Information-Centric Networking Research Group
Internet-Draft
Intended status: Informational
Expires: January 9, 2020

R. Li
H. Asaeda
NICT

July 8, 2019

Hop-by-Hop Authentication in Content-Centric Networking/Named Data Networking
draft-li-icnrg-hopauth-00

Abstract

The unpredictability of consumers, routers, copyholders, and publishers for the in-network data retrievals in Content-Centric Networking (CCN) / Named Data Networking (NDN) poses a challenge to design an authentication mechanism to inhibit the malicious consumers to flood data requests and prevent the fake data from being provided. Signature is adopted as the fundamental function in CCN / NDN, which however can only provide publisher authentication with additional certificate acquisition. This document describes the the Hop-by-Hop Authentication mechanism (HopAuth) integrating certificate collection and packet forwarding potentially with the assistance from certificate authority to provide consumer authentication, copyholder authentication and path authentication to enable the in-network data retrieval to be trustworthy, besides the publisher authentication.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 9, 2020.

Table of Contents

1. Introduction

Information-Centric Networks in general, and Content-Centric Networking (CCN) [3] or Named Data Networking (NDN) [4] in particular, are the emerging network architectures enabling in-network caching and data retrievals through their names. In CCN/NDN, data can be cached at the intermediate routers, close to consumers for reducing delay and redundant bandwidth consumption or for the robustness under dynamic network environment. It has been noticed that CCN/NDN is a promising approach for the application scenarios in disaster networking [5], video streaming [6], and Internet of Things (IoT) [7] .

In CCN/NDN, the basic network operations and these use scenarios with

in-network data caching and retrievals lead the network to be seriously vulnerable under a variety of attacks, such as the impersonation attack, malicious-request attack [8], [9], [10], and the data poisoning attack [11], [12], [13]. The unpredictability of consumers, routers, copy holders, and publishers during data

retrievals in CCN/NDN poses the novel challenge to design data-centric authentication to prevent these attacks. This novel authentication should potentially enable the consumer authentication, copyholder authentication to authenticate the identity of the entity providing data, path authentication to authenticate the path from which data are retrieved, in addition to the publisher authentication.

On the other hand, signature is already adopted as the fundamental function in CCN/NDN, which promises to achieve the integrity and publisher authentication. It can partially prevent the above attacks and but still is insufficient to protect the unpredictable data retrievals in CCN/NDN. The unpredictability with which copy holders provide data, routers cache data, and consumers request data leads to great difficulty in inhibiting malicious-request attacks and data-poisoning attacks. To prevent data poisoning, consumers and routers need to verify data before caching them. If the data are found to be fake, the copy holder providing the data and the path to retrieve the data also need to be discovered in order to disable the further spread of that fake data. To prevent malicious-request attacks, copy holders need to verify the identities of the consumers.

There are many the existing authentication mechanisms with key management schemes in Internet, such as Kerberos [14], MSEC [15], X.509 [16], PGP [17], RPKI [18]. They are designed to achieve different purposes with centralized or decentralized approach based on end-to-end communication paradigm within the Internet. They can only provide the authentications between the consumers and publishers without considering data-centric authentication, and are unable to prevent the malicious-request and data-poisoning attacks. Furthermore, they rely on centralized servers to acquire keys or certificates, thereby increasing authentication delays, which we refer to herein as the delay-enlargement problem. Obviously, they cannot satisfy the requirements for the emerging data-centric communication paradigm in CCN/NDN, because of different security and performance concerns.

In this document, we elaborate HopAuth [19], where forwarding-integrated hop-by-hop certificate collection is performed together with the adaptive replacement for parts of chain with highly trustworthy certificates. In HopAuth, a suspension-chain model (SCM) is used as the trust model, where the neighbor-trust-based certificate chain is suspended by certificate authority (CA)-based trust. The HopAuth avoids reliance on centralized server(s) for chain construction and solves the delay-enlargement problem.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [1].

The following terminology is used throughout this document.

- Cryptographic key: A string of bits used by a cryptographic algorithm to transform plain-text into cipher-text or vice versa.
- Signature: A cryptographic value calculated through public key algorithm from the data and a secret key only known by the signer. It is to validate the authenticity and integrity of a message.
- Certificate: A data structure used to verifiably bind an identity to a cryptographic key.
- Consumer: A node requesting data. It initiates communication by sending an interest packets.
- Publisher: A node providing data. It originally creates or owns the data.
- Router: A node forwarding data. It may hold memory to cache the data.
- Forwarding Information Base (FIB): A lookup table in a router containing the name prefix and corresponding destination interface

to forward the interest packets.

- Pending Interest Table (PIT): A lookup table populated by the interest packets containing the name prefix of the requested data, and the outgoing interface used to forward the received data packets.

- Content Store (CS): A storage space for a router to cache data objects. It is also known as in-network cache.

- Physical entity: an entity that communicates using a physical device. This could be a router or a publisher node (PN) that hosts applications.

- Logical entity: an entity that is involved in an application. This can be an authorizer, a sub-authorizer, or a publisher.

3. System Descriptions

Here we briefly explain the background and basic network operations of CCN/NDN. Different from the end-to-end communications in Internet, CCN/NDN provides data-name-based retrievals as in Fig. 1. It further requires the data-centric authentication, instead of the current end-to-end secure channel establishment.

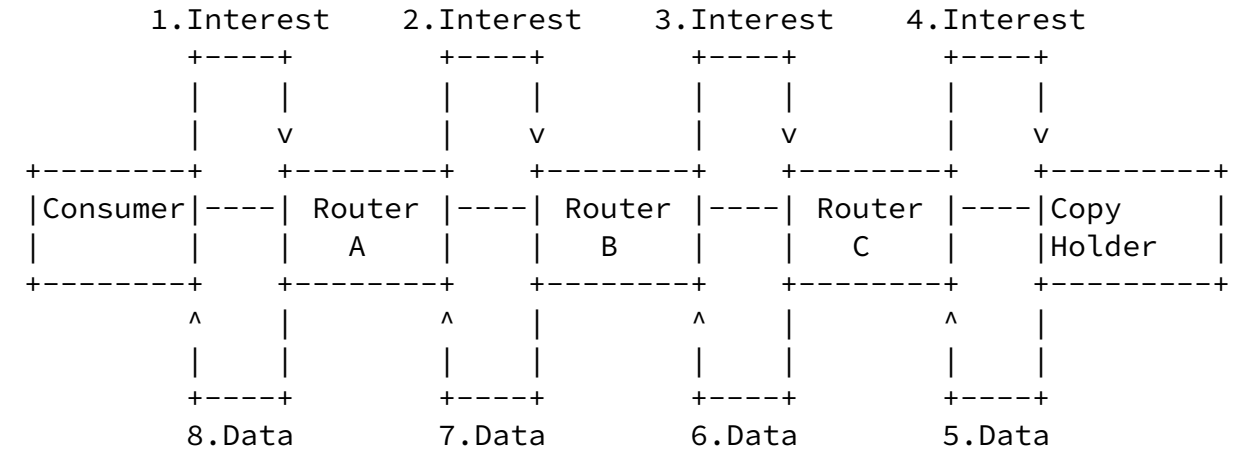


Figure 1: Request and reply messages forwarded by consumer, copy holder and routers.

In a CCN/NDN network, each router in a CCN/NDN network has three main data structures: a FIB for forwarding Interests, a CS for caching data, and a PIT for forwarding data. Basically there are two types of packets: interest and data. As in Fig. 1, consumer requests data by throwing an "interest" packet with the name of data to the network. Regarding the difference to note here between CCN [3] and NDN [4] is that in later versions of CCN, interest packet must carry a full data name, while in NDN it may carry a data name prefix.

Once a router receives an "interest" packet, it performs a series of the following look-up.

The router first checks in the CS to see whether it holds the corresponding data or not. If there is, it returns the data through the reverse path for forwarding interest packet as the copy holder in Fig. 1. If not, it performs a look-up of the PIT. If there is already a PIT entry matching the name of requested data, it is updated with the incoming interface of this new request and the interest is discarded. If there is no matching entry, it creates an entry in the PIT that lists the data name and the interfaces from which it received the interest. Then, the interest undergoes a FIB lookup to discover the outgoing interface.

Once a copy of the "data" packet is retrieved, the router sends it back to the data requester(s) using the trail of PIT entries and remove the PIT state every time that an interest is satisfied. Additionally, it may store the data in its CS.

However, data retrieval with in-network caching in CCN/NDN has been identified to suffer from malicious data-request attacks [8], [9], [10], and the data poisoning attacks [11], [12], [13]. In the former, adversaries impersonate consumers to create a flood of interests, and in the latter, they impersonate copy holders (e.g., routers or publishers) to provide fake data. These attacks are severe, because data are cached in a distributed manner, and copy holders have no way to verify consumers' identities, consumers/ routers have no way to verify copy holders' identities to avoid caching fake data, and the path to retrieve data cannot be verified.

This form of attack can quickly pollute the router caches as the virus spreads, because routers cache the fake data, redistribute them, and other intermediate routers re-cache them. It finally consumes much in-network caches and prevents consumers from retrieving the correct data.

4. HopAuth Designs

Herein we elaborate the design of HopAuth, which has three phases: 1) initial trust-establishment, 2) data-centric certificate management, and 3) forwarding-integrated authenticable data-retrieval. The trust model for HopAuth is a suspension chain model (SCM).

For the first phase, certificates are issued for neighboring entities and the certificate authority (CA) potentially issues certificates to a set of routers to form the highly trusted router group (HTRG). Let CE(A->B) represent the public-key certificate issued from A to B. In the second phase, routers exchange certificates within their neighborhoods, and any compromised entity can be shut down quickly. Finally, the third phase provides a hop-by-hop method for constructing a suspension chain consisting of a physical-entity certificate chain (peCEChain) and a logical-entity certificate chain (leCEChain) for data-centric authentication.

4.1. Initial Trust Establishment

The initial trust establishment has two components: self-certifiable naming and certificate issuing.

Self-certifiable naming defines the rule for naming the principals, including entities, keys, and certificates, to enable the entities to be self-certifiable. As the extension of the cryptographically generated address (CGA), we merge the hash-based self-certifying

names with hierarchical naming. It embeds the public key into the name, which is self-certifiable.

To issue a certificate, an entity should be convinced that a given public key truly belongs to another entity. Under that situation, it issues a certificate for this entity such that the public key is bound to the entity name by its signature. In HopAuth, certificate issuing is the initial trust establishment among entities having

trust relationships, namely as neighbor-based trust for physical entities, CA-based trust, and authorization relationships for logical entities.

For neighbor-based trust, a physical entity creates a public key and the corresponding private key locally by itself. It generates the names using the public key based on the self-certifiable naming method. If physical entity B is a neighbor of entity A, B can announce its public key to A with the key name and A generates the certificate for B.

CA-based trust between two entities is established using the CA as the ``introducer''. The CA is managed by the network operator, and provides certificates to the owner's entities and the highly trusted physical entities close to them in the HTRG. The entities in the HTRG then confer CA-based trust relationships and issue certificates to each other.

All the highly trustable entities register at CA. CA has the whole view of these entities and determines the neighboring relations among the entities in HTRG. After determination of the neighboring relations, an entity, A, sends interest to CA to request the certificates of its neighbors. CA replies with the related certificates to A, such as CE(CA->B). After A receiving the certificates, A verifies them, issues certificates from itself to those neighbors, such as CE(A->B), and sends to CA. CA verifies this certificate and forwards it to B when B requests.

In HopAuth, the physical entities pre-keep the certificates and associate certificates with interfaces in FIB. The physical entity knows the next hop of one interface, and it associates the certificate from that entity to itself with the forwarding interface. This mechanism enables the appending of the relevant certificate to the packets as required when forwarding them.

For logical entities, the trust relationship is defined by the application. Each logical entity also generates a public/private key pair by itself. If logical entity A authorizes the right for entity B to manage a sub-category or publish data, A should provide a certificate for the true public key of B.

4.2. Data-centric Certificate Management

4.2.1. Certificate Exchange

Certificate exchange allows entities to share the certificates that they issue and hold. Each physical entity has a local repository in which to store certificates securely. In HopAuth, the physical entities request and keep all the certificates issued for or by their nearby highly trusted entities in the HTRG and by common neighboring physical entities. To exchange the information of certificates they hold, each entity hashes the names of certificates and exchange hashes to see if there is one missed in the neighborhood. If there is, the entity will send interest to its neighbor to retrieve that certificate. Finally, the physical entities hold all the certificates within a two-hop distance and the certificates with nearby highly trusted entities in the HTRG. These certificates are used to construct chains hop by hop, shorten certificate chains, and check the certificates appended by an up-stream entity. This certificate check is performed by the next hop of the entity to check whether the certificate appended by this entity is the same as the one stored by it. If the certificates are the same, the certificate passes the check. Otherwise, this packet will be dropped because of the fake certificate.

The certificates of the logical entities in an application should be stored in the repository of the PN. When data are published, the trust chain from the PN to the publisher can be automatically formed and appended to the packet. Meanwhile, the neighbors of the PN also keep all the certificates of the PN in order to check them during transmissions.

4.2.2. Certificate Update and Revocation

To guarantee its validity, each certificate in the network is issued with a certificate expiration time, after which the certificate is invalid.

For certificate updates using neighbor-based trust, the subject entity of the certificate should notify the issuer of its interest in updating the certificate. On receiving this interest, the issuer checks whether this entity has been compromised. It then checks whether the mapping between the name and the public key satisfies the naming rule. If all the checks are passed, the issuer considers whether the public key of the subject entity is still trustworthy, generates an updated certificate, and replies with this updated certificate. If any check is failed, the issuer does not provide a certificate update to that entity. For certificate updates using CA-based trust in the HTRG, the subject entity of the certificate

requests the CA to issue an updated certificate and provide it to the related nearby highly trusted routers, who can further issue update certificates.

If one entity no longer wishes to trust another entity, the former may revoke the certificate that it originally issued. Because certificates are issued over a one-hop distance, it is easy to detect the misbehavior of an entity. To revoke a certificate quickly, the revocation is announced over a two-hop distance. The revocation initiator broadcasts the revocation information to all the entities within a two-hop distance. Each entity receiving this information adds the compromised entity or certificate to its blacklist. All the packets from the compromised entities are dropped for a rapid local shutdown.

4.3. Forwarding-Integrated Authenticable Data Retrieval

Here, when forwarding interests and data, we define a forwarding-integrated hop-by-hop approach to construct the suspended chain from unpredictable copy holders to a consumer for authenticating interest, and from a consumer or router to data for authenticating copy holders or publishers or path to retrieve data. We let highly trusted routers replace the parts of chain with highly trusted certificates induced by CA's suspended trust to enhance trustworthiness.

The steps for data retrieval are as follows.

Step 1: The consumer issues an interest appended with its signature. It knows its next hop is the router to which it is connected. It then appends to this interest the certificate from the next-hop router to itself. Finally, it sends out the interest.

Step 2: When the interest is received by an router, it checks whether the previous certificates are correct. If the check succeeds and it belongs to the HTRG, this router attempts to find the previous highly trusted router to replace the related part of the certificate chain with one highly trusted certificate in the suspended chain. If this IPE does not belong to the HTRG, it directly finds the interface to the next hop and appends to the interest the relevant certificate from the next-hop router to itself.

Step 3: This is executed if an router holds the requested data in its cache. The router checks the suspended chain from it to the consumer through the process described later in the SCM. If the verification succeeds, this router replies with the data packet. In the data packet, the suspended chain from this router to the publisher and the

certificate from the next router to this router are appended. This

router sends this data packet to the interface in reply as specified in the PIT.

Step 4: If the PN receives the interest, it verifies the suspended chain from itself to the consumer. If the verification succeeds, the PN discovers the suspended chain from itself to the publisher in its storage, and appends this certificate chain with the certificate from the first router to itself. Next, the PN replies with these data using the reverse path of the interest.

Step 5: After the router receives the data packet, it performs forwarding and possibly caching. When the router intends to cache the data, it first caches them in a temporary cache, which is separated from the data cache. Second, it checks the suspended chain from itself to the publisher. Only if the verification passes, these data can be cached in the data memory and the suspended chain from this router to the publisher will be cached along with the data. The verification is performed offline, which does not affect the speed of data retrieval. At the same time, this router checks the previous certificate. If the check is successful and it belongs to the HTRG, it will discover the previous entity in the suspended chain belonging to the HTRG. If there is a previous entity, the router replaces part of the related certificate path with a highly trusted certificate. Otherwise, the router directly finds the interface to the corresponding certificate from the next hop to itself and appends this certificate to the packet, then forwards this packet to the interface.

Step 6: After the consumer receives the data packet, there should be a certificate chain from it to the publisher. It verifies this suspended chain. If the verification passes, it believes that it gets the authentic public key of the publisher, and utilizes the key to verify the signature of the data. The consumer can also verify the copy holder or the routers on the path.

4.4. Suspension-Chain Model (SCM)

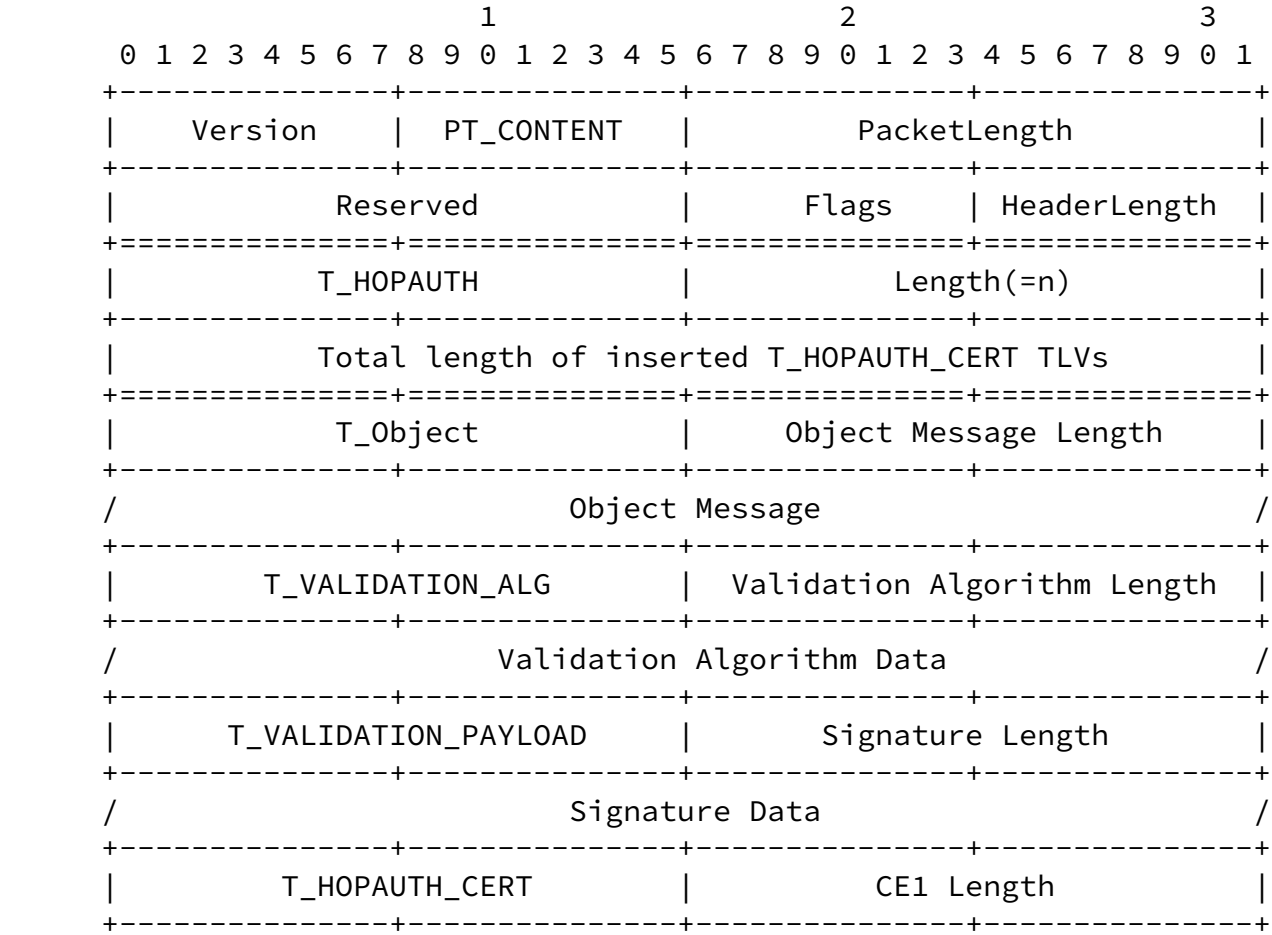
A suspension-chain model (SCM) is the trust model in HopAuth. It is a flexible series of neighbor-trust-based certificates suspended by

CA's trust, which form a suspension chain. In SCM, neighbor-based trust forms the certificate chain to realize data-centric authentication, whereas CA-based trust reduces the length of the chain to solve the trust degradation problem for certificate chain. For CA's trust, the CA assigns certificates to highly trusted routers as the suspension points based on CA's trust, which is the pre-trust between these routers and CA.

Public-key verification is the process for one entity to verify the authenticity of the public key of another entity. Entity X authenticates the public key of the entity Y by verifying the suspension chain in the following steps. First, X verifies the first certificate by its private key. If it is correct, each intermediate public key is used to verify the next direct associated certificate. This process continues for multiple rounds until the final certificate is verified. Finally, X obtains the true public key of the entity Y.

5. Protocol Message Format

HopAuth messages are encoded in the CCNx TLV format [2]. An example of HopAuth packet format with a certificate chain from consumer C0 to router R_n, (CE1(C0->R1), ..., CE_n(R_(n-1)->R_n)), is provided in Figure 2. As in Figure 2, the HopAuth message consists HopAuth header, and HopAuth certificate TLVs.


1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Version
PT_CONTENT
PacketLength
Reserved
Flags
HeaderLength
T_HOPAUTH
Length(=n)
Total length of inserted T_HOPAUTH_CERT TLVs
T_Object
Object Message Length
Object Message
T_VALIDATION_ALG
Validation Algorithm Length
Validation Algorithm Data
T_VALIDATION_PAYLOAD
Signature Length
Signature Data
T_HOPAUTH_CERT
CE1 Length

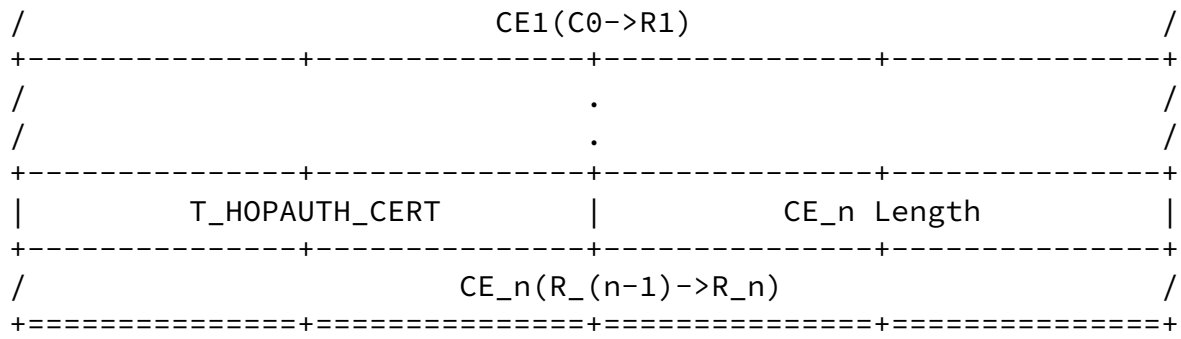


Figure 2: An Example of HopAuth packet format

The HopAuth header is composed of the fields of type, Length, and Total length of inserted T_HOPAUTH_CERT's TLVs. The type is T_HOPAUTH, the Length shows the number of inserted certificates, and Total length of inserted T_HOPAUTH_CERT's TLVs describes the total number of bits occupied by certificate TLVs. In the part of HopAuth certificate TLVs, the certificate name, length and certificate data are included. Take CE1(C0->R1) in Figure 2 as example. The type is T_HOPAUTH_CERT, CE1 Length is the length of certificate CE1(C0->R1), and CE1(C0->R1) is the certificate data for CE1(C0->R1).

6. Security Considerations

This document is entirely about authentication mechanism in CCN/NDN.

7. References

7.1. Normative References

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[2] Mosko, M., Solis, I., and C. Wood, "CCNx Messages in TLV Format", draft-irtf-icnrg-ccnxmessages-09 (work in progress), January 2019.

7.2. Informative References

[3] Jacobson, V., Smetters, D., Thornton, J., Plass, M., Briggs, N., and R. Braynard, "Networking Named Content", Proc. CoNEXT, ACM, December 2009.

[4] Zhang, L., Afanasyev, A., Burke, J., Jacobson, V., Claffy, K., Crowley, P., Papadopoulos, C., Wang, L., and B. Zhang, "Named data networking", ACM Comput. Commun. Rev., vol. 44, no. 3, July 2014.

[5] Seedorf, J., Arumaithurai, M., Tgami, A., Ramakrishnan, K., and N. Melazzi, "Research Directions for Using ICN in Disaster Scenarios", draft-irtf-icnrg-disaster-07 (work in progress), June 2019.

[6] Westphal, C., Lederer, S., Posch, D., Timmerer, C., Azgin, A., Liu, W., Mueller, C., Detti, A., Corujo, D., Wang, J., Montpetit, M., and N. Murray, "Adaptive Video Streaming over Information-Centric Networking (ICN)", RFC 7933, August 2016.

[7] Ravindran, R., Zhang, Y., Grieco, L., Lindgren, A., Burke, J., Ahlgren, B., and A. Azgin, "Design Considerations for Applying ICN to IoT", draft-irtf-icnrg-icniot-03 (work in progress), May 2019.

[8] Afanasyev, A., Mahadevan, P., Moiseenko, I., Uzun, E., and L. Zhang, "Interest flooding attack and countermeasures in named data networking", Proc. IFIP Networking, IFIP, May 2013.

[9] Compagno, A., Conti, M., Gasti, P., and G. Tsudik, "Poseidon: mitigating interest flooding ddos attacks in named data networking", Proc. LCN 2013, IEEE, October 2013.

[10] Nguyen, T., Cogranne, R., and G. Doyen, "An optimal statistical test for robust detection against interest flooding attacks in ccn", Proc. International Symposium on Integrated Network Management (INM), IFIP/IEEE, May 2015.

[11] Ghali, C., Tsudik, G., and E. Uzun, "Network-layer trust

in named-data networking", ACM SIGCOMM Computer Communication Review, vol.44, no. 5, October 2014.

[12] Kim, D., Nam, S., Bi, J., and I. Yeom, "Efficient content verification in named data networking", Proc. ACM Conference on Information-Centric Networking, ACM, September 2015.

[13] Gasti, P., Tsudik, G., Uzun, E., and L. Zhang, "Dos and ddos in named data networking", Proc. IEEE ICCCN 2013, IEEE, August 2013.

[14] Neuman, C., Yu, T., Hartman, S., and K. Raeburn, "The Kerberos Network Authentication Service (V5)", RFC 4120, July 2005.

[15] Baugher, M., Canetti, R., Dondeti, L., and F. Lindholm, "Multicast Security (MSEC) Group Key Management Architecture", RFC 4046, April 2005.

[16] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

[17] Callas, J., Donnerhacke, L., Finney, H., Shaw, D., and R. Thayer, "OpenPGP Message Format", RFC 4880, November 2007.

[18] Bush, R. and R. Austein, "The Resource Public Key Infrastructure (RPKI) to Router Protocol Version 1", RFC 8210, September 2017.

[19] Li, R., Asaeda, H., and J. Wu, "DCAuth: Data-Centric Authentication for Secure In-Network Big-Data Retrieval", IEEE Transactions on Network Science and Engineering, DOI: 10.1109/TNSE.2018.2872049, September 2018.

Authors' Addresses

Ruidong Li
National Institute of Information and Communications Technology

4-2-1 Nukui-Kitamachi
Koganei, Tokyo 184-8795
Japan

Email: lrd@nict.go.jp

Hitoshi Asaeda
National Institute of Information and Communications Technology
4-2-1 Nukui-Kitamachi
Koganei, Tokyo 184-8795
Japan

Email: asaeda@nict.go.jp