Security Analysis of the Access Control Solution of NDN Using BAN Logic

Yuan Fei Huibiao Zhu* Huiwen Wang
Shanghai Key Laboratory of Trustworthy Computing,
School of Computer Science and Software Engineering,
East China Normal University, Shanghai, China

***Abstract*—Named Data Networking (NDN) is a new promising architecture of information-centric networking. For its caching property, traditional mechanisms of access control can no longer work. Hamdane et al. propose a new access control solution for both closed and open environments. In this paper, we make the very first attempt to formally analyze this access control solution. Inspired by the basic BAN logic which is often used to describe protocols by logical formulas, we present our BAN-like logic by adding some new notions to make it suitable for the access control solution. Using the BAN-like logic, the procedures of the access control solution is idealized in the form of the beliefs of principals. Then the idealized procedures are analyzed under several security goals with a set of logical postulates. Several unsatisfied goals may lead the access control solution to be vulnerable to intruders. We give the modification in the idealized procedures to archive more goals. We also present the related modification in the implementation of the access control solution. Our study helps to improve security and protect against various attacks for the access control solution.**

***Keywords*—Named Data Networking (NDN), Access Control Solution, BAN Logic**

I. Introduction

Named Data Networking (NDN) [1] is an architecture of Information-Centric Networking (ICN). ICN aims to offer solutions to problems existing in TCP/IP Internet. Nowadays users pay more attention to named content rather than its location. Though TCP/IP Internet has shown great resilience over the years, it cannot support the newly evolving content distribution model successfully. One of the promising candidates of ICN is NDN, which supports multicast of data and adopts the publish/subscribe model. The data producers mean publishers and the data consumers represent subscribers in NDN. When data consumer needs data, it sends out an *Interest* packet with a required name of the data; according to the name, routers forward the packet over the network; and a *Data* packet is returned to the consumer when a data produced by the data producer is matched. NDN routers can cache previous forwarded *Data* packets, which are able to be reused when a matching *Interest* packet comes. For this caching property, traditional mechanisms of access control, as a way of limiting access to data, can no longer work. Some access control specifications [2], [3] are proposed for NDN. However, each owns several limits. Hamdane et al. [4] put forward a new access control solution to address these limits. We focus on this solution and analyze it step by step with our BAN-like logic.

*Corresponding Author. E-mail address: hbzhu@sei.ecnu.edu.cn (H. Zhu).

The BAN logic [5] was first proposed by Burrows, Abadi and Needham in 1989. It provides a way to formalize the description and analysis of authentication protocols. It has been applied to analyze existing protocols and to find out their flaws [5]. Gaarder et al. [6] introduced new notions based on the basic BAN logic specially for PKCS authentication protocols. In order to reason about cryptographic protocols, Gong et al. [7] added more accurate concepts and definitions to the basic BAN logic. By adding negation, [8] presented the special BAN logic designed for monotonic protocols.

Our BAN-like logic is inspired by the basic BAN logic. We add some new notions to make it suitable for the access control solution. Adopting the BAN-like logic, the procedures of the access control solution in [4] is idealized. Then the idealized procedures are analyzed under several security goals with a set of logical postulates. The results show that some goals could not be archived. It indicates that this access control solution cannot ensure the source of critical keys and data. This may lead the access control solution to be vulnerable to intruders. We give the modification of the idealized procedures and the proofs of them. Meanwhile we also present the related modification in the implementation of the access control solution. Our study helps to improve security and protect against various attacks for the access control solution.

The rest of the paper is organized as follows: Section II briefly introduces the access control solution of NDN and explains how it works. Section III gives the BAN-like logic definitions and the assumptions of the access control solution. In Section IV and Section V, we apply BAN-like logic to analyze the access control solution of write and read operations in a closed environment. Section VI displays the analysis of write and read operation in an open environment. Finally, Section VII concludes and points out the future work.

II. Access control solution of NDN

In this section, we introduce the access control solution [4] for NDN. We give the related entities and assumptions. The write operations and read operations in closed and open environments are proposed.

DOI reference number: 10.18293/SEKE2018-124

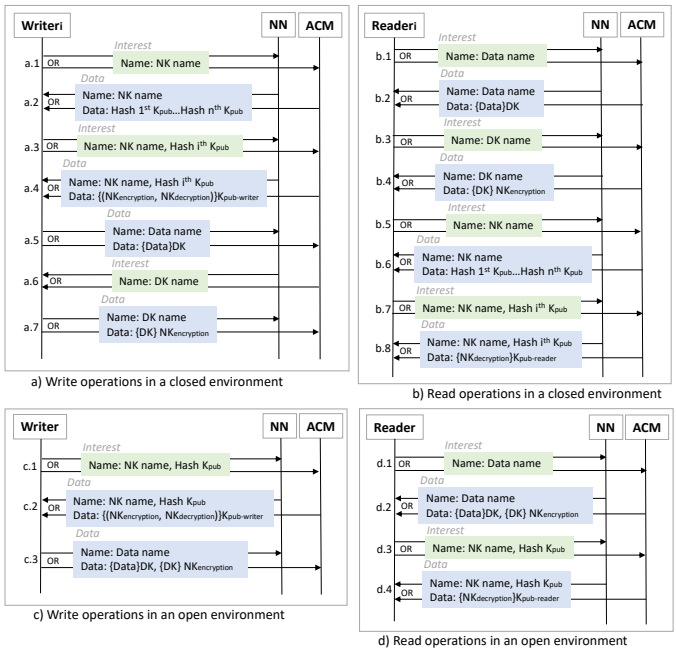

Fig. 1. Write and Read operations in closed and open environments (simplified from Hamdane et al. [4])

A. Entities and Assumptions

Entities in NDN own two roles: readers and writers. An access control manager (ACM) is introduced to control the management of the access control policy. It creates a key pair ($NK_{encryption}$, $NK_{decryption}$), which is designed for encrypting a symmetric data key *DK*. These keys are similar as public and private keys, but both of them are secret. $NK_{encryption}$ is applied to encrypt the *DK* and only be acquired by entities with write privilege. $NK_{decryption}$ allows the *DK* decryption which can be obtained by any entities. Meanwhile, every entity owns the related public and private key pair. The public writer keys are used to encrypt key pair ($NK_{encryption}$, $NK_{decryption}$), and public reader keys are adopted to encrypt key $NK_{encryption}$.

B. Write Operations and Read Operations

We simplify some steps of write and read operations in [4] and only retain the processing related with keys and data. Fig.1 illustrates four situations of write and read operations in different environments. Because the node playing reader's or writers role can be connected with a normal node or the ACM, the communication object of $Writer_i$ and $Reader_i$ can be *ACM* or *NN*. *NN* is only responsible to transit message to *ACM* eventually. As a result, $Writer_i$ and $Reader_i$ are communicating with *ACM* essentially.

1) Write operations in a closed environment: Assuming that $Writer_i$ knows the name of key pair ($NK_{encryption}$, $NK_{decryption}$) in advance, it sends an *Interest* packet containing *NK name* to *ACM* (step a.1). *ACM* returns a *Data* packet with all the hash value of public keys that *ACM* has already known (step a.2). Then $Writer_i$ recognizes its own hash and transmits it together with *NK name* as a new *Interest* to *ACM* (step a.3). *ACM* recognizes which writer is communicating with it, and uses $Writer_i$'s public key to encrypt key pair ($NK_{encryption}$, $NK_{decryption}$). This is added to a new *Data* packet which is fed back to *ACM* (step a.4). Then $Writer_i$ sends the encrypted data *DataDK* to *ACM* (step a.5). *ACM* learns *DK name* and sends the *Interest* to *ACM* (step a.6). *ACM* then uses $NK_{encryption}$ to encrypt *DK* and produces a new *Data* packet as a reply (step a.7).

2) Read operations in a closed environment: $Reader_i$ sends the required *Data name* to *ACM* (step b.1). *ACM* replies $Reader_i$ with *Data* packet including *Data* encrypted with data key *DK* (step b.2). $Reader_i$ knows the name of data key *DK* and sends the *Interest* to *ACM* (step b.3). *ACM* uses $NK_{encryption}$ to encrypt *DK* and creates a *Data* packet to send back to $Reader_i$ (step b.4). $Reader_i$ sends the *Interest* packet carrying name of key pair ($NK_{encryption}$, $NK_{decryption}$) to *ACM* (step b.5). *ACM* returns all the hash value of public keys (step b.6). $Reader_i$ gives *ACM* with *NK name* and its hash value of public key (step b.7). *ACM* uses the related public key to encrypt $NK_{decryption}$ to produce a *Data* packet for *ACM* (step b.8).

3) Writer operations in an open environment: As the environment is open, *Writer* is unknown to *ACM*. As a result, *Writer* needs to send *ACM* its hash value of public key. *ACM* can get the related public key to encrypt information. First, *Writer* sends *NK name* and the hash value of its public key to *ACM* (step c.1). As *ACM* has acknowledged the public key of *Writer*, it uses the key to encrypt key pair ($NK_{encryption}$, $NK_{decryption}$) (step c.2). *Writer* sends the encrypted data $\{Data\}DK$, together with encrypted data key $\{DK\}$ $NK_{encryption}$ (step c.3).

4) Read operations in an open environment: Because the environment is open, *ACM* does not acknowledge the public key of *Reader*. So it is necessary for *Reader* to transmit the hash value of its public key to *ACM*. First, *Reader* sends an *Interest* with *Data name* to *ACM* (step d.1). *ACM* responses the encrypted data $\{Data\}DK$ and the encrypted data key $\{DK\}$ $NK_{encryption}$ (step d.2). *Reader* transmits *NK name* and its hash value of public key to *ACM* (step d.3). Because *ACM* realizes the public key of *Reader*, it applies this key to encrypt $NK_{decryption}$ which is built into a *Data* packet and conveyed to *Reader* (step d.4).

III. An Introduction of BAN-like Logic and Assumptions

In this section, we introduce our BAN-like logic inspired by the basic BAN logic. We add some new notions to make it suitable for the access control solution. Assumptions of access control solution are also presented for further analysis.

A. Statements

There are three sorts of objects in our BAN-like logic: principals, keys, and formulas (also statements). The symbols *P* and *Q* range over principles; *K* ranges over keys; *X* ranges over formulas. The following are basic statements.

- $P \models X$: The principal *P* believes the statement *X* is true.

- $P \triangleleft X$: P sees X, which represents P has received a message X.
- $P \mid\sim X$: P once said X, which also means $P|\equiv X$ when P sent it.
- $P \Rightarrow X$: P governs X, showing that P has an authority on X.
- $\sharp(X)$: The message X is fresh.

There is little difference between the keys in the basic BAN logic and the key used here. We add new notations for a better explanation. The key K represents the public key of asymmetric keys, which can be seen by other agents. Its associated private key K^{-1} will be secret to any other agents except one agent which owns it. The key pair (@K,K@) denotes an asymmetric key pair, in which @K is used for encryption and K@ is devoted to decryption. An agent can only acquire this key pair from the package it received, if it is not the creator of the pair. The symmetric key $\$K$ is applied for both encryption and decryption, which is encrypted and transmitted between agents.

- $\overset{K}{\longmapsto} P$: The encryption key K is the public key of P. The matching private key K^{-1} will be secret to any other principals except P.
- $\overset{@K}{\longmapsto} P$: The key @$K$ is the encryption key of the asymmetric key pair (@K,K@), which is acknowledged by P.
- $\overset{K@}{\longmapsto} P$: P knows the key K@, which is the decryption key of the asymmetric key pair (@K,K@).
- $\overset{\$K}{\longmapsto} P$: P learns that the key $\$K$ is a symmetric key.
- $\{X\}_K$:The statement X is encrypted under the key K. K can also be replaced by @K or $\$K$.

B. Logical Postulates

We introduce four categories of postulates and give their explanations.
(1) The *message-meaning* rules are about interpretation of encrypted messages. We postulate the *message-meaning* rule for symmetric keys as below.

$$\textbf{MM1} \qquad \frac{P \models \overset{\$K}{\longmapsto} P, Q \models \overset{\$K}{\longmapsto} Q, P \triangleleft \{X\}_{\$K}}{P \models Q \mid\sim X}$$

If both P and Q believe that the key K is the symmetric key and P sees a message X encrypted under K, then P believes that Q once said X.

We also present the *message-meaning* rule for asymmetric keys as below.

$$\textbf{MM2} \qquad \frac{P \models \overset{K}{\longmapsto} Q, P \triangleleft \{X\}_{K^{-1}}}{P \models Q \mid\sim X}$$

If P believes that the key K is the public key of Q and sees a message X encrypted under K^{-1}, then P believes that Q once said X.
(2) The *nonce-verification* rule states the check of the freshness of a message.

$$\textbf{NV} \qquad \frac{P \models \sharp(X), P \models Q \mid\sim X}{P \models Q \models X}$$

If P believes a formula X is fresh and P believes that Q once said formula X, then P believes that Q believes X.
(3) The *jurisdiction* rule expresses how jurisdiction effects the belief.

$$\textbf{J} \qquad \frac{P \models Q \Rightarrow X, P \models Q \models X}{P \models X}$$

If P believes that Q has jurisdiction over X and Q believes X, then P trusts X.
(4) The *seeing* rule describes the situation when a principal sees a formula encrypted with different kinds of keys.

$$\textbf{SEE1} \qquad \frac{P \triangleleft \{X\}_K, P \models \overset{K}{\longmapsto} P}{P \triangleleft X}$$

If P sees X encrypted with a public key K and processes the corresponding private key K^{-1}, then P is considered to have seen X.

$$\textbf{SEE2} \qquad \frac{P \triangleleft \{X\}_{K^{-1}}, P \models \overset{K}{\longmapsto} Q}{P \triangleleft X}$$

If P sees X encrypted with a private key K^{-1} and owns the corresponding public key K, then P is regarded as seeing X.

$$\textbf{SEE3} \qquad \frac{P \triangleleft \{X\}_{@K}, P \models \overset{K@}{\longmapsto} P}{P \triangleleft X}$$

If P sees X encrypted with an encryption key @K of a key pair *(@K,K@)* and has the decryption key K@, then P is thought of as seeing X.

$$\textbf{SEE4} \qquad \frac{P \triangleleft \{X\}_{\$K}, P \models \overset{\$K}{\longmapsto} P}{P \triangleleft X}$$

If P sees X encrypted with a known symmetric key $\$K$, then P sees X.

C. Assumptions of access control solution

We refer principals to the Access Control Manager (ACM), readers and writers, which are presented by symbols M, R_1,...,R_n and W_1,...,W_n. KW_1,...,KW_n and KR_1,...,KR_n denote the public keys of writers W_1,...,W_n and readers R_1,...,R_n respectively. KW_1^{-1},...,KW_n^{-1} and KR_1^{-1},...,KR_n^{-1} represent the corresponding private keys. *(@NK,NK@)* is the asymmetric key pair produced by ACM M. *$DK* and *Data* is the symmetric key and data created by writers.

To analyze the access control solution, we first list the following assumptions.

A1: $M \backslash W_i \backslash R_i \models \overset{K_j}{\longmapsto} W_j \backslash R_j$ **A2:** $W_i \models \overset{\$DK}{\longmapsto} W_i$
A3: $M \models \overset{@NK}{\longmapsto} M$ **A4:** $M \models \overset{NK@}{\longmapsto} M$
A5: $W_i \backslash R_i \models M \Rightarrow (@NK,\ NK@)$ **A6:** $M \backslash R_i \models W_j \Rightarrow \$DK$
A7: $M \backslash R_i \models W_j \Rightarrow Data$ **A8:** $W_i \models \sharp(\$DK)$ **A9:** $W_i \models \sharp(Data)$
A10: $P \triangleleft NK@ \rightarrow P \models \sharp(NK@)$ **A11:** $P \triangleleft NK@ \rightarrow P \models \overset{NK@}{\longmapsto} P$
A12: $P \triangleleft (@NK,NK@) \rightarrow P \models \sharp(@NK,NK@)$
A13: $P \triangleleft (@NK,NK@) \rightarrow P \models \overset{@NK}{\longmapsto} P$
A14: $P \triangleleft (@NK,NK@) \rightarrow P \models \overset{NK@}{\longmapsto} P$
A15: $P \triangleleft (\$DK) \rightarrow P \models \sharp(\$DK)$ **A16:** $P \triangleleft (\$DK) \rightarrow P \models \overset{\$DK}{\longmapsto} P$
A17: $P \triangleleft (Data) \rightarrow P \models \sharp(Data)$

Assumptions **A1**-**A4** are about the keys initially known to the principals. Assumptions **A5**-**A7** describe that M is trusted by W_i to make key pair (@NK,NK@) and M trusts that W_i can produce data key *$DK* and data *Data*. Assumptions **A8** and **A9** indicate that W_i believes data *Data* and data key *$DK* are fresh. Assumptions **A10**-**A16** present the situation when a principle sees different keys. The principle P could be replaced by M, R_i and W_i. P learns the related key which is also considered to be fresh. **A17** shows when P sees data *Data* it also confirms the freshness of it.

IV. Analysis of Write operation in a closed environment

In this section, we introduce the specific write operation in a closed environment. Then we apply our BAN-like logic to this procedure. Several security goals are listed to be proved.

A. Write operation in a closed environment

We give the write operation procedure in a closed environment according to Fig.1.(a) as below.

Message 1. $Writer_i \rightarrow ACM : name_{NK}$
Message 2. $ACM \rightarrow Writer_i : name_{NK}, H(K_1)...H(K_n)$
Message 3. $Writer_i \rightarrow ACM : name_{NK}, H(K_i)$
Message 4. $ACM \rightarrow Writer_i : name_{NK}, H(K_i), \{NK_e, NK_d\}_{K_i}$
Message 5. $Writer_i \rightarrow ACM : name_{Data}, \{Data\}_{DK}$
Message 6. $ACM \rightarrow Writer_i : name_{DK}$
Message 7. $Writer_i \rightarrow ACM : name_{DK}, \{DK\}_{NK_e}$

$Writer_i$ sends the name of key *NK* to *ACM*. *ACM* returns $name_{NK}$ together with the hash values of $K_1...K_N$, denoted by $H(K_1)...H(K_N)$. Then $Writer_i$ recognizes the correct hash values of K_i in the hash value sequences and feedbacks it to *ACM* with $name_{NK}$. After receiving the package, *ACM* returns a new message added with the corresponding key pair (NK_e,NK_d). $Writer_i$ uses its special data key *DK* to encrypt the data and sends it to *ACM* for storing. *ACM* can learn the name of data key *DK* and send it to $Writer_i$. Finally, $Writer_i$ returns the message of *DK* encrypted by key NK_e with $name_{DK}$. This can lead *ACM* to acknowledge data key *DK*, which is used to decrypt message $\{Data\}_{DK}$.

B. Analysis of security goals of asymmetric key pair

In order to idealize the procedure, we abstract all the forwarding encrypted messages, modify the forms of keys, and change the expression of formulas in our BAN-like logic as below.

M1: $W_i \lhd \{(@NK, NK@)\}_{KW_i}$
M2: $M \lhd \{Data\}_{\$DK}$
M3: $M \lhd \{\$DK\}_{@NK}$

We hope the procedure should archive several security goals when distributing (@*NK*,*NK*@), *Data* and *$DK*. Considering (@*NK*,*NK*@), there are three authentication goals described in formulas: $W_i \lhd$ (@*NK*,*NK*@), $W_i \models M \models$ (@*NK*,*NK*@) and $W_i \models$ (@*NK*,*NK*@). These mean that W_i should see key pair (@*NK*,*NK*@), W_i also believes that *M* believes the key pair, and W_i believes the key pair respectively.

The first goal can be proved easily. Applying the seeing rule **SEE1** to message **M1** and assumption **A1** yields

S1: $W_i \lhd (@NK, NK@)$.

Now we hence proved that the procedure has achieved the first goal. But we cannot keep carrying forward, as the next two goals need more information. The crux of proving the two remaining goals is to achieve $W_i \models M \mid\sim (@NK, NK@)$ using the message-meaning rule **MM2**. Hence, we need a pair of asymmetric keys for the application of this rule. We assume that *M* owns a public key K_M and a corresponding private key K_M^{-1} used for its signatures. As a result, we do the modification to the idealized procedure. Two new assumptions are added as below.

A'1: $W_i \models \xrightarrow{K_M} M$

A'2: $M \models \xrightarrow{K_M} M$

As the key pair (NK_e,NK_d) is produced by *M*, message **M1** is changed as below.

M1': $W_i \lhd \{\{@NK, NK@\}_{K_M^{-1}}\}_{KW_i}$

Again applying **SEE1** to message **M1'** and assumption **A1** yields

S2: $W_i \lhd \{@NK, NK@\}_{K_M^{-1}}$.

Using the seeing rule **SEE2** to **S2** with assumption **A'1** also produces

S1: $W_i \lhd (@NK, NK@)$.

Then employing the message-meaning rule **MM2** to **S2** and assumption **A'1** gets

S3: $W_i \models M \mid\sim (@NK, NK@)$.

As BAN logic defaults to using Modus Ponens (MP) rule, adopting MP to **S1** and assumption **A12** obtains

S4: $W_i \models \sharp(@NK,NK@)$.

Utilizing the nonce-verification rule **NV** to **S3** and **S4** obtains

S5: $W_i \models M \models (@NK,NK@)$.

Applying the jurisdiction rule **J** to assumption **A5** and **S5** yields

S6: $W_i \models (@NK,NK@)$.

We can conclude that the original formulas can only archive the first goal $W_i \lhd$ (NK_e,NK_d). After our modification, the next two goals $W_i \models M \models$ (NK_e,NK_d) and $W_i \models$ (NK_e,NK_d) can also be satisfied. Hence, in the implementation of the access control solution in Fig.1 (step a.4), the data domain of *Data* packet should be changed from $\{(NK_{encryption}, NK_{decryption})\}K_{pub-writer}$ to $\{\{(NK_{encryption}, NK_{decryption})\}K_M^{-1}\}K_{pub-writer}$.

C. Analysis of security goals of data key

The first series of goals should be archived are about data key *$DK*: $M \lhd \$DK$, $M \models W_i \models \$DK$ and $M \models \$DK$.

Applying the seeing rule **SEE4** to message **M3** and assumption **A5** yields

S7: $M \lhd \$DK$.

In order to push further, we need to obtain the formula $M \models W_i \mid\sim \$DK$ concerning with the message-meaning rule **MM2**. As a result, we need to utilize the private key KW_i^{-1} to encrypt *$DK* first. We also make a little change to the idealized procedure. Message **M3** is altered as below.

M3': $M \lhd \{\{\$DK\}_{KW_i^{-1}}\}_{@NK}$

Utilizing the seeing rule **SEE3** to message **M3'** and assumption **A4** gains

S8: $M \lhd \{\$DK\}_{KW_i^{-1}}$.

Using the seeing rule **SEE3** to **S8** with assumption **A1** produces

S7: $M \lhd \$DK$.

Applying the message-meaning rule **MM2** to **S8** and assumption **A1** yields

S9: $M \models W_i \mid\sim \$DK$.

Adopting the MP rule to **S7** and assumption **A15** gets

S10: $M \models \sharp(\$DK)$.

Utilizing the nonce-verification rule **NV** to **S9** and **S10** gains

S11: $M \models W_i \models \$DK$.

Applying the jurisdiction rule **J** to **S11** and assumption **A6** yields

S12: $M \models \$DK$.

By this point, we have figured out the the formal proof of three goals for data key *DK*. As a result, in the implementation of the access control solution in Fig.1 (step a.7), the data domain of *Data* packet should be changed from $\{DK\}NK_{encryption}$ to $\{\{DK\}K_{pri-writer}\}NK_{encryption}$.

D. Analysis of security goals of data

We focus on the other three goals for *Data* which represents the real data the writer wants to publish. They are $M \triangleleft Data$, $M \models W_i \models Data$ and $M \models Data$. These denote that *M* should see *Data*, *M* also believes that W_i believes *Data*, and *M* believes *Data* respectively.

Similarly, we can easily apply the MP rule to assumption **A15** and **S7** to get

S13: $M \models \overset{\$DK}{\longmapsto} M$.

The seeing rule is also adopted to **S13** and message **M2** to gain

S14: $M \triangleleft Data$.

For further proof, we are required to gain the formula $M \models W_i \mid\sim Data$ springing from the message-meaning rule **MM2**. So we choose to apply the private key KW_i^{-1} to encrypt *$Data* first. Message **M2** is changed as below.

M2'. $M \triangleleft \{\{Data\}_{KW_i^{-1}}\}_{\$DK}$

Utilizing the seeing rule **SEE4** to message **M2'** and **S13** gains

S15: $M \triangleleft \{Data\}_{KW_i^{-1}}$.

Applying the seeing rule **SEE2** to **S15** and assumption **A1** yields

S14: $M \triangleleft Data$.

Employing the message-meaning rule **MM2** to **S15** according to assumption **A1** gets

S16: $M \models W_i \mid\sim Data$.

Using the MP rule to **S14** with assumption **A16** produces

S17: $M \models \sharp(Data)$

Applying the nonce-verification rule **NV** to **S16** and **S17** yields

S18: $M \models W_i \models Data$.

Adopting the jurisdiction rule **J** to assumption **A7** and **S18** gets

S19: $M \models Data$.

Now we have settled the formal proof of the three goals of data *Data*. As a result, in the implement of the access control solution in Fig.1 (step a.5), the data domain of *Data* packet should be changed from $\{Data\}DK$ to $\{\{Data\}K_{pri-writer}\}DK$.

V. Analysis of Read operation in a closed environment

In this section, the specific read operation in a closed environment is presented. We adopt our BAN-like logic to this procedure and analyze it with some important security goals.

A. Read operation in a closed environment

We demonstrate the read operation procedure in a closed environment according to Fig.1.(b) as below.

Message 1. $Reader_i \rightarrow ACM : name_{Data}$
Message 2. $ACM \rightarrow Reader_i : name_{Data}, \{Data\}_{DK}$
Message 3. $Reader_i \rightarrow ACM : name_{DK}$
Message 4. $ACM \rightarrow Reader_i : name_{DK}, \{DK\}_{NK_e}$
Message 5. $Reader_i \rightarrow ACM : name_{NK}$
Message 6. $ACM \rightarrow Reader_i : name_{NK}, H(K_1)...H(K_n)$
Message 7. $Reader_i \rightarrow ACM : name_{NK}, H(K_i)$
Message 8. $ACM \rightarrow Reader_i : name_{NK}, H(K_i), \{NK_d\}_{K_i}$

$Reader_i$ sends the name of *Data* to *ACM*. *ACM* returns *Data* encrypted with data key *DK*. In order to decrypt it, $Reader_i$ sends $name_{DK}$ to *ACM* to request for data key *DK*. *ACM* gives $Reader_i$ with *DK* encrypted with key NK_e. As a result, $Reader_i$ still needs to deliver $name_{NK}$ to get decryption key NK_d. *ACM* returns $name_{NK}$ together with the hash values of $K_1...K_N$, denoted by $H(K_1)...H(K_N)$. Then $Reader_i$ recognizes the correct hash values of K_i in the hash value sequences and feeds it back to *ACM* with $name_{NK}$. Finally, *ACM* returns a new message added with NK_d encrypted with public key K_i. NK_d is the corresponding decryption key of encryption key NK_e, which is applied to decrypt $\{DK\}_{NK_e}$.

B. Analysis of security goals of data

We also idealize the read operation procedure as below.

M4: $R_i \triangleleft \{Data\}_{\$DK}$
M5: $R_i \triangleleft \{\$DK\}_{@NK}$
M6: $R_i \triangleleft \{NK@\}_{KR_i}$

Three security goals described in BAN formulas need to be proved: $R_i \triangleleft Data$, $R_i \models W_j \models Data$ and $R_i \models Data$.

Applying the seeing rule **SEE2** to message **M6** and assumption **A1** yields

S20: $R_i \triangleleft NK@$.

Using Modus Ponens (MP) rule to **S20** with assumption **A9** produces

S21: $R_i \models \overset{NK@}{\longmapsto} R_i$.

Utilizing the seeing rule **SEE3** to message **M5** and **S21** gains

S22: $R_i \triangleleft \$DK$.

Employing the Modus Ponens (MP) rule to **S22** and assumption **A15** gets

S23: $R_i \models \overset{\$DK}{\longmapsto} R_i$.

Applying the seeing rule **SEE4** to message **M4** and **S23** yields

S24: $R_i \triangleleft Data$.

Now the first goal has been proved. To prove the remaining goals, we also change message **M4** slightly. The modification reason is similar with the one in Section IV-D. Supposing that the *Data* is produced by writer W_j, it first is encrypted by W_j's private key KW_j^{-1}.

M4': $R_i \triangleleft \{\{Data\}_{KW_j^{-1}}\}_{\$DK}$

Adopting the seeing rule **SEE4** to message **M4'** and **S23** produces

S25: $R_i \triangleleft \{Data\}_{KW_j^{-1}}$.

Applying the seeing rule **SEE2** to **S25** and assumption **A1** yields

$$\textbf{S24: } R_i \triangleleft Data.$$

Utilizing the Modus Ponens (MP) rule to **S24** and assumption **A16** gains

$$\textbf{S26: } R_i \models \sharp(Data).$$

Using the message-meaning rule **MM2** to **S25** with assumption **A1** produces

$$\textbf{S27: } R_i \models W_j \mid\sim Data.$$

Employing the nonce-verification rule **NV** to **S26** and **S27** gets

$$\textbf{S28: } R_i \models W_j \models Data.$$

Applying the jurisdiction rule **J** to **S28** and assumption **A6** yields

$$\textbf{S29: } R_i \models Data.$$

So far, we have proved three goals of data *Data*. So in the implement of the access control solution in Fig.1 (step b.2), the data domain of *Data* packet should be changed from $\{Data\}DK$ to $\{\{Data\}K_{pri-writer}\}DK$.

VI. Analysis of write and read operation in an open environment

In this section, we show the specific read and write operation in an open envrionment. The BAN-like logic is applied to them. Then we analyze them with some important security goals.

A. Write operation in an open environment

Similar with Section IV-A, write operation procedure in an open environment in 1.(c) can be idealized as blew.

$$\begin{aligned}
&\textbf{M1':} \quad W \triangleleft \{@NK, NK@\}_{KW_w}\\
&\textbf{M2:} \quad M \triangleleft \{Data\}_{\$DK}\\
&\textbf{M3:} \quad M \triangleleft \{\$DK\}_{@NK}
\end{aligned}$$

As *ACM* can figure out key K_w when it received the hash value of it, we can infer that

$$\textbf{A'3: } M \models\!\!\xrightarrow{KW_w} W.$$

There are six goals needing to be proved: $W \triangleleft$ (*@NK,NK@*), $W \models M \models$ (*@NK,NK@*), $W \models$ (*@NK,NK@*), $M \triangleleft Data$, $M \models W \models Data$ and $M \models Data$. With the help of assumption **A'3**, we can also prove them in almost the same way as we do in Section IV.

B. Read operation in an open environment

Like Section V-A, read operation procedure in an open environment in 1.(d) can be idealized as blew.

$$\begin{aligned}
&\textbf{M4':} \quad R \triangleleft \{Data\}_{\$DK}\\
&\textbf{M5':} \quad R \triangleleft \{\$DK\}_{@NK}\\
&\textbf{M6':} \quad R \triangleleft \{NK@\}_{K_r}
\end{aligned}$$

We can also deduce a new assumption.

$$\textbf{A'4: } M \models\!\!\xrightarrow{K_r} R$$

Three goals should be verified: $R \triangleleft Data$, $R \models W_j \models Data$ and $R \models Data$. With assumption **A'4**, we can demonstrate these goals using the similar way in Section V.

VII. Conclusion and Future Work

In this paper, we described a new approach to reasoning about an access control solution of NDN. Our work, which is inspired by the BAN logic, is a special logic to analyze different operation procedures in the access control solution. We idealized the procedures using the BAN-like logic and set several security goals to analyze them. These unsatisfied goals indicate that this access control solution cannot ensure the source of critical keys and data. This may lead the access control solution to be vulnerable to intruders. Then we did some improvement for the unsatisfied goals and made the new access control solution archive the important security goals. This method leads us in identifying mistakes and suggesting corrections. Our study helps to improve security and protect against various attacks for the access control solution. As for the future work, we would like to apply this method to other access control solutions of NDN.

Acknowledgement. This work was partly supported by Shanghai Collaborative Innovation Center of Trustworthy Software for Internet of Things (No. ZF1213).

References

[1] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. D. Thornton, D. K. Smetters, B. Zhang, G. Tsudik, kc claffy, D. Krioukov, D. Massey, C. Papadopoulos, T. Abdelzaher, L. Wang, and E. Yeh, "Named data networking (NDN) project," PARC, Tech. Rep. NDN-0001, 2010.

[2] T. Chen, K. Lei, and K. Xu, "An encryption and probability based access control model for named data networking," in *IEEE 33rd International Performance Computing and Communications Conference, IPCCC 2014, Austin, TX, USA, December 5-7, 2014*, 2014, pp. 1–8.

[3] J. Golle and D.Smetters, "Ccnx access control specifications," Xerox Palo Alto Research Center-PARC, Tech. Rep., 2010.

[4] B. Hamdane, R. Boussada, M. E. Elhdhili, and S. G. E. Fatmi, "Towards a secure access to content in named data networking," in *26th IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE 2017, Poznan, Poland, June 21-23, 2017*, 2017, pp. 250–255.

[5] M. Burrows, M. Abadi, and R. M. Needham, "A logic of authentication," *ACM Trans. Comput. Syst.*, vol. 8, no. 1, pp. 18–36, 1990.

[6] K. Gaarder and E. Snekkenes, "On the formal analysis of PKCS authentication protocols," in *Advances in Cryptology - AUSCRYPT '90, International Conference on Cryptology, Sydney, Australia, January 8-11, 1990, Proceedings*, 1990, pp. 106–121.

[7] L. Gong, R. M. Needham, and R. Yahalom, "Reasoning about belief in cryptographic protocols," in *Proceedings of the 1990 IEEE Symposium on Security and Privacy, Oakland, California, USA, May 7-9, 1990*, 1990, pp. 234–248.

[8] A. D. Rubin and P. Honeyman, "Nonmonotonic cryptographic protocols," in *Seventh IEEE Computer Security Foundations Workshop - CSFW'94, Franconia, New Hampshire, USA, June 14-16, 1994, Proceedings*, 1994, pp. 100–116.