{C}, \hat{S}, \hat{Y}, \hat{N})$, however, the protocol participant $\hat{X}$ possessed the EC-Yaksha server key d_{SS} must be the content publisher S.

In addition, the verification of hash value can prove the integrity of the content C_a. Such we prove the following equation:

$$\text{ECY - CPS} \succ AUTH_{Consumer}^{Source}$$

Finally, the confidentiality of the content C_a can be guaranteed by theorems 1-3 so long as the cryptographic schemes used in this new system are secure. □

We omit the detailed of proofs for theorems 1-3 because of length limitation.

9. Comparison of the Related Solution

We assume that some initialized cryptographic materials have been distributed. From a start point that generates the encrypted key for content, three schemes PRE-SCD [12], BE-SCD [14] and our system are detailedly compared in table 4. Finally, we conclude results in table 5. Table 5 is shown in the next page.

We think that these three schemes can be used in different environments. However, it is hard to distinguish their advantage/deficiency. The main problem that there exists in BE-SCD and our system can be the key breach. That is to say, the content consumer C can leak the decrypted key derived by computing or applying to illegal user, because the decrypted key for same content is same. Identity based proxy re-encryption used in PRE-SCD can efficiently overcome this issue, but it is required that the content producer must be always online. However, it can be most important benefit for our CCMANET environment that any data source (Multi-Source) is allowed to publish content in PRE-SCD and our system. In addition, our solution can defend interest flooding and content pollution attacks because of using the license, and it is allowed that EC-Yaksha server monitors content transmitted on the network. BE-SCD is better fit for an environment relied on CDN.

Table 4. The Related Solution Comparison

		NCO	NEM
The Content Encrypted Key Generation Stage			
BE-SCD	Sends secret share T_j=<x_j, $f(x_j)$> to each user u_j, T_j is encrypted using u_j's public key.	t encryptions	t
PRE-SCD	Don't anything	0	0
ECY-CPS	Don't anything	0	0
Content Publish			
BE-SCD	The content producer firstly generates the Enabling Block (BE) and the encrypted key τ, and then it publishes the encrypted content to the network along with BE, BE contains information for any legitimate user u_j to extract the encrypted key τ using it's secret share T_j.	1 encryption and 1 signature.	1
PRE-SCD	The content producer publishes the content encrypted using the key sk_{id_S} of the respective producer identity.	1 encryption and 1 signature.	1
ECY-CPS	The content producer firstly applies for the encrypted key k_a and the publishing license from EC-Yaksha, and then publishes the license and the encrypted content encrypted using the key k_a.	1 encryption and 2 signatures.	3
Content Subscribe			
BE-SCD	The content consumer broadcasts the interest packet contains its' signature.	1 signature	1
PRE-SCD	The content consumer broadcasts the interest packet contains its' signature.	1 signature	1
ECY-CPS	The content consumer firstly applies for the decrypted key k_a and the subscribing license from EC-Yaksha, and then broadcasts the interest packet contains the subscribing license.	1 encryption and 2 signatures.	3
Content Fetch			
BE-SCD	The user u_j (a content consumer) firstly computes the decrypted key τ from T_j and BE using the lagrangian interpolation method, and then decrypts the encrypted content using the secret key τ.	1 decryption 1 signature verification	0
PRE-SCD	The content consumer firstly requests re-encryption key $rK_{id_C \to id_S}$, and then encrypted the encrypted content using the re-encryption key $rK_{id_C \to id_S}$ again. Now it can decrypt to obtain the content using the key sk_{id_C} of the respective its identity.	2 encryptions, 1 decryption, 2 signatures and 2 signature verifications	2
ECY-CPS	The content consumer decrypts the encrypted content using k_a acquired in content subscribing stage.	1 decryption and 1 signature verification	0

Table 5. Summary of the Related Solution

Protocol	NCO	NEM	OO	MS	RS	DKB
BE-SCD	t+5	t+2	N	N	Y	Y
PRE-SCD	10	4	Y	Y	N	N
ECY-CPS	8	6	N	Y	Y	Y

Note: indication of the symbols used in table 4 and table 5 is shown as follows. NCO, Number of Cryptographic Operations; NEM, Number of Exchange Message; OO, The data source is required Online or Offline; MS, Multi-Source; RS, Rely on the Server; DKB, The Decrypted key Breach. t is threshold of allowed revoked users in BE-SCD.

10. Conclusion

In this paper, we firstly develop an EC-Yaksha system. And then, a secure content publish/subscribe system ECY-CPS for CCMANET is proposed. We prove security properties of ECY-CPS in PCL and make a comparison analysis between existing solution and our system. We find three schemes can be used in different scenes. Our system has three benefits: (1) The EC-Yaksha server can monitor content transmitted on the network. (2) Our system doesn't require that the content producer is always online, and it also may don't rely on the internet infrastructure or CDN. (3) Our system allows any legitimate data source publishes content. In addition, it is one of our future works that implements our system and make experiment analysis in CCNx [4].

Acknowledgments. This work is supported by NSFCNo.61461027, No. 61462060, No. 71303023; Gansu province science and technology plan project under grant No. 145RJZA078, 1308RJZA277; Specialized research fund for the doctoral program of Lanzhou University of Technology; We thank the referees for helpful comments.

References

1. Cisco: Cisco visual network index, Cisco Systems, Inc, San Jose, CA (2014). [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/vni-service-adoption-forecast/Cisco_VNI_SA_Forecast_WP.pdf
2. Xylomenos, G., Ververidis, C.N., Siris, V.A., Fotiou, N.: A Survey of Information-Centric Networking Research. Communications Surveys and Tutorials, Vol. 16, No. 2, 1024-1049. (2014)
3. Ahlgren, B., Dannewitz, C., Imbrenda, C.: A Survey of Information-Centric Networking. IEEE Communications Magazine, Vol. 50, No. 7, 26-36. (2012)
4. PARC: Content Centric Networking project, PARC, Palo Alto, California, USA. [Online]. Available: http://www.ccnx.org/
5. UCLA: NSF Named Data Networking project, University of California, Los Angeles, USA. [Online]. Available: http://www.named-data.net/
6. Amadeo, M., Campolo, C., Nolinaro, A.: Forwarding strategies in named data wireless ad hoc networks: Design and evaluation. Journal of Network and Computer Applications, Vol. 50, 148-158. (2015)
7. Tyson, G., Sastry, N., Cuevas, R., Rimac, I., Mauthe, A.: A Survey of Mobility in Information-Centric Networks. Communications of the ACM, Vol. 56, No. 12, 90-98. (2013)

8. Zhang, Y., Zhang, H. L., lixia Zhang.: Kite: A Mobility Support Scheme for NDN. In Proceeding of the 1st international conference on information-centric networks, ACM press, Paric, France, 179-180. (2014)
9. Wang, L., Afanasyev, A., Kunts, R., Vuyyuru, R., Wakikawa, R., Zhang, L.: Rapid Traffic Information Dissemination Using Named Data. In Proceeding of ACM Workshop on Emerging Name-Oriented Mobile Networking Design-Architecture, Algorithms, and Applications. ACM press, Holton Head Island, South Carolina, 7-12.(2012)
10. Oh, S. Y., Lau, D., Gerla, M.: Content Centric Networking in Tactical and Emergency MANETs. In Proceeding of Wireless days, IFIP 2010, IEEE press, Venice, Italy, 1-5. (2010)
11. Amadeo, M., Molinaro, A., Ruggeri, G.: E-CHANET: Routing, Forwarding and Transport in Information-Centric Multi hop Wireless Networks. Elsevier Computer Communications, Vol. 36, No. 7, 792-803. (2013)
12. Wood, C.A., Uzun, E.: Flexible end-to-end Content Security in CCN. In Proceeding of Consumer Communications and Networking Conference (CNCC), IEEE press, Las Vegas, NV, 858-865. (2014)
13. Blaze, M., Bleumer, G., Strauss, M.: Divertible Protocols and Atomic Proxy Cryptography. In proceeding of Advances in Cryptology-EUROCRYPT'98, Lecture Notes in Computer Science, Vol. 1403. Springer-Verlag, Berlin Heidelberg New York, 127-144. (1998)
14. Misra, S., Tourani, R., Majd, N. E.: Secure content delivery in information-centric networks: design, implementation, and analyses. In Proceeding of the 3rd ACM SIGCOMM workshop on Information-centric networking, ACM press, Hong Kong, China, 73-78. (2013)
15. Duan, Y., Canny, J.: Scalable secure bidirectional group communication. In Proceeding of 26th IEEE international Conference on Computer Communications, IEEE press, Anchorage, AK, 875-883. (2007)
16. Fiat, A., Naor, M.: Broadcast encryption. In proceeding of Advances in Cryptology – CRYPTO '93, Lecture Notes in Computer Science, Vol. 773, Springer-Verlag, Berlin Heidelberg New York, 480-491. (1994)
17. AbdAllah, E. G, Hassanein, H. S., Zulkernine, M.: A survey of Security Attacks in Information-Centric Networking. IEEE Communications Surveys & Tutorials, Vol. PP, No. 99, 1.(2015)
18. Chaabane, A., Cristofaro, E. D., Kaafa, M. A., Uzun, E.: Privacy in Content-Oriented Networking: Threats and Countermeasures. ACM SIGCOMM Computer Communication Review, Vol. 43, No. 3, 25-33. (2013)
19. Ganesan R.: The Yaksha Security System. Communication of the ACM, Vol. 39, No. 3, 55-60. (1996)
20. Rivest, R. L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Communications of ACM, Vol. 21, No. 2, 96-99. (1978)
21. IEEE P1363. Standard specifications for public-key cryptography. Draft version 7. (1998)
22. AnupamDatta, Ante Derek, John C. Mitchell.: Protocol Composition Logic (PCL). Electronic Notes in Theoretical Computer Science, Vol. 172, 311-358.(2007)
23. Roy, A., Datta, A., Derek, A., Mitchell, J. C.: Secrecy Analysis in Protocol Composition Logic. In proceeding of 11th Asian Computing Science Conference, Lecture Notes in Computer Science, Vol. 4435, Springer-Verlag, Berlin Heidelberg New York. 197-213. (2007)
24. Feng, T., Han, S. N., Guo, X., Ma, D. L.: A new method of formalizing anonymity based on protocol composition logic. Security and Communication Networks, Vol. 8, No. 6, 1132-1140. (2015)

Xian Guo is an Associate Professor at the School of Computer and Communication, Lanzhou University of Technology, China. He received his BSc from Northwest Normal University, China, and PhD from Lanzhou University of Technology, China. Dr. Guo's Current research focuses on network and information security, future internet

architecture and security analysis. His research projects are supported by a number of provincial and national research funding agencies. He is a senior member of China Computer Federation.

Tao Feng is a Professor at the School of Computer and Communication, Lanzhou University of Technology, China. He received his BSc from Lanzhou University of Technology, China, and PhD from Xidian University, China. Dr. Feng's Current research focuses on network and information security.

Cheng Chen is a postgraduate at the School of Computer and Communication, Lanzhou University of Technology, China. His research interest focuses on network and information security.

Tong Liu is an associate research fellow of Beijing science and technology academy. She had her postdoc research in Tsinghua university. Her research field is information analysis and information system design. She was selected in Beijing Sci-tech New Star plan. She is the undertaker of National nature science foundation project, Beijing science and technology plans project, the Ministry of science and technology innovation fund for small and medium sized enterprises and other 17 national level or ministerial level subjects.

Received: March 20, 2015; Accepted: September 10, 2015.