Efficient and Secure Device Clustering for Networked Home Domains

Daeyoub Kim, *Member, IEEE*, and Jihoon Lee, *Member, IEEE*

***Abstract*—Virtual private community (VPC) architecture is a hierarchical domain structure for intuitively and securely sharing content with others without privacy invasion. To build a secure domain like a smart home using VPC, a certificate chain-based association (CCA) scheme was proposed. CCA is a secure device clustering method, which consists of a member authentication based on a hierarchical trust model and a secure enrollment process. Since CCA can be operated without using any external authentication systems, consumers can build private domains using their own devices from different vendors. However, CCA has several weak points, such as the absence of the root authentication of a hierarchical trust model, the absence of revocation management, and a high processing burden caused by a chain model. Such weak points cause various attacks/problems like a spoofing attack and time synchronization. Also, they can create bottleneck in the case of VPC with large numbers of devices. So, to practically implement VPC, CCA should be improved so that it will be a reliable and efficient trust mechanism. Hence, this paper proposes a member list chain and member reputation-based association (MLC-R_bA) scheme. MLC-R_bA changes the type of authentication data, from a certificate to a list. Also, it proposes two necessary functions to authenticate the root of a hierarchical trust model without using an external trust authority and to control revoked members without causing user inconvenience and performance degradation. Additionally, to enhance a service performance, it proposes to use a domain hub device which is a representative entity publishing a valid membership status list. So it can solve time synchronization problem and a service bottleneck problem. Like CCA, MLC-R_bA is a local trust management scheme, which does not need any external specific authentication systems. Hence, using MLC-R_bA, consumers can make and manage their own domains using devices from different vendors, more securely and easily, but without performance degradation.**

***Index Terms*—Access control, authentication, cloud computing security, device clustering, CCN/ICN, smart home, social network services.**

I. Introduction

As MOBILE/SMART devices connected over the Internet become popular, consumers would like to seamlessly utilize and enjoy various services using such devices anytime and anywhere. For example, it has already become a strong trend that consumers share and manage content related with their daily lives with others in real time through various social networking services (SNS) or cloud storage services. Also, as AI technology is applied to a home IoT service, a smart home service is also getting attention again [1], [2].

However, as to SNS and cloud services, it is always pointed out that there exists the possibility/risk of invasion of privacy. For example, while it is very easy to copy and redistribute content which is uploaded on a network, it is very difficult to remove the distributed content from a network. Nevertheless, consumers using SNS and cloud services do not pay their attention to such a clear fact. Hence, such services expose consumer-specific data from a personal/home system to external parties, causing potential security and privacy risks. That is, the possibility of the privacy invasion of both SNS and cloud-based services is still an ongoing problem [3]–[7]. Moreover, cloud-based IoT/home services cause another problem such as interoperability among heterogeneous devices from different vendors [8], [9]. It can cause consumers inconvenience. So, a content-centric networking (CCN) based personal community architecture (VPC) was proposed to solve such privacy concerns. A VPC domain is a consumer-managed local and stand-alone domain which does not require external systems, such as a cloud system for storing content. Hence, consumer's content are not delivered and stored at the external cloud system so that not exposed accidentally.

Virtual private community architecture (VPC) proposed a logical and hierarchical domain structure to intuitively and securely share content only with limited members of personal domains [10], [11]. VPC consumers can build their own domains without aforementioned problems such as privacy invasion and user-inconvenience. As shown in Fig. 1, consumers can build a VPC domain with three-layer: VPC-1 (the set of individual devices), VPC-2 (the set of VPC-1s, e.g., a home domain) and VPC-3 (the set of VPC-2s, e.g., a relative domain). The hierarchical level of VPC is scalable as many as consumers want. Hence, it is expected that VPC can be applied to various services with logical and hierarchical structures. Specially, VPC uses CCN known as a future Internet architecture [12]. To enhance network performance, security, and user experience provided by the Internet, CCN provides various functionalities: A packet routing mechanism based on the identity of content, not on the identity of a host; in-network content caching mechanism; encryption-based access control

Manuscript received March 31, 2018; revised June 22, 2018, September 19, 2018, November 25, 2018, and January 24, 2019; accepted February 18, 2019. Date of publication March 1, 2019; date of current version April 23, 2019. This work was supported by the Basic Science Research Program through the National Research Foundation of Korea funded by the Ministry of Education under Grant 2017R1D1A1B03034215 and Grant 2016R1D1A1B03931628. *(Corresponding author: Daeyoub Kim.)*

D. Kim is with the Department of Information Security, Suwon University, Hwaseong 18323, South Korea (e-mail: daeyoub69@suwon.ac.kr).

J. Lee is with the Department of Smart Information and Telecommunication Engineering, Sangmyung University, Cheonan 31066, South Korea (e-mail: vincent@smu.ac.kr).

Digital Object Identifier 10.1109/TCE.2019.2902412

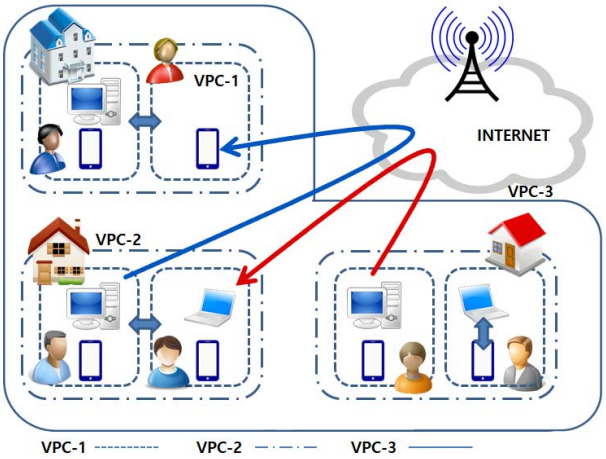

Fig. 1. VPC hierarchical structure. Intuitively VPC-1 is a user, VPC-2 is a family, and VPC-3 is a relative.

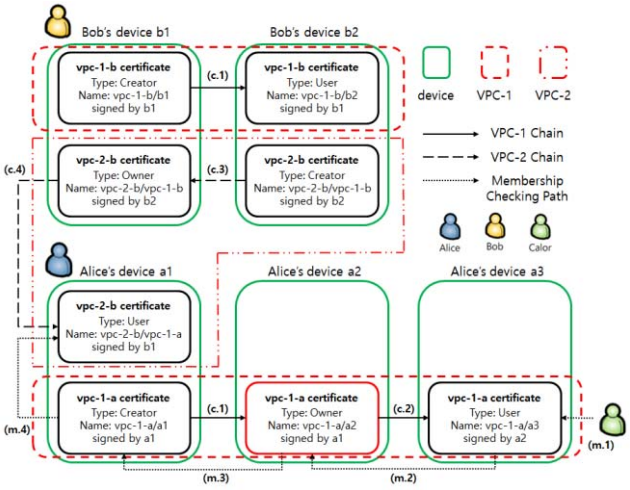

Fig. 2. CCA hierarchical membership for VPC-2 and VPC-1. In step (c.1) and (c.2), Alice and Bob construct their own VPC-1s, respectively. Then in step (c.3) and (c.4), Bob constructs his VPC-2 and invites Alice's VPC-1 to the VPC-2.

for content protection, and so on. Since such CCN functionalities are very usefully for services which share content with others [9], [13], VPC uses CCN as its basic networking technology.

To securely share content between limited members, VPC requires a secure device/member clustering method, essentially. Although various device clustering schemes have been proposed, they are mainly either suggested as network connectivity standards or provided by specific device vendors to build a connection only between their devices. So, in most cases, they require specific authentication services. However, since a consumer domain like a home domain generally consists of devices from several different vendors, a lack of interoperability among heterogeneous devices becomes the main bottleneck which existing device clustering schemes are not widely utilized. Certificate chains based association (CCA) is a device clustering method for VPC under aforementioned heterogeneous devices environment [14], [15]. For that, CCA provides three schemes to securely manage the members of VPC-x which are implemented without using external security systems: an enrollment scheme applied to a hierarchical domain structure, a member authentication based on a hierarchical trust model, and a member authorization like a role based access control. So, CCA showed that a consumer can securely build a VPC domain, even if any external authentication systems like PKI are not utilized at all [15]. However, CCA has several weak points such as the absence of the root authentication of a hierarchical trust model, the absence of revocation management, and a high processing burden caused by a chain model.

Hence, in order that consumers can securely and efficiently share content using VPC, this paper proposes a new device clustering method called as a member list chain and reputation based association scheme (MLC-R_bA). When compared to the CCA, the proposed MLC-R_bA provides meaningful improvements as follows:

- It uses a member list, not the member certificate of CCA to enhance the performance of an authentication procedure.
- It utilizes a reputation-based trust model to authenticate the root of a hierarchical trust model.
- To control revoked devices/members, it additionally defines a revoked-membership status, and then control the access of revoked members using this membership status.
- To improve service congestion caused by a chain model, it uses a domain hub device which is a representative entity publishing a valid membership status list.

Since MLC-R_bA provides these functionalities without using external authentication services provided by specific vendors or organizations, consumers can build/operate a VPC domain with their own devices more securely and reliably, even if these devices are from different vendors.

II. Certificate Chain Based Association Scheme

This section briefly describes CCA for VPC. Since VPC is a hierarchical structure, a member of VPC-x ($x > 1$) is the logical and virtual group of devices. If a member of VPC-x accesses to VPC-x using a device of the member, the device is called as a member-device of VPC-x.

A. CCA Overview

CCA, a device clustering scheme for VPC, is a hierarchical membership association scheme based on a membership certificate chain. VPC is a hierarchical structure of domains. So, if VPC-y_1 and VPC-y_2 are the members of VPC-x, every member of VPC-y_1 must be able to recognize that every member of VPC-y_2 is in the same VPC-x. However, if each member of VPC-y_1 must perform an enrollment process with all members of VPC-y_2 one-by-one, it causes inconvenient to consumers. To solve such a problem, CCA utilizes a concept of a domain representative. So, CCA defines three membership types: *creator*, *owner*, and *user*.

- A member having *creator-membership* is the original builder of VPC-x. It is the representative of VPC-x

TABLE I
THE STRUCTURE OF VPC-X-LIST

Field	Description
Name (NA)	[CCN]/vpc.[x]/[ID]/vpc-[x]-List: [CCN] is the name prefix of CCN packet. [x] is the hierarchical level of VPC domain (x= 1 or 2 or 3). [ID] is a unique device identity of the creator of this VPC-x. Specially, CR means [CCN]/vpc.[x]/[ID].
Version (VR)	The time value to describe when the list is generated.
Member List (ML)	The list of member entries (*ME*s) of the VPC-x. Table II describes the structure of *ME*.
Sign Inform. (SI)	It identifies both a signature algorithm and the information of a signer's public key. It also says a publisher who generates this list.
Signature (ST)	It is a digital signature computed upon above fields, {*NA*, *VR*, *ML*, and *SI*}.

TABLE II
THE STRUCTURE OF A ME OF VPC-X-LIST.ML

Field	Description
Name (mN)	It is an identity of a member relevant to this ME. In vpc-x-list, if x = 1, a member of VPC-x is a device. Hence, *mN* is a device ID. Assume that each device has a unique device key pair of a public key and a private key. The hash value of the public key of each device as a unique identity of the device. If x=2 (or 3), a member of VPC-x is a VPC-1 (or VPC-2). In this case, *mN* is *vpc-1-List.CR* (or *vpc-2-List.CR*).
Membership (mM)	It is a member type. The *type* is 'C' (creator) or 'O' (owner) or 'U' (user) or 'R' (revoked-member). In the case of 'R', it should be mentioned along with the original type, like 'C:R' or 'O:R' or 'U:R'. It means that the revoked member previously had 'O' or 'U' membership type.
Time Stamp (mT)	It is the time when this member joins/leaves the VPC-x.
Inviter (mI)	It is a member' identity that invites this member to the VPC-x.
Sign Record. (mR)	It identifies both a signature algorithm and the information of a signer's public key.
Signature (mS)	It is a reputation data of the M_c of the VPC-x. It is a digital signature that this member generates on *vpc-x-List.CR* using its device private key.

and can assign *owner-membership* to other members of VPC-x. It also has the *owner-membership* of VPC-x.
- A member having *owner-membership* can invite others to VPC-x and assign *user-membership* to invitees.
- A member having *user-membership* can access to VPC-x as a member of VPC-x, but cannot invite others to VPC-x.

For a hierarchical membership association, CCA proposes that if VPC-y enrolls as a member of VPC-x using a member-device of VPC-y, an owner of VPC-x does not issue a membership certificate of VPC-x to the enrolling device, but to the member with *creator-membership* of VPC-y. Then all members of VPC-y automatically have access entitlement to VPC-x through inheritance without an additional registration process.

Fig. 2 shows an example how to verify the membership of VPC-2 to describe the hierarchical membership association of CCA.

(1) *Provision (Step c.1 and c.2):* Alice constructs VPC-1 named as "vpc-1-a" which consists of three devices (a_1, a_2, and a_3). Bob's VPC-1 named as "vpc-1-b" consists of two devices (b_1 and b_2).

(2) *VPC-2 Enrollment (Step c.3 and c.4):* Bob makes VPC-2 named as "vpc-2-b" using a device b_2. To invite "vpc-1-b" as a member of VPC-2, b_2 issues a membership certificate of "vpc-2-b" to b_1 (the creator of "vpc-1-b"). Then "vpc-1-b" becomes a member of "vpc-2-b". So, all members of "vpc-1-b" can access to "vpc-2-b". Similarly, if Alice's a_3 asks b_1 having an owner certificate of "vpc-2-b" to issue a membership certificate of "vpc-2-b", b_1 issues a certificate for a_1 (the creator of "vpc-1-a"). Now, "vpc-1-a" is regarded as a member of "vpc-2-b".

(3) *VPC 2 Member Authentication (Step m.1∼m.4):* To confirm whether a_3 can access to "vpc-2-b" or not, a verifier check whether "vpc-2-b"-certificate has been issued to a_1 (the creator of "vpc-1-a"). In step m.1∼m.3, Calor first checks a_3 is a member of "vpc-1-a". For that, she verifies "vpc-1-a"-certificates of a_1, a_2, and a_3 in order. If these certificates are all valid, then a_3 is a valid member of "vpc-1-a". Next, in step m.4, she checks whether "vpc-1-a" is a valid member of "vpc-2-b". For that, she verifies "vpc-2-b"-certificate issued to a_1. If valid, she regards a_3 as a member-device of "vpc-2-b".

B. CCA Weaknesses

Generally, a device clustering scheme for a domain service like a smart home supports four security functions as follows: Device identification, authentication, authorization, and revocation.

- *Device Identification:* A consumer can identify a device which tries to access to his/her domain. For that, each device should have a unique identity.
- *Device Authentication:* When a device accesses to a domain, the members of the domain can authenticate the device.
- *Device Authorization:* When a device tries to use the content/resources of a domain, the authorized members of the domain can verify the access entitlement of the device. Also, if a device tries to access the resources/content of the domain beyond its access entitlement, such a trial should be discovered and properly controlled.
- *Device Revocation:* If the access entitlement of a device is revoked by the authorized member of a domain, other devices in the domain can recognize this fact. So, the device must properly be controlled so that the device can neither access to the domain nor perform any activities needing entitlement.

Considering such security function requirements, CCA still has several weak points as follows:

First, CCA uses a hierarchical trust model like PKI. When applying a hierarchical trust model, it is important but very difficult to define how to authenticate the highest component (i.e., root) of a hierarchical trust model. To authenticate the creator of VPC-x which is the root of CCA hierarchical trust model, CCA uses a self-signed membership certificate which is similar to the root CA certificate of PKI. However, unlike PKI, since CCA does not use a trust authority which is publicly trusted, the self-signed certificate of CCA can be forged. It makes VPC vulnerable to a spoofing attack. That is, an attacker can impersonate the creator of VPC-x if the attacker generates both the fake key pair of the creator of VPC-x and the fake creator certificate of VPC-x, and then distribute them over CCN.

Second, if either a consumer loses his/her device of VPC-x or the key pair of a device is exposed, such a device should be controlled not to access to VPC-x. However, CCA does not consider how to control the revoked devices/members of VPC-x. Revocation scheme looks an unnecessary requirement when considering only a device pairing method. But, since the absence of revocation mechanism makes it possible for malicious users to illegally access VPC-x, revocation management is the essential component of a VPC trust model.

Also, although consumers can utilize traditional revocation systems like PKI/CRL, another problem may occur when applying such systems to VPC. That is, if the membership of a VPC-x member is revoked so that the member's certificate becomes invalid, other members invited by the revoked member will be regarded as invalid members because their certificate cannot be authenticated. In this case, the members invited by the revoked member must re-enroll in VPC-x to access to VPC-x. For example, as shown in Fig. 2, if a_2 is revoked from "vpc-1-a", regardless of that a_3 is not revoked, the "vpc-1-a"-certificate of a_3 becomes unverifiable because the certificate chain of a_3 includes the invalid certificate issued to a_2. In this case, to access "vpc-1-a", another owner of "vpc-1-a" must invite a_3 to "vpc-1-a" and issue a new "vpc-1-a"-certificate to a_3. However, such a re-enrollment process causes consumers inconvenience.

Third, CCA utilizes a certificate chain-based trust model. A chain-based model can cause inconvenience for controlling revoked members as previously mentioned as well as a high processing burden if VPC-x consists of large numbers of devices. When verifying a member of VPC-x, a verifier generally checks three certificates, from a user-membership certificate to a creator-membership certificate. Hence, to check whether a device is a member-device of VPC-3, a verifier should check the validity of 9 certificates, which leads to long service latency.

III. Member List Chain and Reputation Based Association Scheme

To improve the weak points of CCA so that consumers can securely and efficiently build their VPC domains, this paper proposes a new device clustering scheme, MLC-R_bA.

A. Member List and Membership Type

MLC-R_bA proposes to publish a member list (denoted with *vpc-x-List*) of VPC-x in order to publicly announce the valid members of VPC-x. Tables I and II show the structure of *vpc-x-List*. Since *vpc-x-List* enumerates both members' identities (*vpc-x-List.ML.ME.mN*) and their membership statuses (*vpc-x-List.ML.ME.mM*), a verifier can verify the members of VPC-x if the verifier simply checks the latest and valid *vpc-x-List* of VPC-x.

MLC-R_bA uses four types of membership status as follows: *creator*, *owner*, *user*, and *revoked-member*.

- A member (M_c) having *creator-membership* is the creator of VPC-x. M_c can assign *owner-membership* to other members. M_c also has the *owner-membership* of VPC-x.
- A member (M_o) having *owner-membership* can invite others to VPC-x and assign *user-membership* to the invitees. Also, M_o can revoke other members' membership except for M_c. Only M_o can edit the *vpc-x-List* of VPC-x and republish the modified list when the membership statuses of members of VPC-x are changed.
- A member (M_u) having *user-membership* can access content/resources in VPC-x. However, M_u can neither invite a new member nor revoke the membership of other members of VPC-x.
- A member (M_r) having a *revoked-member-membership* cannot access VPC-x anymore even though it was the valid member of VPC-x before.

Since these membership statuses define the role of each member of VPC-x, it seems to be a role-based access control scheme (RBA), especially, to be a hierarchical RBA. Since the consumers of RBA are generally enterprises/organizations, the structure/kind of roles defined by a RBA system is generally very complex. But, MLC-R_bA defines only one specific role, that is, the role of an inviter. Actually, both M_c and M_o have the role of an inviter for creating/managing VPC-x. Hence, MLC-R_bA simplifies a hierarchical RBA so as to be suitable to VPC.

B. VPC-x Member Authentication

To authenticate a member of VPC-x, a verifier must first check the latest *vpc-x-List* of VPC-x after getting it through CCN. Since only "valid" owners of VPC-x can publish the *vpc-x-List* of VPC-x, the verifier must authenticate the publisher of the latest *vpc-x-List*. For that, the verifier checks two facts as follows:

- The publisher of the latest *vpc-x-List* is the valid owner of VPC-x that is invited by the creator of VPC-x.
- The creator of VPC-x inviting the publisher is the valid creator of the VPC-x.

Then the verifier confirms the membership status of a member which is recorded in *vpc-x-List* as follows.

1) VPC-x Creator Verification: MLC-R_bA assumes that an invitee can identify and verify its inviter so that it would trust the inviter. Actually, using the enrollment process of CCA [14], an invitee can identify its inviter. So, if the owner (M_o) of VPC-x invites a new member, the new member trusts M_o. Also, since the creator (M_c) of VPC-x has invited M_o, M_o trusts M_c. Hence, it seems reasonable to assume that a new member (i.e., invitee) can trust M_c because it would trust M_o. Using such an assumption, MLC-R_bA proposes that every member of VPC-x together participates in generating the reputations (i.e., the proof of authentication) of the M_c of VPC-x, which will be used to authenticate M_c: When a new member (invitee) joins VPC-x, the M_o (inviter) of the VPC-x provides the invitee with M_c's identity and then asks the invitee to generate M_c's reputation (*vpc-x-List.ML.ME.mS*). If the invitee would trust M_o, the invitee generates/sends a reputation (*mS*) to M_o. Then M_o generates a member entity data (*ME*) for the invitee and adds the generated *ME* to the latest *vpc-x-List.ML*. Finally, M_o publishes the modified *vpc-x-List* to announce the information of the new member.

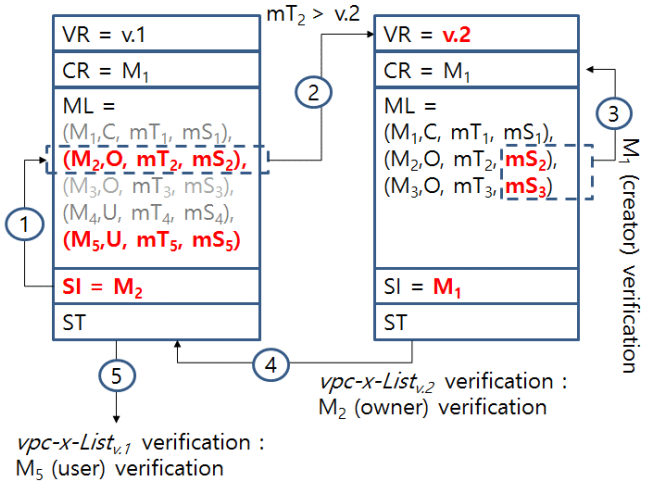

Fig. 3. A member verification for VPC-x: ME = {mN, mM, mT, mS}.

After that, if a verifier checks all *ME.mS's* of the latest *vpc-x-List*, it can confirm that all members of VPC-x trust M_c. If the verifier trusts some members of VPC-x, it is sufficient to check only *ME.mS*'s generated by the trusted members. If these *mS*'s are valid, the verifier regards M_c as the valid creator of VPC-x.

2) VPC-x Owner Verification: To verify an owner of VPC-x, a verifier must check *vpc-x-List* published when the creator of VPC-x invites the owner as well as the latest *vpc-x-List*. Fig. 3 describes an example to authenticate the owner of the VPC-x. It assumes a scenario that for verifying a member (M_5) of VPC-x, a verifier gets the latest *vpc-x-List* with version *v.1* (denoted with *vpc-x-List*$_{v.1}$) published by the owner (M_2) of VPC-x. The verifier checks that M_2 is the valid M_o of VPC-x as follows: Let ME_i be the member entity of M_i.

(1) To verify M_2 publishing *vpc-x-List*$_{v.1}$, the verifier reads the ME_2 of *vpc-x-List*$_{v.1}$.

(2) Then the verifier gets another *vpc-x-List* that was published shortly after $ME_2.mT$. Let the version of the *vpc-x-List* be *v.*2 and be denoted with *vpc-x-List*$_{v.2}$. In practice, since M_2 may be the owner of VPC-x, the inviter of M_2 must be M_c. Hence, *vpc-x-List*$_{v.2}$ was published by M_c when inviting M_2.

(3) Then the verifier checks that the publisher (M_1) of *vpc-x-List*$_{v.2}$ is the valid M_c of VPC-x using *vpc-x-List*$_{v.2}$*.ML.ME.mS's*, that is, mS_2 and mS_3 as described in previous section.

(4) The verifier confirms *vpc-x-List*$_{v.2}$*.ST* that is generated by a valid M_1. If valid, it regards *vpc-x-List*$_{v.2}$ as a valid list and then trusts M_2 as the valid M_o of VPC-x.

3) VPC-x User Verification: As shown in Fig. 3, after verifying the publisher (M_2) of *vpc-x-List*$_{v.1}$, the verifier checks *vpc-x-List*$_{v.1}$*.ST* generated by M_2. If valid, it checks whether *vpc-x-List*$_{v.1}$*.ML.ME*$_5$*.mM* is not 'R'. If not, it regards M_5 as the valid member of VPC-x.

Unlike CCA, when verifying M_5, it is not a necessary condition that M_2 is the inviter of M_5. Instead, it is sufficient that M_2 is one of the valid owners of VPC-x. So, even if the inviter of M_5 is revoked, M_5 can still be verified as the valid member of VPC-x. Hence, M_5 does not need reregistration.

4) Hierarchical Authentication: Since VPC has a hierarchical structure, it is necessary to check whether a device of VPC-1 is also a member-device of VPC-2/3 containing VPC-1. For that, like CCA, MLC-R_bA utilizes a recursive procedure to check the hierarchical membership of VPC. That is, if the creator of VPC-1 is a valid member of VPC-2, then every device of VPC-1 is accessible to VPC-2. If the creator of the VPC-2 is the valid member of VPC-3, then every device of the VPC-1 is also accessible to VPC-3.

C. VPC-x Member Revocation

A revoked member of VPC-x should not be able to access VPC-x. A general revocation solution like PKI/CRL is that a trusted and centralized authority publishes a revocation list recording the identities of revoked members. However, since MLC-R_bA does not utilize external specific authentication systems, a M_o of VPC-x directly change the membership status (i.e., *vpc-x-list.ML.ME*) of a revoked member, and then publicly announces that the member has been revoked. For simplification, assume that VPC-x becomes permanently closed if the M_c of VPC-x becomes revoked. Hence, MLC-R_bA only considers how to revoke the membership of M_o/M_u as follows:

(1) If M_o revokes one of members except M_c of VPC-x, it obtains the latest *vpc-x-List*. Then it reads the *vpc-x-List.ML.ME* of the member which it will revoke.

(2) Then it changes the *ME.mM* from 'U' (or 'O') to 'U:R' (or 'O:R').

(3) Finally, it publishes the updated *vpc-x-List* to publicly announce the revocation situation of the member.

To check whether M_o (or M_u) is revoked, a verifier gets the latest *vpc-x-List*. Then it checks the *vpc-x-List.ML.ME.mM* of the M_o (or M_u). If the *mM* is 'O:R' (or 'U:R'), it regards the M_o (or M_u) as being revoked.

IV. MLC-RbA Implementation

Generally, it can happen that a revocation list published periodically by a trust authority does not include revocation information which was generated very recently. Also, if there are several publishers issuing revocation lists, then there can be several branches of a list at the same time. Hence, such time synchronization is a common issue of revocation schemes like PKI/CRL. So alternatives such as OCSP are being utilized.

Since every M_o of VPC-x can update/publish *vpc-x-List* of VPC-x, it can also happen that two M_o's publish two *vpc-x-List*s referring to the same *vpc-x-List*. In this case, if these two *vpc-x-List*s have the same version, that is, they are published at the same time, a verifier randomly gets only one of the two *vpc-x-List* through CCN as the latest *vpc-x-List*. Otherwise, the old version *vpc-x-List* among these two *vpc-x-List*s will not be utilized by verifiers because verifiers always get the latest *vpc-x-List*. If a verifier cannot refer to one of these two *vpc-x-List*s, some authentication information can be missing when authenticating members. Then such an authentication result becomes unreliable. Hence, it is important how to synchronize *vpc-x-List*.

In addition, since MLC-R_bA uses the hierarchical chain of *vpc-x-List*s, if the latest *vpc-x-List* is not published by the M_c of VPC-x, a verifier must additionally get another previous version of the latest *vpc-x-List* such that it was published by the M_c. However, it causes a service delay.

To solve such problems, this paper proposes to utilize a device (a domain management hub, DMH) for each VPC-2.

A. Domain Management Hub

To utilize DMH, assume that

- In each VPC-2, there is a member-device, DMH, which is in operation 24 hours a day. For example, the home gateway of a home network service is proper to take on the role of DMH.
- When a member-device of VPC-2 communicates with other member-devices in VPC-3, the member-device in VPC-2 should communicate through the DMH of VPC-2. For that, VPC can use a custodian-based forwarding model for CCN [16].

DMH-based VPC-x construction procedure is very simple as follows:

(1) A consumer first constructs VPC-1 (denoted VPC-1_{DMH}) consisting of only DMH.

(2) Then the consumer constructs another VPC-1 which consists of other devices. That is, there are two VPC-1s generated by the consumer.

(3) The consumer then constructs VPC-2 as well as VPC-3 using the DMH. In this case, VPC-$1_{DMH}$is the creator of VPC-2 and the VPC-2 generated by VPC-1_{DMH} is the creator of VPC-3.

B. Republishing VPC-2-List and VPC-3-List

Let DMH_c be a member-device of the M_c of VPC-2/3. That is, DMH_c is a device used for generating VPC-2/3.

For simple description, distinguish two kinds of *vpc-x-List* logically as follows.

- *c-vpc-x-List* is published by the M_c of VPC-x. A verifier utilizes the latest *c-vpc-x-List* to authenticate members of the VPC-x.
- *o-vpc-x-List* is published by the M_o of VPC-x. Only M_c utilizes *o-vpc-x-List* for updating *c-vpc-x-List*.

DMH_c continuously and periodically updates *vpc-x-List* based on *vpc-x-List*s published by the M_o's of VPC-x before. That is, DMH_c obtains all *o-vpc-x-List*s recently published by every M_o of VPC-x and then updates *c-vpc-x-List* based on these *o-vpc-x-List*s, as described in Table III.

Now, a verifier does not need to get two versions of *vpc-x-List* for verifying a member of VPC-x. Instead, it is sufficient to obtain and verify the latest *c-vpc-x-List*. Also, since *c-vpc-x-List* includes every *o-vpc-x-List* recently published by every M_o, it can solve both time synchronization and authentication information omission.

Since a very small size domain does not need several owners for managing the memberships of the domain, it is sufficient to assign both creator- and owner-membership only to DMH_c. In this case, since only DMH_c can publish vpc-x-List, it does not need to consider a time synchronization problem.

TABLE III
REPUBLISHING VPC-X-LIST PSEUDO-CODE

```
republish_vpc_x_List:
input: the latest c-vpc-x-List,  o-vpc-x-List [1,…n]
output: update c-vpc-x-List
sort o-vpc-x-List [1,…n] according to their version from the earliest;
for k = 1 to n :
  for each x = index of o-vpc-x-List[k].ML.ME[]:
    read the ME[x] of o-vpc-x-List[k].ML ;
    compare ME[x]  with  every c-vpc-x-List.ML.MEs ;
    if ME[x] is not matched to any c-vpc-x-List.ML.ME then
      insert ME[x] into c-vpc-x-List.ML ;
    else
      read ME[p] of c-vpc-x-List.ML that is matched to ME[x] ;
      if ME[p].mM is not equal to ME[x].mM then
        if ME[x].mT is later than ME[p].mT then
          update c-vpc-x-List.ML.ME[p] with  ME[x] ;
        end if
      else
        if ME[p].mT is not equal to ME[x].mT then
          if  ME[p].mM is not 'Revoked' and
              ME[x].mT is earlier than ME[p].mT then
              update c-vpc-x-List.ML.ME[p].mT with ME[x].mT ;
          else if ME[p].mM is 'Revoked' and
                      ME[x].mT is later than ME[p].mT  then
                  update c-vpc-x-List.ML.ME[p].mT with ME[x].mT;
                end if
           end if
        end if
      end if
    end if
  end for
end for
return updated c-vpc-x-List ;
```

V. SECURITY AND PERFORMANCE EVALUATION

A. Security Evaluation

A secure device clustering scheme should satisfy functional requirements described in Section II: Device identification, authentication, authorization, and revocation. In this section, MLC-R_bA is evaluated whether it meets these requirements with regard to attacks/issues such as spoofing attacks and time synchronization.

1) Spoofing Attack: In general, a spoofing attack is related to identification, authentication, and authorization.

- *Device Identification:* To guarantee the uniqueness of a device identity, MLC-R_bA utilizes the hash value of a device public key as a unique identity of the device. Enrollment process presented in [14] asks a device to

generate a public and private key pair. Also, it is possible to use a key pair generated by device vendors. Since the key pair of a device is unique, the identity of the device is also unique.

- *Device/Member Authentication:* To impersonate the valid member of VPC-x, an attacker should forge and publish *vpc-x-List* which recorded the attacker as a valid member. To prevent such a spoofing attack, MLC-R_bA utilizes a hierarchical trust model with 3 layers: M_u, M_o, and M_c. A member authentication process is always progressed, from the authentication of M_c to the authentication of M_u, hierarchically and sequentially. So, the attacker must forge the fake reputations of M_c as if they are real reputations of M_c collaboratively generated by every valid member of VPC-x. To generate such fake reputations of M_c, the attacker must be able to generate the digital signatures of all valid members. Although it is not impossible, at least it is very difficult.
- *Device/Member Authorization:* MLC-R_bA controls the access/activity of a member according to the type and status of its membership recorded on the latest *vpc-x-List*. It is similar to a hierarchical role-based access scheme. To steal someone's access/activity entitlement, an attacker should modify the latest *vpc-x-List*. Since a verifier should check the validity of M_c when verifying *vpc-x-List*, the attacker must forge the reputations of M_c. However, it is very difficult as previously mentioned.

2) Revocation and Time Synchronization Attack: To reliably handle an unexpected situation such as which a consumer loses a device or which a device key is externally exposed, MLC-R_bA adds a new membership status (i.e., a revoked-membership), and then controls the access of revoked devices/members. If such situations are happened so that it is needed to revoke a member of VPC-x, a valid M_o of VPC-x changes the membership status of the member recorded on the latest *vpc-x-List*, and then publishes the modified *vpc-x-List*. Then, if a verifier checks the *ME.mT* of the member recorded on the modified and latest *vpc-x-List*, the verifier can know the member cannot access VPC-x anymore.

If a M_o of VPC-x is revoked, another M_o of VPC-x modifies the latest *vpc-x-List* so as to change the membership-status of the revoked M_o. But, a member invited by the revoked M_o is still recorded as a valid member in the modified *vpc-x-List*. Hence, a member invited by the revoked M_o is still authenticated as a valid member without going through a re-registration process.

To maintain time synchronization between several *vpc-x-List*s published by the M_o's of VPC-x, MLC-R_bA utilizes DMH so that it periodically publishes the latest *vpc-x-List* referring to all *vpc-x-List*s generated by the M_o's of VPC-x. Since a verifier checks only *vpc-x-List* generated by DMH_c, it can check all the authentication information generated by the M_o's of VPC-x without missing. Also, using DMH, MLC-R_bA can solve a service delay problem caused by a chain-based trust model as presented in next section.

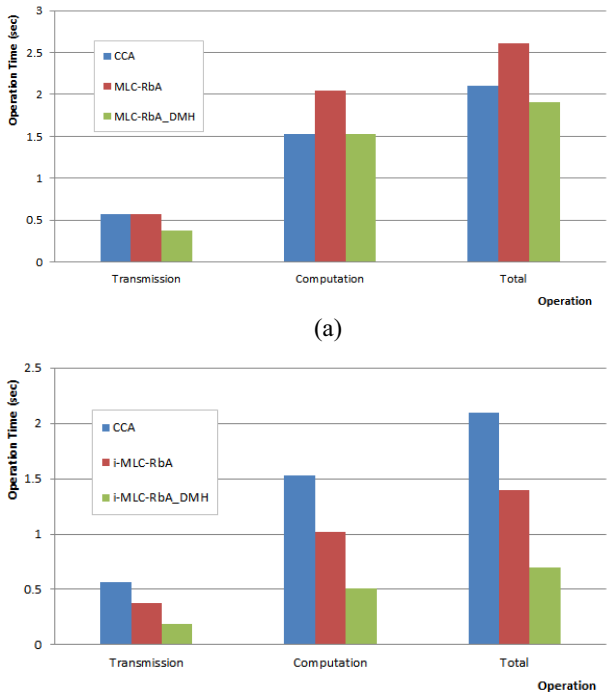

Fig. 4. Performance evaluation of MLC-R_bA / MLC-R_bA$_{DMH}$s. (a) MLC-R_bA and MLC-R_bA$_{DMH}$ Performance. (b) i-MLC-R_bA and i-MLC-R_bA$_{DMH}$ Performance.

B. Performance Evaluation

To evaluate performances, we utilized basic CCN configuration and constructed a domain with a hierarchical network topology with depth 3. The domain is regarded as a VPC-3 domain.

- VPC-1 has three devices except for VPC-1$_{DMH}$. Hence, each VPC-1 is consisted of four devices.
- VPC-2 domains consists of four VPC-1 domains respectively. Hence, each VPC-2 is consisted of 16 devices.
- VPC-3 domain consists of two VPC-2 domains. Hence, VPC-3 is consisted of 32 devices.
- Each device has a storage which can cache 1000 content and the life time of a cache is 24 hours.

Suppose that random devices request content which is randomly selected by them for simulation. We assume a case to revoke 10% of devices of the domain sequentially and a case to revoke both a VPC-2 domain and a VPC-1 domain. Also, for each time a device is discarded, DMH publishes an updated membership list as soon as each time when a device is discarded. Then we measures both transmission overheads and computation overheads for verifying transmitted each content.

1) A Valid Member-Device Check: When verifying a valid member-device of VPC-3, while CCA verifies 9 certificates, MLC-R_bA verifies 6 member-lists. If DMH is applied to MLC-R_bA (denoted with MLC-R_bA$_{DMH}$), it is sufficient to verify 3 member-lists. Fig. 4-(a) shows the result of a performance evaluation comparing MLC-R_bA/ MLC-R_bA$_{DMH}$ with CCA. A verifier always select two *ME*'s in *vpc-x-List.ML* in order to verify the M_c of VPC-x. Since MLC-R_bA asks a verifier to confirm two member lists for each VPC-x, MLC-R_bA may

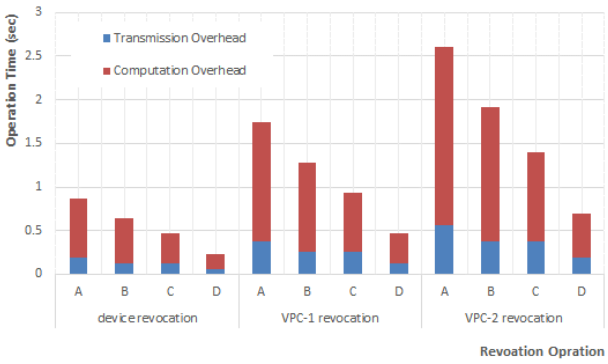

Fig. 5. Revocation check performance evaluation. "A" is MLC-R_bA. "B" is MLC-R_bA_{DMH}. "C' is i-MLC-R_bA. "D" is i-MLC-R_bA_{DMH}.

increase transmission and computation overheads sometimes as compared with CCA. However, MLC-R_bA_{DMH} always reduces the transmission and computation overheads. As shown in Fig. 4-(a), the performance of MLC-R_bA_{DMH} is about 20% more efficient than that of CCA.

Since MLC-R_bA asks a verifier to confirm all/some *ME.mS*'s of *vpc-x-List.ML* for verifying the M_c of VPC-x, it may cause performance degradation of MLC-R_bA. To rectify such a problem, it assumes that a verifier is also a member of VPC-x and that each member of the VPC-x saves a valid device ID of the M_c of the VPC-x when joining to the VPC-x. Then, the member just compares the saved device ID with the *vpc-x-List.CR* to verify the M_c of the VPC-x. This method is denoted with i-MLC-R_bA (or i-MLC-R_bA_{DMH}). Fig. 4-(b) shows performance comparison results in terms of operational time among i-MLC-R_bA, i-MLC-R_bA_{DMH}, and CCA. As shown in Fig. 4-(b), both i-MLC-R_bA and i-MLC-R_bA_{DMH} are more efficient than CCA. Interestingly, the performance of i-MLC-R_bA_{DMH} is about 70 percent more efficient than that of CCA.

2) A Revoked-Member Check: Fig. 5 shows the operation time it takes to verify if a member is revoked. Fig.5-{A, B, C, D} denotes MLC-R_bA, MLC-R_bA_{DMH}, i-MLC-R_bA, i-MLC-R_bA_{DMH}, respectively. We considered three cases: A device of VPC-1 is revoked, a VPC-1 domain is revoked, and a VPC-2 domain is revoked. Generally, a PKI solution takes more than 0.3 second to check a revocation list [17]. Since an end-user PKI certificate is generally issued by a sub-root CA, it is needed to check two kinds of revocation list. So, it takes more than 0.6 second to check the status of the user certificate. The evaluation of [17] is for one-layer member revocation like the device revocation of VPC. So, as shown in Fig. 5, when considering a device revocation check process, although MLC-R_bA shows 0.86 seconds to check the member status of a revoked device of VPC-1, the performance of i-MLC-R_bA_{DMH} is more efficient than a general PKI solution.

VI. Conclusion

CCN is the most popular technology of the many future Internet technologies. However, the main research topics of CCN so far have been concentrating on both a packet routing scheme and a packet naming scheme. In recent years, some researchers have begun to apply CCN to IoT-based applications/services such as connected vehicles/homes. VPC is one of such applications/services of CCN. Although CCA was first proposed to implement VPC, as mentioned in [15] CCA showed technical feasibility that consumers can build a secure personal community domain on their own without the support of any external authentication systems like PKI.

However, CCA still has several weak points such as the absence of the highest component authentication of a hierarchical trust model, the absence of revocation management, and a service delay caused from a chain model. To solve such weak points of CCA so that consumers can build their VPC more securely and efficiently, MLC-R_bA (specially, MLC-R_bA_{DMH}) proposes an enhance local trust management model and provides the efficient operation method of the proposed trust model.

This paper makes the main two points. First, MLC-R_bA provides a enhance trust model for VPC. For that, MLC-R_bA proposes both the root authentication scheme of a hierarchical trust model and a revocation management scheme which are not need external authentication systems. Generally, in a hierarchical trust model, it is assumed that the trust of the highest component like a root CA of PKI is publicly recognized and accepted. MLC-R_bA utilizes the reputations of the creator of VPC-x which are collaboratively generated by all members of VPC-x. Since VPC-x is a closed domain utilized only by the members of VPC-x, it is reasonable that the members of VPC-x directly agree on the creator's trust of VPC-x. Especially, MLC-R_bA can prevent a spoofing attack since an attacker cannot forge these all reputations. Moreover, since MLC-R_bA proposes an revocation management scheme. So, it can handle unexpected situations which a consumer loses a device or which a device key is externally exposed. Furthermore, even if the owner of VPC-x is revoked, since other members invited by the revoked owner are not affected at all by the revocation of their inviter, the proposed revocation management scheme is suitable to a hierarchical trust model for the viewpoint of usability.

Second, MLC-R_bA provides enhanced operation efficiency for VPC. A chain-based trust model generally asks consumers to verify all authentication components (e.g., certificates) of an authentication chain for verifying the end component of the chain. However, if a domain size is large so that the length of an authentication chain is long, it causes long service latency. Also, revocation management schemes periodically publishing a revocation list generally encounter a time synchronization problem. MLC-R_bA uses DMH which is a representative entity publishing a valid membership status list. Using DMH, MLC-R_bA provides the performance improvement of an authentication procedure as well as the solutions for the time synchronization problem of a revocation management.

References

[1] Emarketer.com. *Number of Social Network Users Worldwide From 2010 to 2021*. Accessed: Jul. 2017. [Online]. Available: https://www.statista.com/statistics/278414/number-of-worldwide-social-network-users/

[2] S. J. Vaughan-Nichols, *The Top 10 Personal Cloud-Storage Services*, zdnet, Feb. 25, 2013. [Online]. Available: http://www.zdnet.com/article/the-top-10-personal-cloud-storage-services
[3] P. Powale and G. D. Bhutkar, "Overview of privacy in social networking sites (SNS)," *Int. J. Comput. Appl.*, vol. 74, no. 19, pp. 39–46, Jul. 2013.
[4] Y. Al-Saggaf and M. Islam, "Privacy in social network sites (SNS): The threats form data mining," *Int. J. Commun. Ethics*, vol. 9, no. 4, pp. 32–40, 2012.
[5] C. Soghoian, "Caught in the cloud: Privacy, encryption, and government back doors in the Web 2.0 era," document 2009-07, Berkman Center Res., Cambridge, MA, USA, Oct. 2012.
[6] M. B. Mollaha, M. A. K. Azada, and A. Vasilakosb, "Security and privacy challenges in mobile cloud computing: Survey and way ahead," *J. Netw. Comput. Appl.*, vol. 84, no. 15, pp. 38–54, 2017.
[7] S. Singh, Y.-S. Jeong, and J. H. Park, "A survey on cloud computing security: Issues, threats, and solutions," *J. Netw. Comput. Appl.*, vol. 75, pp. 200–222, Nov. 2016.
[8] R. Brennan *et al.*, "Consumer-managed federated homes," *IEEE Commun. Mag.*, vol. 52, no. 6, pp. 194–201, Jun. 2014.
[9] W. Shang, Z. Wang, A. Afanasyev, J. Burke, and L. Zhang, "Breaking out of the cloud: Local trust management and rendezvous in named data networking of things," in *Proc. 1st IEEE Int. Conf. IoT Design Implement.*, Pittsburgh, PA, USA, Apr. 2017, pp. 3–14.
[10] J. Kim, M.-W. Jang, B.-J. Lee, and K. Kim, "Content-centric network-based virtual private community," in *Proc. IEEE Int. Conf. Consum. Electron.*, Las Vegas, NV, USA, Jan. 2011, pp. 843–844.
[11] M. Huh, D. Kim, E. Kim, and B.-J. Lee, "Secure virtual personal cloud service based on CCN/VPC," in *Proc. IEEE Int. Conf. Consum. Electron.*, Las Vegas, NV, USA, Jan. 2012, pp. 642–643.
[12] V. Jacobson *et al.*, "Networking named content," *Commun. ACM*, vol. 55, no. 1, pp. 117–124, 2012.
[13] O. Waltari and J. Kangasharju, "Content-centric networking in the Internet of Things," in *Proc. 13th IEEE Annu. Consum. Commun. Netw. Conf. (CCNC)*, Las Vegas, NV, USA, Jan. 2016, pp. 73–78.
[14] D. Kim and J. Lee, "CCN-based virtual private community for extended home media service," *IEEE Trans. Consum. Electron.*, vol. 57, no. 2, pp. 532–540, May 2011.
[15] K. Pentikousis *et al.*, "Information-centric networking: Baseline scenarios," IETF, Fremont, CA, USA, RFC 7476, Mar. 2015. [Online]. Available: https://tools.ietf.org/html/rfc7476
[16] V. Jacobson *et al.*, "Custodian-based information sharing," *IEEE Commun. Mag.*, vol. 50, no. 7, pp. 38–43, Jul. 2012.
[17] G. Denker, J. Millen, and Y. Miyake, "PKI and revocation survey," Comput. Sci. Lab., SRI Int., Menlo Park, CA, USA, SRI Rep. SRI-CSL-2000-01, Aug. 2000.

Daeyoub Kim (M'09) received the B.S., M.S., and Ph.D. degrees in mathematics from Korea University, South Korea, in 1994, 1997, and 2000, respectively. He has been with Telemann since 1997 and developed CAS. He joined Secui.com in 2000 and developed PKI. From 2002 to 2011, he was a Senior Researcher with Samsung Electronics and developed content protection schemes and future Internet security. He is currently an Assistant Professor with the Department of Information Security, University of Suwon. His research interests include DRM/CAS, home network, e-health care and cloud service security, and future Internet security.

Jihoon Lee (M'09) received the B.S., M.S, and Ph.D. degrees in electronics engineering from Korea University, Seoul, South Korea, in 1996, 1998, and 2001, respectively. From 2002 to 2011, he was with Samsung Electronics as a Senior Research Member. He is currently an Associate Professor with the Department of Smart Information and Telecommunication Engineering, Sangmyung University. His research interests include information centric networking, Internet of Things, mobile cloud, and network security.