A Credential and Encryption Based Access Control Solution for Named Data Networking

Balkis Hamdane[1,2] and Sihem Guemara El Fatmi[1]

[1]Digital Security Research Unit, Higher School of Communication of Tunis (Sup'Com), Tunisia
[2]Télécom ParisTech, Paris, France
balkis.hamdane@telecom-paristech.fr
sihem.guemara@supcom.rnu.tn

***Abstract*—Named Data Networking (NDN) represents a promising candidate for the future Internet architecture adopting the Information Centric Networking (ICN) approach. For a more effective content delivery, it leverages in-network caching. However, security can no longer be tied a particular location. It becomes a property of the content and its name, regardless where it is situated. To ensure access control that represents an important security feature, NDN proposes the use of an encryption-based model; sensitive data can be encrypted then decrypted only by legitimate entities. Many solutions adopting this model have been proposed but they require prior knowledge of all authorized entities. In this paper, we propose an encryption-based access control solution that does not have such requirements and which is valid in an open environment. This solution assigns access rights based on certified encrypted credentials provided by the different entities. To confirm the security of this proposal, a formal security analysis is provided.**

I. Introduction

Named Data Networking (NDN) [1] represents one of the most promising Information Centric Networking (ICN) projects. It considers the named content as the central element rather than IP addresses and it relies on in-network caching. A requested content can be recovered from any node possessing a copy of such content and this based on its name. Security in NDN becomes a property of the content and its name regardless where it is situated. It can no longer be ensured by traditional mechanisms linked to the content location.

To ensure the access control which is a fundamental security feature, NDN proposes the use of an encryption-based access control model; sensitive data can be encrypted then decrypted only by legitimate entities. A solution adopting this model is already implemented in its current prototype, CCNx [2]. This solution is based on Access Control Lists (ACL) and symmetric encryption to restrict the management, the writing and the reading of the sensitive content.

We analyzed in [3] this solution and we identified some weaknesses as the ability of an entity with only a read access to write a content under a protected namespace and the inappropriate use of ACL with a large number of entities. To mitigate the identified problems, we propose an enhancement that is essentially based on a new cryptographic model [3]. This proposal associates a different key for each access right. This key is stored encrypted using the public keys of the entities benefiting from the corresponding right. Despite the fact that this enhancement eliminates the use of ACL and prevents an entity to perform an unauthorized action, it is only valid in a particular context where all authorized entities are known in advance. It does not fit with an open environment characterized by a dynamic and constantly evolving population.

To satisfy the requirements of such an environment, we propose in this paper a credential-based access control solution. This proposal is based on content-encryption and it also associates a different key for each access right. However, this key is no longer stored encrypted in advance but an entity must provide trusted credentials satisfying a certain policy to gain access to this key.

The organization of the rest of this paper is as follows. Section II provides an overview of the NDN project. Section III analyzes the related work. Section IV describes the new access control solution. Section V provides a formal validation of this solution. Finally, section VI concludes the paper.

II. Named Data Networking

The communication model in the NDN project uses two packet types: Interest and Data. The Interest packet is broadcasted to request a content. It carries a hierarchical and human-readable name identifying the required content and a nonce value. Once the packet reaches any node maintaining a content with the same name, a Data packet is sent as a response to the content requester by following the reverse path. This packet is composed of the same name as that received in the Interest packet, the desired content, a signature linking the content to its name and information about the signature.

To ensure the content availability to consumers, NDN sets up repositories. Each repository is responsible for the permanent storage of the content published under a specific namespace by respecting an enforced policy [2].

For a more effective content delivery, NDN supports on-path caching. However, traditional security mechanisms linked to the content location can no longer be used. Therefore, NDN adopts a content-centric security model that attaches the security functions on the data and their names[4][5][6]. For example to ensure data validity and producer authentication, each chunk of data is concatenated to its name and signed using the producer private key. For access control, NDN proposes the use of an encryption-based model. The main solutions adopting this model are described in the next section.

III. Related work

Misra et al. propose in [7] an encryption-based access control solution that aims to securely distribute content in ICN. This solution is based on Broadcast Encryption and requires a registration phase before any content encryption. In this phase, the user should create a verified profile. He sends it encrypted to the Content Provider (CP) using the public key of the latter in a signed Interest packet. On successful access to the system, the CP sends to the client a Data packet containing the parameters necessary to decrypt the content. This solution controls only the read access.

Another encryption-based access control solution [8] that grants the rights of reading, writing and management is proposed and already implemented in CCNx. In this solution, protected content are encrypted using symmetric keys, named Data Keys (DK). The access rights management is organized around the concept of namespaces which is characterized by a tree-like structure [8]. To control the access to a specific namespace a marker, an ACL and a symmetric Node Key (NK) are created and associated with the root of this namespace. The marker is used to indicate the presence of an access control policy for this namespace. The ACL specifies the access rights of the authorized entities. The NK key is used to encrypt the DK Keys. It is stored encrypted with the public keys of all entities in the associated ACL. The inheritance of the rights in the name-tree is based on a modified hierarchical access control model. In this model, if no ACL is associated with it, a child node inherits that of the root node as well as the associated access rights. The key of this child node NK_{child} is derived from the root Node Key. To modify its access rights , a child node is assigned a new ACL and a new key NK.

Despite the fact that the described access control solution is simple, it preaches some weaknesses. Indeed, an entity can generate a random DK Key and it can use it to encrypt the content. With only a read right, it can recover the NK Key encrypted with its own public key. It decrypts it and uses it to encrypt the DK key. It finally succeeds in publishing the DK key and an encrypted content in a protected namespace. A second problem consists in the use of the ACL which is not recommended with a large number of entities.

To mitigate the identified problems, we propose in [3] an enhancement which is essentially based on a new cryptographic model for the access rights management. The proposed solution keeps the same principles as the solution implemented in CCNx. However, it replaces the use of the symmetric Node Key by a pair of keys ($NK_{encryption}$, $NK_{decryption}$). $NK_{decryption}$ assigns the read right. It is used to decrypt the Data Key and it is stored encrypted with the public keys of entities with a read privilege. $NK_{encryption}$ assigns the write right. It is used to encrypt the Data Key and it is stored encrypted with the public keys of entities with a write privilege. The private key associated to the root of a particular namespace assigns the management right. While ($NK_{encryption}$, $NK_{decryption}$) is like a key pair of a public key system, both of them are secret.
The proposed enhancement prevents an entity with only a read right from writing a protected contents since the possession of the $NK_{encryption}$ key is required as proof for its writing right. It also eliminates the use of the ACL because every access right is associated with a particular key.

IV. Proposed solution

The access control solution proposed in [3] represents an significant enhancement of the solution already implemented in CCNx and in contrast to the proposal of Misra et al. [7], it grants the reading, writing and management rights. However, it is only valid in a particular context where the manager of a namespace knows in advance all authorized entities. It is not adequate in an open environment characterized by a dynamic and constantly evolving population and when the knowledge beforehand of all entities is impossible. To satisfy the requirements of such an environment, we propose in this paper an access control solution that assigns access rights based on certified credentials provided by different entities.

A. Entities in the proposed solution and assumptions

Our access control solution is mainly composed by the following entities: the Access Control Manager (ACM), Network Nodes (NN) and users. The ACM defines and enforces access control policies for a specific namespace. It detains the private key of the root of this namespace and it has consequently the associated management right. The NN are responsible for the storage and the delivery of content. They include permanent storage repositories and all nodes having a cache. Users can be grouped into three categories: (1) Users without access rights, (2) readers and (3) writers.

In the proposed access control solution, it is assumed that malicious users try to write data in protected namespaces and to read sensitive content and that NDN ensures data validity, producer authentication and relevance. These services are guaranteed by the adoption of a suitable naming system combined to cryptographic mechanisms using trusted keys[9][6].

B. Overview of the proposed solution

To understand the general operating principle of the proposed solution, we present in figure 1 the interactions between different entities. First, the Access Control Manager creates an access control policy, and enforces it to the specific name prefix "'/telecom-paristech.fr"' in a repository (step 1). This policy sets the credentials that users have to provide to benefit from a specific access right. A credential represents a digitally signed statement made by a trust party to prove a qualification, it asserts a link between an entity and a property. Later, a user (Alice) wants to publish new content with the prefix "'/telecom-paristech.fr"'. She sends an Interest packet containing the marker "%C1.M.ACCESS", indicating whether this namespace is controlled. If so, she receives a Data packet specifying the used access control policy (step 3). If she can meet the requirements of this policy, Alice sends to the ACM a request of the Node Key (step 4). This Interest packet also contains the requested certified credentials in the nonce field and it is signed using the private key of the user. The ACM evaluates the received credentials. If Alice benefits from the read and write permissions, the $NK_{encryption}$ and $NK_{decryption}$ keys are encrypted using her public key $K_{pub-Alice}$ and they are sent to her (step 5). By receiving this packet, Alice generates a symmetric Data Key and uses it to encrypt the content. She prepares two Data packets. The first one contains the DK key encrypted using $NK_{encryption}$ as well as the policy used in this namespace. The second one

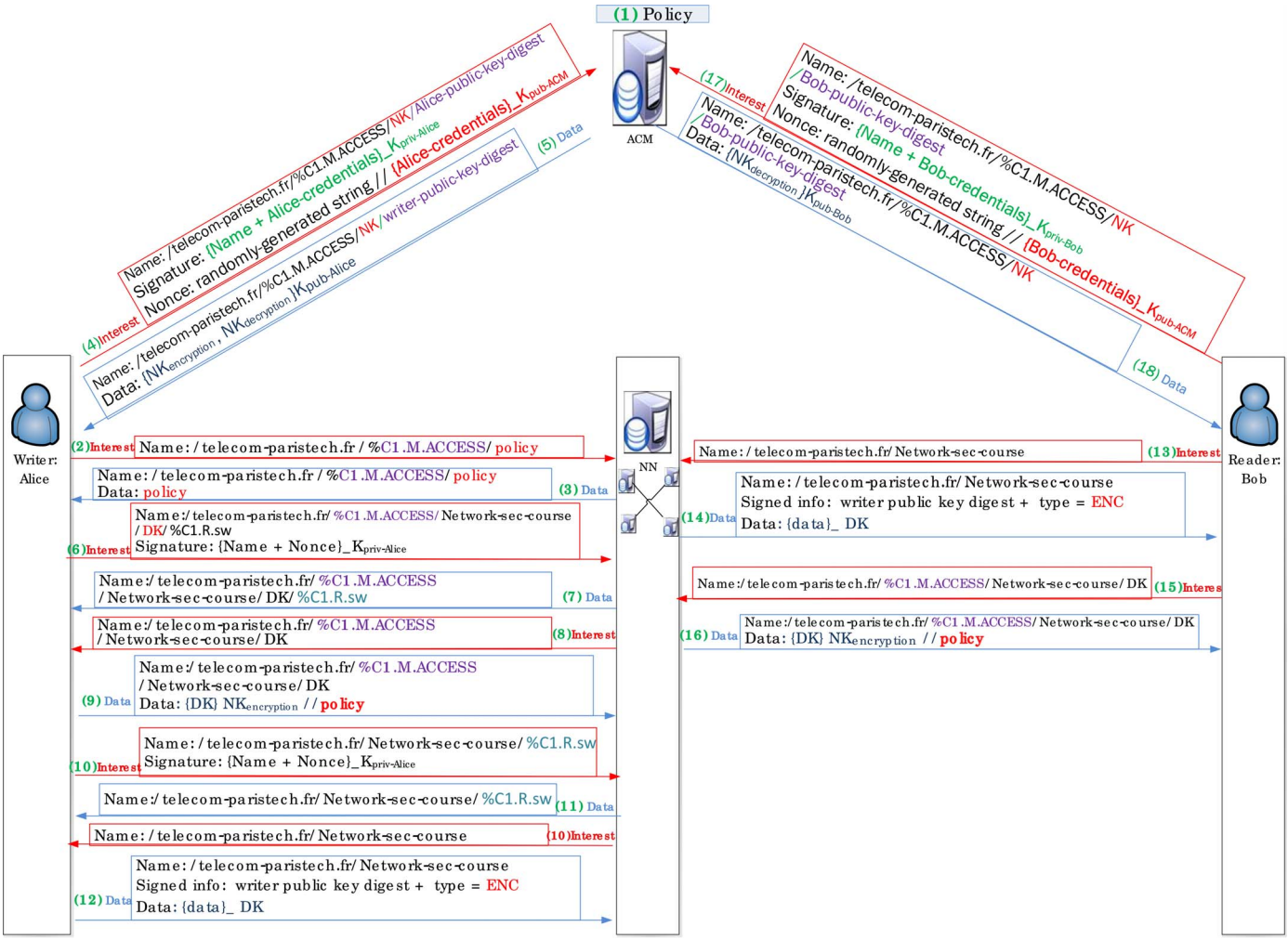

Fig. 1. General operating principle of the proposed access control solution

contains the content encrypted using DK. To indicate that the content is encrypted in this last packet, the field type is equal to "ENC". The publication of each of these packets requires firstly a write request sent to the repository responsible for the prefix "/telecom-paristech.fr". This request is expressed by the command "%C1.R.sw". In case of acceptance, this repository sends an Interest packet requesting the concerned Data packet. Alice sends as a response, the associated Data packet that will be stored in the repository. These exchanges are represented in the figure 1 by the steps going from 7 to 12.
Another user (Bob) requests the content published by Alice (step 13). He receives as response the Data packet containing the encrypted content and discovers that this content is protected since the field type is equal to "ENC" (step 14). Bob requests the Data Key (step 15). He recovers it encrypted using $NK_{encryption}$ concatenated to the policy used in this namespace (step 16). If he can meet the requirements of this policy, he sends an Interest packet requesting the $NK_{decryption}$ key and containing the required credentials (encrypted) in the field nonce (step 17). If Bob has the read right, the ACM encrypts the $NK_{decryption}$ key using his public key $K_{pub-Bob}$ and sends it to him (step 18). The user can now decrypt the DK key as well as the content.

For the sake of simplicity, we presented in the figure 1 a grant of access based on the credentials of individual users (Alice and Bob). However, our solution supports also the notion of groups. In this case to have an access right, a user belonging to a particular group sends an Interest packet containing the credentials of his group and signed by the private key of this group. If these credentials are previously verified and validated, the $NK_{encryption}$ and / or $NK_{decryption}$ keys are already encrypted using the public key of the group and they can be recovered from the nearest Network Node having a copy of this key.

C. Discussion

1) Supporting the notion of groups: The notion of groups is supported in our solution for two reasons. The first one is to reduce the workload of the ACM. This entity will not have to check the credentials of each user and to encrypt the NK Key with the corresponding public key, but this work will be done only once for a group of users. The second reason is to take advantage of the property of caching since the NK Key can be recovered from the nearest NN storing this key.

2) Signature of the Interest packet and credentials encryption: To have an access right, a user sends a signed Interest packet to the ACM. This packet contains a random value and the user credentials encrypted using the ACM public key in the field nonce. It also includes the signature of both the name

```
role manager (...
1. State=1/\ RCV(RequestNK.{Writer_credentials'}_Kmanager.{RequestNK.
{Writer_credentials'}_Kmanager}_inv(Kwriter).Kwriter.{Kwriter.W}_inv(Kca)
... =|> State'=10 /\ Write_Right_writer'= true/\ SND(RequestNK.
{NK}_Kwriter.{RequestNK.{NK}_Kwriter}_inv(Kmanager). Kmanager.
{Kmanager.M}_inv(Kca).Write_Right_writer')
/\ witness(M,W, authen_cred_writer,Writer_credentials')
        10. State=10/\ RCV(RequestNK.{Reader_credentials'}_Kmanager
.{RequestNK.{Reader_credentials'}_Kmanager}_inv(Kreader). Kreader.
{Kreader.R}_inv(Kca) ) /\ Reader_credentials' = Reader_credentials
=|> State'=11/\ SND(RequestNK.{NKdecryption'}_Kreader.{RequestNK.
{NKdecryption'}_Kreader}_inv(Kmanager). Kmanager.{Kmanager.M}_inv(Kca))
/\ witness(M,R, authen_cred_reader,Reader_credentials') ...
role writer(...
0. State=0 /\ RCV(start) =|> State'=2/\ SND(RequestNK.{Writer_credentials}
_Kmanager. {RequestNK.{Writer_credentials}_Kmanager}_
inv(Kwriter). Kwriter. {Kwriter.W}_inv(Kca) )
/\ witness(W, M, authen_cred_writer,Writer_credentials)
/\ secret (Writer_credentials,sec_cred_writer, {W,M})
      2. State=2 /\ RCV(RequestNK.{NK'}_Kwriter.{RequestNK.{NK'}_Kwriter}
_inv(Kmanager).Kmanager.{Kmanager.M}_inv(Kca).Write_Right_writer')
/\ Write_Right_writer'= true
=|> State'=4 /\.../\ SND(Name.Write_Right_writer')
/\ witness(W, R, id,NK'.Write_Right_writer')
      4. State=4 /\ RCV(Name)
=|> State'=8 /\ SND(Name.H(Kwriter).HDK. EncData.{Name.H(Kwriter).HDK.
 EncData}_inv(Kwriter).Kwriter.{Kwriter.W}_inv(Kca))
      8. State=8/\ RCV(HDK) =|> State'=13 /\ SND(H(inv(NK)).{DK}_NK)
    /\ secret (Data,sec_data, {W,M,R}) ....
role reader (...
3. State=3 /\ ...
      5. State=5 /\ RCV(Name.H(Kwriter').HDK'.EncData'.{Name.H(Kwriter').
HDK'. EncData'}_inv(Kwriter').Kwriter'.{Kwriter'.W}_inv(Kca))
=|> State'=11 /\ SND(RequestNK.{Reader_credentials}_Kmanager.{RequestNK.
{Reader_credentials}_Kmanager}_inv(Kreader).Kreader.{Kreader.R}_inv(Kca))
/\ witness(R, M,authen_cred_reader,Reader_credentials)
/\ secret (Reader_credentials,sec_cred_reader, {R,M})
      11. State=11/\... =|> State'=9 /\ SND(HDK)
      9. State=9 /\RCV(H(inv(NK')).{DK'}_NK') /\H(inv(NK'))=NKdecryption
=|> State'=12 /\ wrequest(R,W, id, NK.Write_Right_writer) ...
goal
weak_authentication_on id, authen_cred_writer, authen_cred_reader
secrecy_of sec_cred_reader, sec_cred_writer, sec_data ...
```

Fig. 2. Extract of HLPSL specifications

and the credentials calculated using the user private key. By receiving this packet, only the ACM can decrypt credentials. It can then verify the signature and the rights associated with the user. In case of validation, it sends $NK_{encryption}$ and / or $NK_{decryption}$ encrypted using the user public key.

The credentials encryption ensures confidentiality and protects user privacy. The signature included in the Interest packet prevents a malicious user to request the Node Key encrypted with its own public key using the credentials of a legitimate user. Indeed, given that it cannot decrypt the credentials, he cannot forge an Interest packet containing a signature linking the requested key's name to these credentials.

V. Formal validation

To verify the robustness of the proposed access control solution, a formal security analysis is done using the automatic formal security analyzer AVISPA [10]. This tool uses the HLPSL language which is based on two categories of roles: (1) the simple roles describing the actions of agents during the execution of a protocol, (2) the composed roles instantiating several simple roles to model the execution of the whole protocol. To check the safety of a specified protocol, the required security properties are specified as security goals in HLPSL. AVISPA integrates different back-ends that analyze the possible behaviors of the protocol.

To specify the proposed access control solution using the HLPSL language, three simple roles are specified: the manager, the writer and the reader. Extracts from the writer specification is presented in figure 2. This specification aims at verifying that: (1) the ACM, represented here by the role manager, authenticates the reader and the writer by their credentials, (2) those credentials are confidential, (3) the protected content can be read only by the manager, the reader or the writer. The expressions starting with "wrequest", "witness" and "secret" in the transitions of the simple roles as well as the expressions beginning with the words "weak_ authentication_ on" and "secrecy_ of" on the goal section, in red in figure 2, allow the checking of these security goals. The AVISPA execution proves the safety of our proposal and doesn't detect any vulnerability.

VI. Conclusion

With caching property in NDN, a named data can be situated in multiple locations. An access control solution built-in the content itself regardless where it is situated must be adopted. Many encryption-based solutions have been proposed but all of them are only valid in a particular context where all authorized entities must be known in advance. We propose in this paper a solution valid when the knowledge beforehand of all entities is not necessary. This solution is still based on content encryption and it is organized around the concept of namespaces. However, it assigns access rights based on certified credentials provided by different entities. These credentials are encrypted which ensures confidentiality and protects users privacy. They are sent in signed Interest packets thus preventing credentials from spoofing attacks. To verify the security of the proposed solution, a formal security analysis was done using AVISPA tool. Our future work focuses on the integration of this proposal in CCNx to evaluate its performances

References

[1] Z. Lixia, A. Alexander, B. Jeffrey, J. Van, c. kc, C. Patrick, P. Christos, W. Lan, and Z. Beichuan, "Named data networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 32, pp. 66–73, 2014.

[2] P. Mahadevan, "Ccnx 1.0 tutorial," *Technical Report, Xerox Palo Alto Research Center-PARC*, 2014.

[3] B. Hamdane, A. Serhrouchni, and S. G. E. Fatmi, "Access control enforcement in named data networking," in *International Conferece For Internet Technology And Secured Transactions, 2013*. IEEE, 2013.

[4] D. Smetters and V. Jacobson, "Securing network content," *PARC Tech Report TR-2009-1, Xerox Palo Alto Research Center-PARC*, 2009.

[5] X. Zhang, K. Chang, H. Xiong, Y. Wen, G. Shi, and G. Wang, "Towards name-based trust and security for content-centric network," in *Network Protocols (ICNP)*. IEEE, 2011, pp. 1–6.

[6] B. Hamdane, A. Serhrouchni, A. Fadlallah, and S. G. El Fatmi, "Named-data security scheme for named data networking," in *International Conference on the Network of the Future (NoF)*. IFIP - IEEE, 2012.

[7] S. Misra, R. Tourani, and N. E. Majd, "Secure content delivery in information-centric networks: design, implementation, and analyses," in *Proceedings of the 3rd ACM SIGCOMM workshop on Information-centric networking*. ACM, 2013, pp. 73–78.

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

[9] A. Ghodsi, T. Koponen, J. Rajahalme, P. Sarolahti, and S. Shenker, "Naming in content-oriented architectures," in *Proceedings of the ACM SIGCOMM workshop on Information-centric networking*. ACM, 2011.

[10] A. Team *et al.*, *AVISPA v1.1 User manual*, June 2006.