Mandatory Content Access Control for Privacy Protection in Information Centric Networks

Qi Li, *Member, IEEE*, Ravi Sandhu, *Fellow, IEEE*, Xinwen Zhang, *Member, IEEE*, and Mingwei Xu, *Member, IEEE*

Abstract—Several Information Centric Network (ICN) architectures have been proposed as candidates for the future Internet, aiming to solve several salient problems in the current IP-based Internet architecture such as mobility, content dissemination and multi-path forwarding. In general, security and privacy are considered as essential requirements in ICN. However, existing ICN designs lack built-in privacy protection for content providers (CPs), e.g., any router in an Internet Service Provider in ICN can cache any content, which may result in information leakage. In this paper, we propose Mandatory Content Access Control (MCAC), a distributed information flow control mechanism to enable a content provider to control which network nodes can cache its contents. In MCAC, a CP defines different security labels for different contents, and content routers check these labels to decide if a content object should be cached. To ensure correct enforcement of MCAC, we also propose a design of a trusted architecture by extending existing mainstream router architectures. We evaluate the performance of MCAC in the NS-3 simulator. The simulation results show that enforcing MCAC in routers does not introduce significant overhead in content forwarding.

Index Terms—Access control, privacy protection, information centric networks

1 Introduction

Information Centric Networks (ICNs) have been recently proposed to explore clean slate approaches for future Internet architectures [19], [21], which aim to provide efficient content distribution with in-network caching mechanisms. At the same time, security and privacy have been considered as essential requirements for these new network architectures. For example, in Content-Centric Network (CCN) [19], each data packet is digitally signed so that content receivers can verify its authenticity and integrity. Also, each content request packet does not encode source and destination addresses but only content names, which enhances the privacy of content receivers. However, we find that current ICNs still lack some important security features for controlled content distribution in large scale. For example, CCN enables content caching in any router along a forwarding path to improve content delivery performance. Such unrestricted content caching mechanism can reveal user and content privacy easily. Many content providers (CPs) have policies to specify which contents can or cannot be cached in different networks. For example, HIPAA privacy policy requires that healthcare data cannot be cached by any entity during data delivery [7]. Therefore it is very desirable to have a lightweight and scalable mechanism to enable CPs to control which routers and Internet Service Providers (ISPs) can cache which contents in their network.

Information flow control has been widely applied in traditional operating systems (OSes) for access control and data protection [26], [34], [39]. Applying information flow control in ICN routers could potentially be a lightweight approach to protecting privacy. Specially, ICN routers normally have small and monolithic operating systems, which makes it easy to enforce information flow control in router OSes. However, the traditional information flow control models [26], [34], [39] primarily control information flows between different processes with labels. They provide fine-grained control to information within a relatively isolated OS. The role that ICN router operating systems play is not to process the information generated in the local system, but rather deliver the information (i.e., content) between different network nodes (i.e., ISPs/routers) in a large scale network, e.g., the Internet. Traditional information flow control models may be restrictive and unable to control information propagation within ICNs. Thus, a new information flow control model is required to provide fine-grained information flow control applied to contents within ICNs so as to enhance privacy protection.

Traditionally, to enforce information flow control in distributed systems, public key cryptography is applied to establish trust between information generators and receivers [39]. However, this approach is unsuitable for ICNs because public key cryptographic operations during runtime packet forwarding are heavyweight and introduce significant overhead in content forwarding [18]. To address this issue, Zeldovich

- *Q. Li is with the Graduate School at Shenzhen, Tsinghua University, Shenzhen 518055, China, and the Institute for Cyber Security, The University of Texas at San Antonio, San Antonio, TX 78249. E-mail: qi.li@sz.tsinghua.edu.cn.*
- *R. Sandhu is with the Institute for Cyber Security, The University of Texas at San Antonio, San Antonio, TX 78249. E-mail: ravi.sandhu@utsa.edu.*
- *X. Zhang is with the Samsung Research Center, Santa Clara, CA. E-mail: xinwenzhang@gmail.com.*
- *M. Xu is with the Department of Computer Science, Tsinghua University, Beijing 100084, China, and Tsinghua National Laboratory for Information Science and Technology, Beijing 100084, China. E-mail: xmw@cernet.edu.cn.*

Manuscript received 13 Oct. 2014; revised 22 Aug. 2015; accepted 11 Sept. 2015. Date of publication 26 Oct. 2015; date of current version 1 Sept. 2017.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TDSC.2015.2494049

et al. [39] propose to store secure label binding between information producers and receivers so as to avoid frequent public key cryptographic operations. Unfortunately, this proposal cannot be adopted in ICNs since information flow control in ICN will be enforced not only by information producers and receivers but also by information forwarders, e.g., routers delivering content in ISPs. In particular, it is very difficult to achieve a global consensus among CPs and ISPs on label binding to enforce the information flow control policy on the contents. Therefore, it remains challenging to enforce information flow control models in ICN networks.

In this paper, we propose MCAC—Mandatory Content Access Control, which is a generic security model to enforce information flow control in ICN. The goal of MCAC is to provide privacy protection for different components in ICNs, i.e., users, contents and content caches, by realizing restrictive content caching via information flow control. Different from previous work, MCAC allows CPs to attach labels to contents that are delivered within the whole ICN, and specify flexible content caching policies for different contents at the granularity of routers, ISPs, and autonomous systems (ASes). Moreover, we explore the possibility to enforce MCAC in existing mainstream routers. We propose a MCAC router design by extending the existing mainstream router architecture with the help of hardware-rooted trust. The proposed MCAC router architecture attests enforcement of information flow control in neighbor routers and thus builds trust between neighbors to ensure enforcement of information flow control, which lays out the foundation for privacy protection in ICN. With hardware-rooted trust, MCAC enforcement in a router avoids heavyweight cryptographic operations during content forwarding. Each component in a router is allowed to process a content only if it is assigned with a higher security level. This is lightweight and similar to the MAC address filter mechanism [5] implemented in current router architectures. We evaluate the performance of our design by simulations. The results show that enforcing MCAC in routers does not introduce significant overhead.

The rest of the paper is organized as follows. We briefly review the basic service model and discuss the security challenges in Section 2. The MCAC model is presented in Section 3. We present a design of MCAC router architecture consistent with existing mainstream router architectures in Section 4. Section 5 provides a security analysis, while Section 6 presents performance evaluation results. We present related work in Section 8 and conclude the paper in Section 9.

2 ICN and Privacy Threats

2.1 An Abstract ICN Model

An ICN normally has three types of entities: *content providers* who produce and own contents, *content consumers* who request and "consume" contents, and content routers operated by *Internet Service Providers* that connect CPs and content customers acting as content delegatees to disseminate contents from the CPs to the consumers. Basically, ICN has two types of packets, *content requests* and *contents*. Content consumers ask for contents by sending out *content requests*, and *content* packets are transmitted to content consumers in response to the content requests [19], [23].

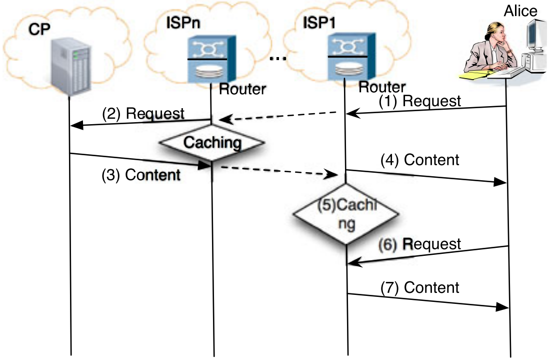


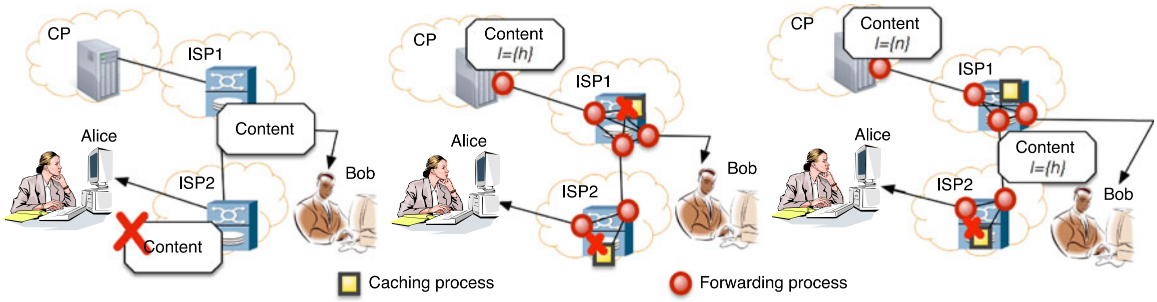
Fig. 1. The basic service model of ICN.

There are two categories of ICN designs. The first design is to develop a "pull" model to retrieve contents [19]. The content consumers, i.e., users, send out requests to pull contents that they want and CPs instantly respond with corresponding contents to them. The second design uses a "push" model in which contents are published into networks with registration services and then users can subscribe content requests to the registration services to retrieve the contents [21], [23]. In the "push" mode, CPs or ISPs push requested contents to users according to the request records in the registration services. In this paper, we illustrate the privacy threat model and our solution with the "pull" model. However our approach can be applied to the "push" model as well.

Two fundamental differences of ICN beyond traditional IP-based Internet are content-based content requesting and routing, and universal caching in content forwarding. Fig. 1 shows an abstract service model of a content request in ICN. Firstly, a content consumer Alice sends a content *request packet* with content name to retrieve a content in the network (step 1). Routers in ISPs forward the request to a CP based on routing protocols in ICN (step 2). Upon receiving the request, the CP responds to the router in the last hop ISP, i.e., ISPn, with a *content packet* corresponding to the content request packet (step 3), which forwards it backward to downstream routers along the routing path to be finally returned to Alice (step 4). In order to do this forwarding, a *forwarding table* is used in each ICN router to record which content request packet from which requester is not fulfilled yet. A request record in ICN specifies the port at which content is sent or received in a router. During the content packet forwarding, all routers along the forwarding path cache the content in their local storage (step 5), such that one router does not need to send out the content request to the CP again when receiving future requests with the same content name (see step 6 and 7). Note that an ISP may have a collection of routers to forward and cache contents.

To realize these functionalities, each ICN router has two primitive functions internally: *content forwarding* that is performed by forwarding processes in the router, and *content caching* that is performed by a caching process. In general, one router has multiple forwarding processes to communicate with different neighbor routers.

Compared to the existing IP network architecture, ICN has two distinct security features. (i) An ICN router directly uses content names to request contents and content request packets do not have address notations, i.e., packet source and



(a) ISP1 and ISP2 can cache when they wanted in ICN.

(b) Content is forwarded with label No ISP can cache the content.

(c) Content is forwarded with label *n*. Only ISP1 can cache the content.

Fig. 2. A two-ISP topology where different ISPs can have different permissions to cache contents.

destination information. Thus, ICN packets do not reveal CPs and content consumers. (ii) Each content data packet is digitally signed such that a content consumer can verify the integrity and authenticity of the content, no matter that the content is retrieved from its original CP or from an ICN router.

2.2 Threat Model for Privacy in ICN

Privacy has become a major concern in our digital life nowadays. From the technology point of view, content privacy preserving is tightly interrelated with content distribution and dissemination control. In our research, we focus on the analysis of what mechanisms are desired for privacy protection in ICN designs.

- *Content privacy:* Content providers lose control of contents after the contents are sent out to the networks. That is, a CP cannot prevent content caching by ISPs once the content has been delivered to one content consumer. For example, as shown in Fig. 2a, after the CP sends out content to ISP1, CP may not allow ISP1 to cache and track the content. However, CP cannot prevent these behaviors performed by ISP1. Also, ISPs can easily track content usage by investigating information encoded in the meta-data of the contents and content requests [15].
- *Cache privacy:* ISPs can easily retrieve cached content from their neighbors without any restrictions, and they can further cache the retrieved content. Assume the CP in Fig. 2a is willing to let ISP1 cache its content. However, in current ICN designs, the CP cannot control the disseminating of the cached content by ISP1. Actually, ISP2 can arbitrarily retrieve and cache the content from routers in ISP1, and further distribute them to other ISPs without the CP's approval.
- *User privacy:* ISPs can profile their customers (i.e., users or their customer ISPs) by monitoring their content access histories [15]. Then, ISPs may easily know what their customers request. As shown in Fig. 2a, Alice sends out her content requests to ISP2 and ISP2 can record and obtain Alice's private information by tracking the complete request list that Alice generated. ICN should provide anonymity, e.g., sender-receiver unlinkability [15].

2.3 End-to-End Encryption is Not Enough

End-to-end application layer content encryption has been widely used for content confidentiality and helps privacy protection [3], [15], [35]. However, several issues are not well addressed with these approaches.

- *Anonymity:* For confidential content packets, ICN needs to provide an anonymity property, i.e., sender-receiver unlinkability [18]. However, pure end-to-end content encryption cannot achieve this property [18]. Moreover, many content providers needs to have policies to specify contents can or cannot be cached in different networks, e.g., HIPAA privacy policy requires that healthcare data cannot be cached by any entity during data delivery [7].
- *Usability:* ICN aims to improve content forwarding performance by providing caching mechanisms in the network. However, end-to-end encryption impairs the effectiveness of content caching because the cached contents can only be decrypted by some specific nodes and content consumers, and cache retrieval by other nodes will be in vain. For instance, in Fig. 2a, we assume that Alice sends out the content requests to the CP and the contents delivered between them are encrypted with a shared key between Alice and the CP. ISP1 caches the encrypted contents when the contents are forwarded. If Bob is asking for the same contents from the CP, ISP1 directly responds to him with the cached content. It is obvious that the contents obtained by Bob cannot be read by him because the contents can only be decrypted by Alice. Although group encryption [36] could be leveraged to address this issue, it will introduce significant key management overhead. For example, normally, it is not easy for CPs to manage the keys for their users who may be always ready to change their content subscriptions. Therefore, usability with end-to-end content encryption is poor.
- *Performance.* End-to-end content encryption, e.g., group encryption [36], will incur significant overheads in processing and forwarding the contents. Normally, it can be used to prevent the contents that are highly confidential and used by delay-tolerant applications. In particular, it is not desirable to encrypt VoIP or real-time multimedia streaming packets.

Summary. As aforementioned, end-to-end content encryption does not suitably address the issues of anonymity, usability, and performance. In particular, end-to-end encryption cannot ensure the anonymity property of sender-receiver unlinkability. Actually, ICN shifts the semantics of the application layer protocols into the network layer protocols. Therefore, it is consistent with the spirit of ICN to leverage network layer privacy protections to address the issues in the network layer protocols of ICN. We can enable network layer privacy protection by designing and implementing distributed information flow control mechanisms [26], [34], [39] in ICN.

To respond to the threat model and the privacy protection challenges, this paper provides a generic security architecture to enable network layer privacy protection in ICN designs.

3 MCAC: Mandatory Content Access Control

Our idea for content distribution and dissemination control towards privacy protection mimics the information-flow based security mechanism in many operating systems. Specifically, we introduce the Mandatory Content Access Control model for this purpose in ICN. For simplicity, we assume one ISP only has one router in the rest of the paper.

3.1 The Basic MCAC Model

Traditional Mandatory Access Control (MAC) models [26], [34], [39] provide confidentiality by enforcing the simple security property.

Simple security property. Given a subject s at a given security level s_l and an object o at a security level o_l, where s_l has a lower security level than o_l, i.e., $s_l < o_l$, the subject s cannot read the object o.[1]

The security level information is normally valid within one system and is not universal within a large scale distributed system, e.g., the Internet. Thus, the above property only works for an isolated system but is typically unable to control information flows within a network. Hence, it may not be directly applied to secure different objects (i.e., content requests and content data packets) in ICNs where CPs want to control object accessing and caching in ISPs.

To address this issue, we assign labels to subjects (processes in content routers) and objects (content requests and contents) in ICN, such that we can determine if operations on objects are allowed in a router, e.g., contents are allowed to be cached, by comparing the labels. Basically, we can classify objects into four levels $\{h, n, d, p\}$, where h denotes the highest protection level, n denotes the non-delegatability level, d denotes the delegatability level, and p denotes the publicly accessing level (see Fig. 3). Objects are assigned labels as follows.

- Objects that should be highly secured by CPs and cannot be cached by any router in any ISP are at the level of h, which is used to realize *non-caching* policy.
- Objects that can be cached only once by caching processes in routers of one ISP on one object forwarding path are at the level of n, which is used to realize *1-level caching* policy.

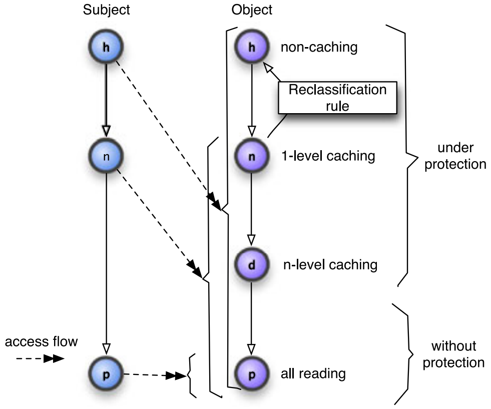


Fig. 3. Information flow controlled by the MCAC model.

- Objects that can be cached by caching processes in routers of more than one authorized ISPs in the network are at the level of d, which is used to realize *n-level caching* policy.
- Objects that can be cached and read by any processes in routers of any ISP in the network are at the level of p, which is used to realize *all-reading* policy. The objects with this security level are publicly available, e.g., the packets for the purpose of the network diagnostic.

Note that, objects at d security label allow ubiquitous object caching. Such objects should not be accessible by arbitrary processes in the control plane of the nodes (see Section 4). However, objects in router OS are not only content requests and contents. For example, routing control messages can be another type of objects. These objects can be read by any network node, including arbitrary processes in the control plane, by setting their labels to p. In general, in order to enable arbitrary object access in MCAC for objects that are not ICN specific, we apply the p security level to such objects.

According to the ICN designs, CPs cannot know who are exact content consumers and what are content forwarding paths that the routers will use; that is, they do not know which remote ISPs will deliver the contents produced by them. Therefore, what CPs can do is only to control content caching in their first hop ISPs. If an ISP wants to build collaborations with a CP and cache contents produced by the CP, they need to build direct peer links with it, which is similar to the inter-domain routing operations in the current Internet [37].

The subjects, i.e., processes, in a router OS, can be classified into two categories.

- *Sensitive processes:* There are only two processes in this category, i.e., the content forwarding process and the content caching process. These processes take charge in processing content requests and contents in ICN. The content forwarding process forwards (*i.e., a type of reading operations*) content requests and contents delivered within the network in a router OS. Since the processes should have permissions to read all objects to disseminate them, its security level should be set to h. The content caching process is used to cache contents after reading these

[1] We assume the security levels are totally ordered, which suffices for our purpose in this paper.

objects. The caching process is allowed to read only cacheable objects, so its security level is set to n.

- *Non-sensitive processes:* Other processes in a router OS are in this category. All processes in this category are with the label of p.

Therefore, as shown in Fig. 3, subjects with h security label can access all objects with different security labels, and subjects with n security label can access objects with n, d, and p security labels. However, subjects with p security label can only access objects with p security labels. Here, d security label is only used to label objects but not subjects.

To summarize our MCAC model consists of the following components in a router OS.

- S denotes the set of subjects (i.e., sensitive and non-sensitive processes).
- O denotes the set of objects (i.e., contents and content requests).
- $\mathcal{L} = \{h, n, d, p\}$ are labels that define protection classes. These labels form a lattice under a partial relationship "$>$" where $h > n > d > p$.
- A function $init$: $S \leftarrow \mathcal{L}$ assigns a security label to each subject, which is normally performed by ICN router vendors.
- A function $assign$: $O \leftarrow \mathcal{L}$ assigns a security label to each object.

MCAC policies. Different from traditional information flow control models which need to control both read and write operations, the processes in ICN router only have the *read* operation. To secure information flows in router OS, we need to control the flows interacted with sensitive processes, i.e., *the content forwarding process* and *the content caching process*. The content forwarding process is labeled with h, which indicates that it can read all objects with different security levels. The content caching process is labeled with n, which means it can only read the object with the labels lower than h. However, the non-sensitive processes are only allowed to read objects with label p. With these settings, the MCAC model has the following policy rules.

- *Object reading rule:* An object o can be read by the content forwarding process in a router under any security level $l \in \mathcal{L}$.
- *Object caching rule:* An object o can be read by the content caching process in a router only when its security level $l < h$ (see Fig. 3).

We explain some example content cache controls following the sample network topology in Fig. 2a. Before serving any content request packet, the CP has set the security level of its content to h (see Fig. 2b). Note that, a router normally has many linecards connecting with different neighbors. Each linecard has a forwarding process for forwarding contents. All content forwarding processes in ISP1 and ISP2 have permissions to read and forward the content to the forwarding processes of their neighbors because the processes have the same security level with the content (see Object Reading Rule). However, the flow from the forwarding process to the caching process is not allowed since the cache process has lower security level than the content label (see Object Caching Rule). Therefore, the content cannot be cached in the router. Finally, the content is delivered to Alice and Bob. Similarly, n-level caching and all-reading policies can be enforced by specifying corresponding labels for contents, respectively. For example, if a content label is set to p, the content can be cached by any process and routers in any ISP, which enables all reading policy. Note that the policies are defined by CPs according to their contracts with ISPs. The ISPs only need to perform the correct operations based on labels. The operations are similar to packet forwarding according to IP prefixes in the current Internet. In Section 4, we will address MCAC enforcement in routers such that CPs can ensure that the contracting ISPs forward contents according to the labels. Therefore, CPs do not need to reach consensus with ISPs and other CPs about policy configuration.

Secure content request delivery in MCAC is similar to content delivery procedures. Since content requests can only be read by both content forwarding and caching processes, the label of the content requests is set to d. Thus, the sensitive processes in routers are allowed to cache content requests such that routers can respond to the requesters after obtaining the requested contents (see Fig. 1). For simplicity, content requests and contents are collectively named as *contents* in the rest of the paper.

3.2 Reclassification Rule

The basic MCAC model only implements non-caching, n-level caching, and all-reading policies. In order to realize the 1-level caching policy, we introduce a content reclassification mechanism for ICN. Specifically, by following a CP's security policy, a content object can be reclassified accordingly by a content router to support 1-level caching policy, while the reclassification operation is controlled by the CP.

- *Reclassification rule:* An object o can be re-labeled to h, i.e., $assign(o, h)$, if its original security level $l = n$.

Basically, the reclassification rule is to upgrade the levels of content labels in ISPs, which is controlled by CPs. As shown in Fig. 3, reclassification ensures the security levels of the contents are upgraded if the contents with label n are delivered by the first hop ISP. Thus, the contents cannot be arbitrarily cached during content delivery in different ISPs. The object with label n is reclassified to label h such that the object can be read by the content caching process by only one authorized ISP in the content forwarding path. This rule ensures that the CP can authorize the first hop ISPs to cache contents and other are not allowed. Note that, since contents are cached with the corresponding labels in the authorized ISP, the reclassification rule is applied to cached contents (from ISPs) in ICN.

We explain how the reclassification rule supports 1-level caching policy to protect content privacy. As shown in Fig. 2c, ISP1 upgrades the security level of the content to h before further delivering it to ISP2 according to the reclassification rule, no matter whether the content is obtained from ISP1's caching service or directly from the CP. With this new label, routers in ISP2 cannot cache the content. In this setting, ISP1 can directly respond to Bob's request with cached content. However, ISP2 cannot directly provide the content for Alice, but has to forward after retrieving the content from ISP1.

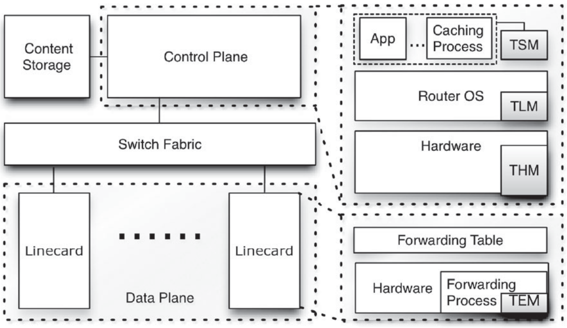


Fig. 4. MCAC router architecture.

4 Enforcing MCAC in ICN Routers

In this section, we present a MCAC router architecture that enforces MCAC polices in a lightweight manner. In the next section we analyze the security properties achieved with this architecture.

4.1 MCAC Router Architecture

Enforcing MCAC policies is done by individual content routers in ICN. Existing routers in the Internet consist of two elements, control plane and data plane, which are connected by a switch fabric (see the left part of Fig. 4). The forwarding process in data plane delivers data packets with fully distributed processors according to the routes computed in control plane. The control plane in the router OS does not process normal packets received from neighbors, except route update information. To implement ICN design in this mainstream router architecture, the control plane should be extended to make forwarding decision upon each content request and provide storage to realize content caching.

Normally, routers have small and monolithic operating systems. Router OSes are generally closed system and not general-purpose, and normally shipped with router hardware by vendors. However, it is still challenging to enforce MCAC in routers in a trusted manner. In particular, ISPs have different routing instances with variant implementation versions in their networks. To address this issue, we realize a small trust computing base (TCB) in a router to ensure the enforcement of information flow control by establishing hardware-rooted trust between routers. The hardware-rooted trust is built by attesting the integrity of the modules in the ICN router architecture with the help of Trusted Hardware Module (THM) [25], [40], e.g., Secret Protection (SP) [17] and Trusted Platform Module (TPM) [8]. Fig. 4 shows our MCAC router architecture which has three major modules.

- *Trusted Storage Module (TSM)* negotiates secret keys with TSM on neighbor routers, and writes the secret keys into TEM linecards for content encryption. The secret keys are used to encrypt highly confidential contents, e.g., contents with label h, delivered to neighbors, which prevent against content monitoring in networks. The content caching process handles caching contents sent from TSM and encrypting/decrypting cached contents with the keys sealed in THM.
- *Trusted Labeling Modules (TLM)* is responsible for enforcing the MCAC model for different contents, which is determined by content labels encoded in data packets by CPs. Basically, TLM checks if each read operation is permitted by reading the content label. For instance, if a router is allowed to cache content, the TLM allows the OS to deliver the content to TSM. Otherwise, the content is dropped by TLM.
- *Trusted Enforcement Modules (TEM)* in linecards are responsible for read operations in data planes to achieve privacy protection. Since content requests are recorded by the content forwarding process in TEM, content delivery can be directly performed in the content forwarding process according to the content request records in linecards. Also, TEM reclassifies outgoing contents according to possible reclassification rule and encrypts the contents to the neighbor routers with session keys negotiated by TSM. These operations can be performed by hardware such that contents are forwarded at line speed.

Note that THM is used to protect the integrity of the other trusted modules' executions with attestation mechanism [8], [17], which ensures that all modules will correctly perform their operations. For example, if TSM or TLM in the control plane is tampered, e.g., TLM is compromised and modifies the reclassification rule, the secure channels between this router and its neighbor routers cannot be successfully built and then the router cannot obtain any content from the neighbor routers. The mechanism effectively defends against man-in-the-middle attacks launched by malicious routers. Similarly, if a TEM module in the linecard is compromised, i.e., TEM is modified to inspect content packets, the secure channel between TSM and TEM cannot be built. Then, the control plane cannot select the linecard to propagate content requests, and contents cannot be distributed to the linecard. As aforementioned, TEM can be implemented with dedicated hardware such that contents can be processed in line speed in linecards. For simplicity, we can trust the hardware in linecards so that THM may not need to attest TEM in practice. Note that, since THM is widely deployed in different generic and embedded system platforms [27], this architecture can be easily extended and implemented in these platforms, which ensures that different content forwarders and consumers can process the contents according to the MCAC policies defined by CPs.

4.2 Establishing Trusted Router Communications

In this section, we use examples to illustrate how MCAC routers build trust and communicate with each other to enforce MCAC policies. Fig. 5 shows the procedures of router communications. When routers are bootstrapping, TSM will be invoked during its bootstrap phase after the integrity of TSM, TLM, and TEM are validated by the trusted components on the platform built upon THM (steps 1 and 2). All keys in each router will unseal from THM to TSM after successful attestation. The keys are used to negotiate with neighbor routers to compute session key S (steps 3 to 6). Here, for simplicity, we only

	CP		**Router 1**		**Router 2**
// Stage (I): Bootstrapping and initialization					
1.			module attestation		module attestation
2.			TSM retrieves k_1 from THM		TSM retrieves k_2 from THM
3.			$X = g^{k_1} mod\ p$		
4.				$\xrightarrow{X}$	$Y = g^{k_2} mod\ p$
5.				$\xleftarrow{Y}$	
6.			$S = Y^{k_1} mod\ p$		$S = X^{k_1} mod\ p$
7.			S delivered to TEM		S delivered to TEM
// Stage (II): Router Communications					
// Case (1): CP generates contents with label l, where l = h					
8.		$\xrightarrow{C_l}$	CHECK(C_l) by TLM		
9.			$E_C \leftarrow$ ENC$_S(C_l)$ by TEM	$\xrightarrow{E_C}$	$C_l \leftarrow$DEC$_S(E_C)$ by TEM
10.					CHECK(C_l) by TLM
// Case (2): CP generates contents with label l, where l = n					
11.		$\xrightarrow{C_l}$	CHECK(C_l) by TLM		
12.			C_l delivered to TSM		
13.			RECLASS(C_l) by TEM		
14.			$E_C \leftarrow$ ENC$_S(C_l)$ by TEM	$\xrightarrow{E_C}$	$C_l \leftarrow$DEC$_S(E_C)$ by TEM
15.					CHECK(C_l) by TLM
// Case (3): CP generates contents with label l, where l = d					
16.		$\xrightarrow{C_l}$	CHECK(C_l) by TLM		
17.			C_l delivered to TSM		
18.				$\xrightarrow{C_l}$	CHECK(C_l) by TLM
19.					C_l delivered to TSM
// Case (4): CP generates contents with label l, where l = p					
20.		$\xrightarrow{C_l}$	CHECK(C_l) by TLM		
21.			C_l delivered to TSM		
22.			C_l delivered to other Apps		
23.				$\xrightarrow{C_l}$	CHECK(C_l) by TLM
24.					C_l delivered to TSM
25.					C_l delivered to other Apps

Fig. 5. The MCAC router communication procedures.

show use a basic Diffie-Hellman key exchange protocol [16] to negotiate session keys. In real practice, to prevent man-in-the-middle attacks, routers can use an authenticated key agreement protocol to negotiate and refresh session keys. In a later section, we will present a complete authenticated key agreement protocol and analyze the correctness of the protocol. Note that keys are sealed by router vendors and can be authenticated by peers, which is similar to the authenticated key agreement protocols [24]. If module attestation fails, e.g., the modules are tampered by adversaries, routers cannot get any keys from THM to negotiate with neighbors. Moreover, any malicious routers cannot have correct keys to communicate with benign routers to retrieve contents. After session key S is negotiated, it will be delivered to the corresponding TEM (step 7).

We now discuss how MCAC routers can communicate each other to deliver contents. As shown in case (1) in Fig. 5, a content with label h is sent to router 1. After router 1 gets the content from CP, TEM in linecards will check the label embedded in the content. Since the content is with label h, it will not be delivered to TSM but directly forwarded to router 2 by linecards. Before the content is sent out, the content will be encrypted using the negotiated session key S. Similarly, router 2 will firstly decrypt the received content and directly forward the content. The downstream routers will have a similar procedure to process the content. In case (2), where the content is sent out with label n, the content is allowed to be delivered to TSM in router 1. However, TLM in router 1 will reclassify the content and change the content label from n to h, which indicates that all downstream routers cannot access or cache the content but only forward it. In case (3), the content is with label d, which means that the content can be accessed by TSM in all routers. In the meanwhile, the content will be forwarded further. It can be accessed by TSM in the downstream routers as well. It will not be encrypted during forwarding. Case (4) is similar to case (3). In case (4), the content with label p can be accessed by all applications in the routers. Note that, if contents from CPs with labels h and n are encrypted, the communications between CPs and the first hop routers are similar to that between router 1 and router 2. For simplicity, we do not illustrate this in Fig. 5.

5 Security Analysis

In this section we analyze the security properties of the secure communication protocols using compositional logic [13]. Based on the privacy threats in Section 2 and the security properties of the secure communication protocols in MCAC, we analyze whether the MCAC router architecture is secure with respect to these threats.

5.1 Correctness of the Communication Protocol

In this section, we analyze the security of the communication protocol in MCAC. The secure communication protocol presented in Section 4.1, built upon the basic Diffie-Hellman key exchange protocol (shown in Fig. 5) can be translated into the following cord calculus [13]. Here, the cord calculus is illustrated with the arrows to show the messages delivered between two agents, i.e., R1 and R2.

$$\begin{array}{lllll} \texttt{R1} = [(vx) & < \hat{X}, \hat{Y}, g^x > & & & (\hat{Y}, \hat{X}, y, z)] \\ & \Downarrow & & & \Uparrow \\ \texttt{R2} = [& (\hat{X}, \hat{Y}, x) & (vy) & & < \hat{Y}, \hat{X}, g^y >], \end{array}$$

where (vx), $< x >$, and (x) refer to the actions of nonce generation, sending a message, and receiving a message [41].

According to the results in [13], the postcondition *ActioninOrder(Send(R1,* msg1*), Receive(R1),* msg1*), Send(R2),* msg2*), Receive(R2),* msg2*))* cannot hold, which means that the actions of principals $R1$ and $R1$ may not occur in a certain order. If principals $R1$ and $R2$ can be sure of this order, then $R1$ is assured that $R2$ saw certain actions occur in a certain order and vice versa [13]. With this protocol, two principals cannot be in the form of matching conversations due to man-in-the-middle attacks and thus the authentication property cannot hold. Hence, this protocol above only provides the key secrecy property, but cannot prevent man-in-the-middle attacks. Therefore, it is not secure. In order to address this issue and achieve the mutual authentication property, we extend the protocol to an authenticated secure protocol with signatures of nonce and the peer identity as follows:

$\texttt{R1} \rightarrow \texttt{R2:}\ \texttt{g}^{k_1}$
$\texttt{R2} \rightarrow \texttt{R1:}\ \texttt{g}^{k_2}\texttt{,}\ SIG_{R_2}(g^{k_1}, g^{k_2}, R_1)$
$\texttt{R1} \rightarrow \texttt{R2:}\ SIG_{R_1}(g^{k_1}, g^{k_2}, R_2).$

The extended protocol can be represented as follows using the cord calculus:

$$\begin{array}{l} \texttt{R1} = [(vx) \qquad < \hat{X}, \hat{Y}, g^x > \qquad (\hat{Y}, \hat{X}, y, z) \\ \qquad (z/\{|g^x, y, \hat{X}|\}_{\overline{Y}}) < X, Y\{|g^x, y, \hat{Y}|\}_{\overline{X}} >] \\ \texttt{R2} = [\qquad\qquad (\hat{X}, \hat{Y}, x) \qquad\qquad (vy) \\ \qquad < \hat{Y}, \hat{X}, g^y, \{|x, g^y, \hat{X}|\}_{\overline{Y}} > \qquad (\hat{X}, \hat{Y}, z) \\ \qquad (z/\{|x, g^y, \hat{Y}|\}_{\overline{X}})], \end{array}$$

where (vx), $< x >$, (x), $\{|x|\}_{\overline{X}}$ refer to the actions of nonce generation, sending a message, receiving a message, and the signature by key named with $\overline{X}$ [41].

With the cords above, according to the results in [41], *ActioninOrder(Send(R1,* msg1*), Receive(R1),* msg1*), Send(R2),* msg2*), Receive(R2),* msg2*))* holds, and R1 and R2 have matching conversations. Hence, the protocol is secure and correct. We can conclude that the communication protocol with an authenticated protocol is secure. Therefore, in MCAC any routers can communicate with each other to compute an authenticated shared secret for secure communications with each other. Actually, the analysis can always be applied to the protocol to achieve similar security goals if the protocol is adapted from other secure communication protocols, e.g., IKE [22], JFKi [9], and JFKr [9].

5.2 Security Properties of the Proposed Architecture

In the MCAC architecture, content requests and content packets are propagated with proper labels such that only processes with the appropriate labels in routers can handle them. Moreover, the communications between two routers are secured by encryption. Thus, content requests and content packets will not be sniffed and tracked by other entities and effectively prevent man-in-the-middle attacks in the network. Therefore, content privacy is ensured in the MCAC architecture. We have the following proposition.

Observation 1. The MCAC architecture ensures content privacy protection in ICN.

In the MCAC architecture, caching contents is only allowed by the authorized routers, which is ensured by setting the proper labels in the contents by CPs. Any malicious routers cannot communicate with legitimate routers to obtain content caches because they cannot get any valid keys to establish sessions with the legitimate routers. Therefore, caching contents is restrictively controlled by CPs, and content privacy is secured accordingly.

Observation 2. The MCAC architecture ensures cache privacy protection in ICN.

Since content requests and content packets are only accessed by authorized processes with the appropriate labels, other processes in routers cannot access and record the information. Furthermore, the information also cannot be tracked by other entities because the communication channels between two routers are encrypted. In particular, any unauthorized entities, including unauthorized routers, are unable to track content information so that they cannot infer connections between content senders and receivers [15]. Hence, user privacy is protected with the MCAC architecture.

Observation 3. The MCAC architecture ensures user privacy protection in ICN.

The above propositions establish the security properties of privacy protections achieved in the MCAC architecture. Therefore, the MCAC architecture achieves the desired property of privacy protections.

6 Evaluation

To quantify the overheads imposed by the MCAC architecture, we measure the costs of enforcing MCAC. Since hardware-based ICN routers are not publicly available yet, we do not fully implement our scheme but simulate it based on a software-based prototype to evaluate the runtime overhead. It is obvious that router attestation introduces extra delays during router bootstrapping. Our previous study shows that router attestation incurs around 30 percent delays [27]. Since it is only a one-time operation during router bootstrapping, this will not impact runtime content forwarding, so the overhead is acceptable. In this section, we will focus on evaluating the overhead incurred by enforcing the MCAC policies during packet forwarding.

We extend the NS3-based ICN simulator, i.e., ndnSIM [6], with the MCAC functionalities and evaluate the incurred content forwarding overhead. In the simulator, each content request or content packet contains a label indicating its security level. The packet labels are randomly assigned. We extend different classes in ndnSIM so that these classes include a security label. All labels are assigned according to the policies given in Section 3. We modify the APIs in different classes, e.g., the ContentStore and AppFace classes, to enforce the *object reading and caching* rules. Meanwhile, the ForwardingStrategy class is extended to enforce the *object reclassification* rule. In simulations, when a content request or a content packet is received, the packet forwarding is same as the default Named Data Networking (NDN) protocol. However, when other components want to access the packet, they are authorized only when the access complies with MCAC policies. A lower security level process by default cannot perform any operation on packets that have a higher security level. We run the simulations using different Rocketfuel topologies adopted in ndnSIM [6]. We randomly select leaf nodes in the topologies as CPs and content consumers, and randomly assign packets generate by CPs with different labels. We do not observe any overhead during packet forwarding in simulations, even for the contents that are reclassified during their forwarding. This result is reasonable because the packet processing delays introduced by enforcing MCAC policies, e.g., labeling packets, are negligible compared to the packet forwarding delays in NDN [38].

Since the MCAC architecture adopts content encryption for contents with label h or n, we also evaluate the resulting packet forwarding delays. We assume that the architecture uses AES in OFB mode to encrypt packets. We measure the computation overhead of the AES algorithm. Content encryption and decryption on average incur 3 millisecond each. The data is seeded into simulations as parameters. For simplicity, we only measure the content forwarding delays in one of the Rocketful topologies [6], i.e., the Sprint network topology in the simulation. We conduct three independent simulation runs with the seed. In each simulation run, we randomly pick two nodes as a content producer and a user, respectively. Fig. 6 shows the performance in the MCAC architecture with content encryption with respect to that without encryption. Content encryption introduces around 35 percent extra overhead in forwarding contents with label h and n. Since MCAC achieves similar security properties with ANDaNA [15], we only compare the performance of ANDaNA here. Compared to ANDaNA that doubles the content forwarding delay, the content forwarding delay in MCAC is much lower. In particular, such an overhead is only applied to the contents with high confidentiality, which is different from ANDaNA that encrypts all delivered contents.

7 Discussion

7.1 Trust Establishment among ISPs

Trust establishment among ISPs is performed upon successful router platforms attestation. A router can communicate with other routers only when TSM in the router can get keys from THM and use these keys to establish sessions with neighbors. Therefore, successful session establishment means that the router platform is successfully attested and the MCAC policies are correctly enforced. Note that, the MCAC policies only specify how to authorize packet access requests according to security levels of the requesters and packets. To enforce the policies, routers only need to compare different labels to authorize the requesters, which is similar to the MAC address filter mechanism [5] that filters packets according to the MAC addresses. Moreover, THMs for different platforms are available [8], [17], [27]. Thus, vendors can enable platform attestation in their routers and implement MCAC routers. Different vendors can have their own attestation implementation as long as the implementation complies with the MCAC design. Similarly, caching providers can also build trust with CPs and ISPs provided THM is installed in their systems. Note that, MCAC only attests routers and CPs' modules and does not require attesting their polices. Hence, it is easy for ISPs to establish a common trust base for secure content forwarding with other ISPs. Actually, the trust establishment with MCAC is similar to BGP session establishment between different ISPs in the current Internet, which only requires BGP routers complying with the BGP protocol specifications. Each ISP can still have its own routing policies in its BGP routers to set route preferences.

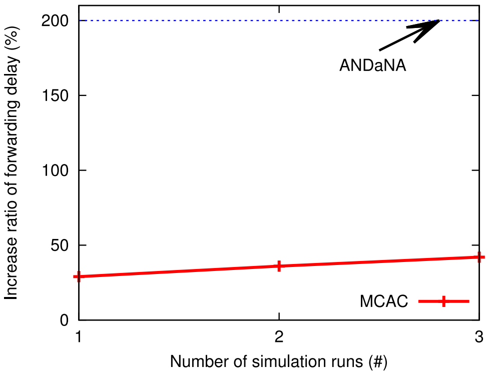


Fig. 6. The increased content forwarding overhead in NDN for the contents with labels h and n.

7.2 Runtime Integrity Protection

Normally THM can only ensure the correctness of the MCAC router architecture during bootstrapping. However, it cannot prevent runtime intrusion. For example, an adversary may compromise a TSM module after the router is bootstrapped so that the compromised router can still obtain keys to communicate with neighbor routers and retrieve contents. To address this issue and preserve a good runtime environment of a MCAC router, several runtime protection mechanisms can be applied, such as ARM's TrustZone [2], Intel's Trusted Execution Technology (TXT) [4] and AMD's Pacifica technology [1]. These mechanisms enable runtime measurement of a protected component at any time during operation of the component. Runtime measurement dynamically computes cryptographic hash over the component code before the component is executed [31]. This process reinitializes CPUs to an initial well-known state, and computes a cryptographic hash over the relevant code region before execution of the component. The attestation procedure is similar to that during the bootstrapping process. The computed hash value will be reported to THM. If the router is compromised, the computed hash value is different from what THM expects. TSM cannot get any keys from THM and the router cannot communicate with neighbors. Therefore, this measurement process ensures runtime integrity of the router architectures by enabling THM-based dynamic attestation, which is similar to the existing dynamic attestation mechanisms that are widely used to ensure the correct execution of security-critical software [10], [28], [29], [31], [32], [33]. In particular, compared with the execution environment in traditional routers, dynamic attestation in MCAC allows a MCAC router to construct a small isolated execution environment to run trusted components so that the router attestation will be performed efficiently [31], [32].

Moreover, to enable multi-dimensional aspects of content policies, e.g., content can be cached for a week only or for a time specified by the content provider, content providers can produce the same content with different security levels during different periods. However, there may be policy consistency issues among different intermediate nodes that cache the content because the cached content may be generated in different periods and with different security labels.

7.3 Security Policy Enforcement

In MCAC, the security level of content is defined by the providers that produce the content. The goal of the MCAC architecture is to correctly enforce information flow control on content according to the security level defined by the providers such that only authorized nodes can access the content. In this sense, whether the content is sensitive or not is completely defined by the content providers. The intermediate nodes only need to comply with these policies. Note that, the secure execution environment built in our scheme is similar to other Trusted Computing Bases (TCBs) built upon TPM [31], [32]. The only difference is that the secure execution environment in our scheme is built on hardware routers with different linecards. The distributed hardware router architecture includes an embedded operating system with multiple layers of protocols and millions of lines of code [31], [32]. Our scheme can be implemented with a tiny amount of code with the security functions enabled by the hardware.

7.4 Real Deployment on the Internet

In Section 3, we present and enforce the MCAC model and policies within a network where each ISP only has one router. In real practice, each ISP may have more than one router. We can easily address this issue by extending the implementation of the MCAC router architecture. In the extended implementation, if a router is communicating with neighbors within the same ISP, the reclassification rule will not be reactivated. However, routers that connect to routers in the neighbor ISPs will still enforce the rule. Fig. 7 shows a network topology extended from Fig. 2. R1 in ISP1 will not reclassify the content label if the content will forwarded to R2 but will reclassify the label when the content will delivered to ISP2. Thus, the content caching process in R2 can read the content but ISP2 still cannot read it. A similar mechanism is already enabled in the existing inter-domain routing protocols, i.e., Border Gateway Protocol (BGP) [37].

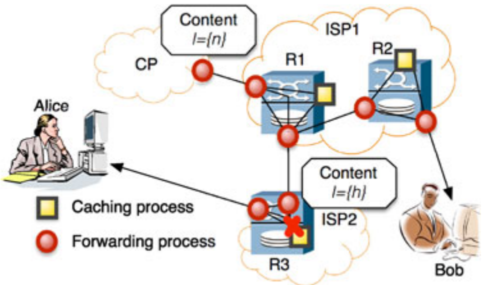


Fig. 7. MCAC deployment: an ISP has more than one ICN router.

In BGP, the attributes of route updates are modified only when route information will be announced to the neighbor ISPs over external BGP sessions [37]. Actually, our extension can work with BGP since existing ICN designs still use BGP to disseminate routing paths. The reclassification rule is only enforced in the routers connecting to neighbors with external BGP sessions. Otherwise, the rule will not be activated. Note that, similar to the basic MCAC router design (see Section 4), any adversaries cannot tamper the extended mechanism since the router architecture including the extension is attested by THM. Note that, MCAC routers do not change operations of the current Internet. In the current Internet, ISPs need to cooperate with each other to deliver packets as well. In MCAC, for different security level of content, ISP do not need to have special policies since MCAC routers can directly enforce the policies according to the security labels embedded in the content.

7.5 Incremental Deployment

MCAC is incrementally deployable. Since content labels can be piggybacked in content packets [19], routers that are not enabled with MCAC, i.e., non-MCAC routers, can still forward the packets to the next hops if the packets are delivered from MCAC routers. Normally the packets forwarded by non-MCAC routers are with the label of p. If the packets are sensitive, MCAC routers will not deliver the packets to non-MCAC routers to protect the packets. Two remote MCAC routers can attest each other and build secure sessions across multiple non-MCAC routers. In this setting, MCAC routers can deliver confidential content to each other but the intermediate routers cannot access the content because the content is encrypted with the secure sessions.

Moreover, MCAC routers can be directly deployed with existing network topologies, and will not impact them. Similarly, under incremental deployment, MCAC routers can have similar routing decisions as legacy routers except that the MCAC routers can have richer routing policies to enforce information flow control, e.g., confidential content will be only delivered within MCAC routers.

8 Related Work

Bell-LaPadula (BLP) model is the first confidentiality-based MAC model which is used to protect privacy within an isolated OS [11]. Many real OSes, such as Trusted Solaris and

SELinux, implemented the protection models similar to BLP. These models assume that the subjects and objects are either trusted or untrusted. However, in distributed systems, e.g., ICNs, subjects are diversified with different security levels, and objects may have different security requirements. Thus, these models may be unable to provide privacy protections for ICN. MCAC classifies subjects and objects in a network into different levels of security so as to enable fine-grained access control to different objects in ICN. Furthermore, our MCAC model uses four labels to enforce information flow control and realize restrictive content caching, requiring minimal configuration by object owners, i.e., CPs. Many integrity protection models, e.g., the Biba model [12], use a set of static security rules similar to BLP model, so as to ensure data integrity based on the fixed integrity levels. Usable mandatory integrity protection (UMIP) model [26] was proposed to maintain dynamic integrity levels for subjects and object, which are orthogonal to the MCAC model. MCAC can benefit from the integrity models to provide content integrity protection.

Jaeger et al. [20] leveraged IPSec to label network connections, which allows OSes in different machines to control the communications between themselves. This approach requires external mechanism to establish trust and define mapping between IPSec keys and labels. The leveraged IPSec approach is employed in Shamon [30] to enable Distributed Mandatory Access Control between two virtual machines across different physical machines. Different from Shamon, MCAC focuses on defining MAC policies for content packets delivered among multiple machines and enforcing information flow controls on these packets to prevent information leakage in the network. Zeldovich et al. [39] proposed a distributed information flow model to secure distributed systems by exporting labels between different machines, which is somewhat similar to MCAC. The approach translated labels between OSes and applications to secure data, which is not necessary for router OSes. MCAC allows CPs to assign labels to corresponding contents by defining content security polices such that information flow control is enforced in routers based on these labels.

Language-based information flow security was extensively studied in the literature [14], [34]. This line of work analyzed the information flow within programs, which is orthogonal to our work. MCAC provides information flow control within OSes. We can use the techniques in language-based information flow security to ensure correctness of process inputs in router OSes, to prevent privacy leakages caused by program bugs.

Recently, cryptography-based approaches were applied to privacy protection in ICNs [15], [35]. DiBenedetto et al. [15] applied onion routing in *poll* mode based ICN designs, which use multiple cryptographic operations to provide end-to-end content privacy protections. The approach is the only one that can provide the anonymity property but introduces significant overheads in content forwarding. Other approaches leverage pure end-to-end content encryption techniques. Nabeel and Bertino use Paillier homomorphic cryptographic system to secure different messages in the "push" mode based ICNs [35]. These approaches may impair the performance benefits in ICN. ICN designs improve the network performance using content caching mechanisms, but end-to-end content encryption makes content caching useless. Moreover, it may not be easy deploy these approaches in routers because they introduce significant overheads in processing and forwarding contents. Our MCAC model realizes privacy protections by restrictive content caching with distributed information flow control. The anonymity property is achieved in MCAC by preventing information leakage among subjects with different security levels across the ICNs. The proposed MCAC router architecture does not generate significant overheads in content caching and forwarding procedures, which is different from the cryptography-based approaches.

9 Conclusion

Proposed for future Internet architectures, Information Centric Networks have gained considerable attentions in research community with features of name-based content retrieval, in-network caching and security. To enhance the privacy protection for content providers in ICN, we propose a mandatory content access control model, which is a distributed information flow control approach for controlling a router's caching capability. An important aspect of our MCAC model is that contents are assigned with different labels according to security requirements of CPs. Labels form a linear security class order such that each CP can use its own separate annotations to control where contents can be read and cached with flexible constraints of ISPs, locations and time periods. To enforce MCAC policies in ICN, we present a design of MCAC router extended from the mainstream router architecture with the help of hardware-rooted trust.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 61572278 and 61133015, in part by the 973 Program of China under Grant 2012CB315803, in part by the State High-Tech Development Plan under Grant 2015AA015603, in part by the Air Force Office of Scientific Research, Arlington, VA, USA, through the Multidisciplinary University Research Initiative Programme under Grant FA9550-08-1-0265, and in part by a gift from Cisco.

References

[1] Advanced Micro Devices, AMD64 virtualization codenamed "pacifica" technology: Secure virtual machine architecture reference manual, May 2005.

[2] (2009). ARM security technology, Building a secure system using trustzone technology [Online]. Available: http://infocenter.arm.com/help/topic/com.arm.doc.prd29-genc-009492c/PRD29-GENC-009492C_trustzone_security_whitepaper.pdf

[3] (2010). CCNx access control specifications [Online]. Available: http://redmine.confine-project.eu/projects/ccnx/repository/revisions/2e4fa601aed28d4cd2b397b3aec5d48a3707185f/entry/doc/specs/AccessControl/AccessControlSpecs01.pdf

[4] (2015). Intel corporation, Intel trusted execution techonolgy [Online]. Available: http://www.intel.com/content/dam/www/public/us/en/documents/white-papers/trusted-execution-technology-security-paper.pdf

[5] (2014). MAC filtering [Online]. Available: http://en.wikipedia.org/wiki/MAC_filtering

[6] (2014). NS-3 based named data networking (NDN) simulator [Online]. Available: http://ndnsim.net.

[7] (2014). Summary of the HIPAA privacy rule [Online]. Available: http://www.hhs.gov/hipaa/for-professionals/privacy/laws-regulations/index.html

[8] Trusted computing group, Trusted Platform Module (TPM) specifications, Apr. 2006.

[9] W. Aiello, S. M. Bellovin, M. Blaze, R. Canetti, J. Ioannidis, A. D. Keromytis, and O. Reingold, "Just fast keying: Key agreement in a hostile internet," *ACM Trans. Inf. Syst. Secur.*, vol. 7, no. 2, pp. 242–273, 2004.

[10] D. Arora, N. Aaraj, A. Raghunathan, and N. K. Jha, "Invisios: A lightweight, minimally intrusive secure execution environment," *ACM Trans. Embed. Comput. Syst.*, vol. 11, no. 3, pp. 60:1–60:20, 2012.

[11] E. D. Bell and J. L. La Padula, "Secure computer system: Unified exposition and multics interpretation," MITRE Corp., Bedford, MA, USA, Tech. Rep., 1976.

[12] K. J. Biba, "Integrity considerations for secure computer systems," MITRE Corp., Bedford, MA, USA, Tech. Rep., 1977.

[13] A. Datta, A. Derek, J. C. Mitchell, and D. Pavlovic, "A derivation system and compositional logic for security protocols," *J. Comput. Secur.*, vol. 13, no. 3, pp. 423–482, 2005.

[14] D. E. Denning, "A lattice model of secure information flow," *Commun. ACM*, vol. 19, pp. 236–243, 1976.

[15] S. DiBenedetto, P. Gasti, G. Tsudik, and E. Uzun, "ANDaNA: Anonymous named data networking application," in *Proc. Netw. Distrib. Syst. Security Symp.*, 2012.

[16] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theor.*, vol. IT-22, no. 6, pp. 644–654, Nov. 1976.

[17] J. S. Dwoskin and R. B. Lee, "Hardware-rooted trust for secure key management and transient trust," in *Proc. 14th ACM Conf. Comput. Commun. Security*, 2007, pp. 389–400.

[18] P. Gasti, G. Tsudik, E. Uzun, and L. Zhang, "DoS and DDoS in named-data networking," in *CoRR*, 2012.

[19] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. Briggs, and R. Braynard, "Networking named content," *Commun. ACM*, vol. 55, no. 1, pp. 117–124, 2012.

[20] T. Jaeger, D. H. King, K. R. B. Butler, S. Hallyn, J. Latten, and X. Zhang, "Leveraging IPsec for mandatory per-packet access control," in *Proc. SecureComm Workshops*, 2006, pp. 1–9.

[21] P. Jokela, A. Zahemszky, C. Esteve Rothenberg, S. Arianfar, and P. Nikander, "LIPSIN: Line speed publish/subscribe inter-networking," in *Proc. SIGCOMM Conf. Data Commun.*, 2009, pp. 195–206.

[22] C. Kaufman, P. Hoffman, Y. Nir, and P. Eronen, Internet key exchange protocol version 2 (IKEv2), 2010.

[23] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," in *Proc. Conf. Appl., Technol., Archit., Protocols Comput. Commun.*, 2007, pp. 181–192.

[24] L. Law, A. Menezes, M. Qu, J. Solinas, and S. Vanstone, "An efficient protocol for authenticated key agreement," *Des. Codes Cryptography*, vol. 28, no. 2, pp. 119–134, 2003.

[25] D. Levin, J. R. Douceur, J. R. Lorch, and T. Moscibroda, "Trinc: Small trusted hardware for large distributed systems," in *Proc. 6th USENIX Conf. Netw. Syst. Design Implementation*, 2009, pp. 1–14.

[26] N. Li, Z. Mao, and H. Chen, "Usable mandatory integrity protection for operating systems," in *Proc. IEEE Symp. Security Privacy*, 2007, pp. 164–178.

[27] Q. Li, M. Xu, J. Wu, X. Zhang, P. P. C. Lee, and K. Xu, "Enhancing the trust of internet routing with lightweight route attestation," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 2, pp. 691–703, Apr. 2012.

[28] H. Liu, S. Saroiu, A. Wolman, and H. Raj, "Software abstractions for trusted sensors," in *Proc. Int. Conf. Mobile Syst., Appl. Services*, 2012, pp. 365–378.

[29] M. Mannan, B. H. Kim, A. Ganjali, and D. Lie, "Unicorn: Two-factor attestation for data security," in *Proc. 18th ACM Conf. Comput. Commun. Security*, 2011, pp. 17–28.

[30] J. M. McCune, T. Jaeger, S. Berger, R. Cáceres, and R. Sailer, "Shamon: A system for distributed mandatory access control," in *Proc. ACSAC*, 2006, pp. 23–32.

[31] J. M. McCune, Y. Li, N. Qu, Z. Zhou, A. Datta, V. Gligor, and A. Perrig, "Trustvisor: Efficient TCB reduction and attestation," in *Proc. IEEE Symp. Security Privacy*, 2010, pp. 143–158.

[32] J. M. McCune, B. Parno, A. Perrig, M. K. Reiter, and A. Seshadri, "How low can you go? recommendations for hardware-supported minimal TCB code execution," in *Proc. 13th Int. Conf. Archit. Support Program. Lang. Oper. Syst.*, 2008, pp. 14–25.

[33] J. M. McCune, B. J. Parno, A. Perrig, M. K. Reiter, and H. Isozaki, "Flicker: An execution infrastructure for TCB minimization," in *Proc. 3rd ACM SIGOPS/EuroSys Eur. Conf. Comput. Syst.*, 2008, pp. 315–328.

[34] A. C. Myers and B. Liskov, "A decentralized model for information flow control," in *Proc. 16th ACM Symp. Operating Syst. Principles*, 1997, pp. 129–142.

[35] M. Nabeel and E. Bertino, "Efficient privacy preserving content based publish subscribe systems," in *Proc. 17th ACM Symp. Access Control Models Technol.*, 2012, pp. 133–144.

[36] S. Rafaeli and D. Hutchison, "A survey of key management for secure group communication," *ACM Comput. Surv.*, vol. 35, no. 3, pp. 309–329, 2003.

[37] Y. Rekhter and T. Li, A border gateway protocol 4 (BGP-4), RFC 4271, 1994.

[38] C. Yi, A. Afanasyev, L. Wang, B. Zhang, and L. Zhang, "Adaptive forwarding in named data networking," *Comput. Commun. Rev.*, vol. 42, no. 3, pp. 62–67, 2012.

[39] N. Zeldovich, S. Boyd-Wickizer, and D. Mazières, "Securing distributed systems with information flow control," in *Proc. 5th USENIX Symp. Netw. Syst. Des. Implementation*, 2008, pp. 293–308.

[40] X. Zhang, Z. Zhou, G. Hasker, A. Perrig, and V. Gligor, "Network fault localization with small TCB," in *Proc. IEEE 19th Int. Conf. Netw. Protocol*, 2011, pp. 143–154.

[41] R. Zuccherato and M. Nystrom, ISO/IEC 9798-3 authentication SASL mechanism, 2001.

Qi Li received the BSc and PhD degrees in computer science from Tsinghua University, Beijing, China. He is currently an associate professor with the Graduate School at Shenzhen, Tsinghua University. He was a researcher at ETH Zurich, and a postdoctoral fellow in the Institute for Cyber Security, the University of Texas at San Antonio. His research interests include system and network security, Internet, and large-scale distributed systems. He is a member of the IEEE.

Ravi Sandhu received the BTech and MTech degrees from IIT Bombay and Delhi, and the MS and PhD degrees from Rutgers University. He is an executive director in the Institute for Cyber Security, University of Texas at San Antonio, where he holds the Lutcher Brown Endowed Chair in Cyber Security. Previously, he was on the faculty at George Mason University (1989-2007) and Ohio State University (1982-1989). A prolific and highly cited author, his research has been funded by the US National Science Foundation (NSF), NSA, NIST, DARPA, AFOSR, ONR, AFRL and private industry. His seminal papers on role-based access control established it as the dominant form of access control in practical systems. His numerous other models and mechanisms have also had considerable real-world impact. He served as an editor-in-chief of the *IEEE Transactions on Dependable and Secure Computing*, and previsouly as a founding editor-in-chief of *ACM Transactions on Information and System Security*. He was the chairman of ACM SIGSAC, and founded the ACM Conference on Computer and Communications Security, the ACM Symposium on Access Control Models and Technologies and the ACM Conference on Data and Application Security and Privacy. He has served as a general chair, Steering Committee chair, Program chair and a Committee member for numerous security conferences. He has consulted for leading industry and government organizations, and has lectured all over the world. He is an inventor on 30 security technology patents and has accumulated over 27,000 Google Scholar citations for his papers. At the Institute for Cyber Security he leads multiple teams conducting research on many aspects of cyber security including secure information sharing, social computing security, cloud computing security, secure data provenance, and botnet analysis and detection, in collaboration with researchers all across the world. His web site is at www.profsandhu.com. He is a fellow of the IEEE, ACM, and AAAS, and has received awards from the IEEE, ACM, NSA, and NIST.

Xinwen Zhang received the PhD degree in information security from George Mason University in 2006. He is a principal engineer with Samsung Research America. His current research interests include security policies, models, architectures, mechanism in general computing and networking systems, secure and trusted network architecture, cloud computing, mobile platforms, and storage systems. He is a member of the IEEE.

Mingwei Xu received the PhD degree from Tsinghua University. He is a full professor in the Department of Computer Science, Tsinghua University. His research interest includes computer network architecture, high-speed router architecture, and network security. He is a member of the IEEE.