Producer Anonymity Based on Onion Routing in Named Data Networking

Kentaro Kita, *Student Member, IEEE*, Yuki Koizumi, *Member, IEEE*, Toru Hasegawa, *Member, IEEE*, Onur Ascigil, *Member, IEEE*, and Ioannis Psaras, *Member, IEEE*

***Abstract*—Named Data Networking (NDN) is one of promising next generation Internet architectures that aim to realize efficient content distribution. However, in terms of producer anonymity, NDN has a serious problem that adversaries can easily learn who publishes what content due to its feature that content is inherently tied to the producer by the content name and the signature. In this article, we first define producer anonymity rigorously in terms of content-producer unlinkability, and then design a system to achieve it. Our design is based on hidden service, which is an onion routing-based system in IP, however, we improve it to take full advantage of NDN. We demonstrate that our system provides a level of anonymity comparable to hidden service with lower overhead through analysis and experiment.**

***Index Terms*—Named data networking, producer anonymity, onion routing.**

I. Introduction

IT IS becoming increasingly common for the Internet to be used to distribute content rather than to interconnect hosts to enable them to communicate with each other as originally intended. This change has led researchers to design alternative Internet architectures. One of the promising candidates is Named Data Networking (NDN) [1], which shifts focus from host to content. In NDN, each content is carried in a Data packet, which is identified and located by a URL-like content name and signed by its producer. A consumer who wishes to obtain some specific content issues an explicit request, called an Interest packet, specifying the content name. Since every content is signed, consumers can verify the integrity and provenance of the content regardless of who returned it. Thus, Interest packets can be satisfied by producers or intermediate routers caching content. In-network caching yields several benefits in terms of efficiency of content distribution, such as reductions in delay, bandwidth usage, and producer load.

Manuscript received March 24, 2020; revised July 10, 2020; accepted August 3, 2020. Date of publication August 24, 2020; date of current version June 10, 2021. The research leading to these results has received funding from NICT (Contract No. 191). The associate editor coordinating the review of this article and approving it for publication was H. Zhang. *(Corresponding author: Kentaro Kita.)*

Kentaro Kita, Yuki Koizumi, and Toru Hasegawa are with the Graduate School of Information Science and Technology, Osaka University, Suita 5650871, Japan (e-mail: k-kita@ist.osaka-u.ac.jp; ykoizumi@ist.osaka-u.ac.jp; t-hasegawa@ist.osaka-u.ac.jp).

Onur Ascigil and Ioannis Psaras are with the Department of Electrical and Electronic Engineering, University College London, London WC1E 6EA, U.K. (e-mail: o.ascigil@ucl.ac.uk; i.psaras@ucl.ac.uk).

Digital Object Identifier 10.1109/TNSM.2020.3019052

In addition to realizing efficiency, it is desirable to incorporate mechanisms to guarantee privacy and anonymity in NDN as pervasive monitoring and censorship on the Internet have become serious issues of concern [2], [3]. Anonymity is required in some privacy-sensitive applications and protocols, such as location-based services and cryptocurrencies [4], [5], [6], [7], to hide users' identities and activity histories. In NDN, anonymity can be classified into consumer anonymity and producer anonymity, depending on who appears to be anonymous. Intuitively, consumer/producer anonymity mean to hide who requests/publishes what content. In terms of unlinkability, they can be defined as *request-consumer unlinkability* and *content-producer unlinkability*, respectively. Because request-consumer unlinkability refers to requests and their senders, consumer anonymity can be regarded as a type of sender anonymity [8], [9]. ANDāNA [10] has been designed for NDN based on onion routing [11] to achieve a level of consumer anonymity comparable to Tor [12], which is the most widely used system to achieve sender anonymity in IP. In onion routing, encapsulated packets pass through *circuits*, each of which consists of several voluntary hosts called *anonymizing routers* (or *onion routers*), to conceal their origins.

In contrast, no systems to achieve producer anonymity have been proposed for NDN. Before designing such systems, however, we should rigorously define producer anonymity because conventional anonymity definitions in IP are not appropriate for producer anonymity in NDN. For example, receiver anonymity [8], which is used as an anonymity model for hosts publishing content in IP, is defined as request-receiver unlinkability. However, this definition does not completely capture the notion of producer anonymity because each request can be satisfied by any intermediate entity before the intended producer receives it in NDN. In this article, we define producer anonymity by focusing on packet flow on the networks, whereas some previous studies define receiver anonymity by focusing on hosts that could have received a particular packet [8], [13]. Intuitively, a producer is said to be anonymous if content publishing performed by the producer has only a negligible effect to the network flow observed by adversaries.

Producer anonymity is difficult to achieve in naive NDN due to its feature that each content is inherently tied to its producer by the *producer name*, which is a globally routable name of the producer, and the signature. To solve this issue, we design a system to achieve producer anonymity in NDN based on hidden service [12], [14], which is implemented on Tor to achieve receiver anonymity in IP. Specifically, we improve

hidden service to take full advantage of NDN by leveraging the RICE protocol [15], while using *onion addresses* and *rendezvous points* in hidden service. The advantages of our system are summarized as follows.

First, our system improves RTT in content retrieval by building circuits on RICE's *reverse paths*, each of which consists of temporal FIB entries in a sequence of intermediate routers. Hereinafter, the network layer routers are called *regular routers* to distinguish them from anonymizing routers. In our system, consumers' Interest packets are forwarded along reverse paths created by producers. This enables producers to hide their producer names even to the first-hop anonymizing routers in contrast to the fact that receivers must reveal their IP addresses to the first-hop anonymizing routers in hidden service. This producer anonymity provided at the network layer reduces by one the number of anonymizing routers in each circuit required to achieve a comparable level of anonymity to hidden service.

Second, our system offers better security than hidden service against the *predecessor attack* [16], in which adversaries wait until producers unfortunately choose compromised anonymizing routers as the members of their circuits. The predecessor attack is a substantial threat to anonymity systems based on onion routing and no comprehensive countermeasures have been proposed. Hidden service mitigates this attack by employing *entry guards*, which are the first-hop anonymizing routers hiding receivers' IP addresses [17]. In our system, the first-hop regular routers of producers play the role of entry guards because producers' identifiers, such as MAC addresses, are disclosed only to them. Moreover, rendezvous points in our system play the role of *exit guards*, which are the fixed last-hop anonymizing routers similar to entry guards. We prove that these changes decrease the probability of the predecessor attack succeeding.

The contributions of this article are summarized as follows:

- We rigorously define producer anonymity in terms of content-producer unlinkability. To the best of our knowledge, this is the first study which addresses producer anonymity under a realistic adversarial model.
- We design a system to achieve producer anonymity by incorporating RICE in hidden service.
- We prove that our system achieves a level of anonymity comparable to hidden service with one fewer anonymizing router and show that our system has better security against the predecessor attack.

The rest of this article is organized as follows: Section II describes existing studies on onion routing-based systems in IP and NDN. We define producer anonymity in Section III. Section IV describes the design of our system. We analyze the anonymity of our system in Section V and the second half of Section VI. The the first half of Section VI presents the performance evaluation. Section VII summarizes related works. Section VIII concludes this article.

II. Preliminaries

A. Tor and Hidden Service

We describe Tor [12] and hidden service [12], [14] to examine how sender and receiver anonymity are achieved based on onion routing in IP. In this article, we focus on communication in which a sender issue a content request to a receiver in the context of IP communication for the sake of simplicity. According to the definition in [8], we define sender/receiver anonymity as unlinkability of a plaintext packet and its sender/receiver, respectively.

In Tor, the sender first builds a circuit by exchanging secret keys with several anonymizing routers incrementally. Then, the sender issues a packet encapsulated in multiple layers of secret key encryption along the circuit. Each anonymizing router decrypts the top layer and forwards it to the next anonymizing router or the receiver. Because the packet is forwarded through distributed anonymizing routers while altering its bit pattern by decryption, its origin is mixed with other senders from the perspective of adversaries. Although the sender periodically changes the circuit, the first-hop anonymizing router hiding the sender's IP address is used repeatedly for a longer period of time. Such fixed first-hop anonymizing routers are called entry guards. Entry guards have been proposed to mitigate the predecessor attack [16], [17] as described in detail in Section V-C.

Hidden service is deployed on Tor so that receivers can hide their identities even from senders. In the following description, we assume that all entities communicate through circuits. The receiver first generates a pseudonym called an *onion address* from her/his public key. Then, the receiver asks several anonymizing routers to act as *introduction points*, which relay senders' connection requests to the receiver. If the anonymizing routers accept the requests, the receiver generates a *descriptor*, which contains the IP addresses of the introduction points. The descriptor is uploaded to several anonymizing routers called *descriptor directories*. The sender learns the onion address in some out-of-band way, downloads the descriptor, and asks an anonymizing router to play the role of a *rendezvous point* by building a circuit which includes the rendezvous point as the last-hop anonymizing router. Then, the sender issues a connection request through one of the introduction points. This connection request contains the IP address of the rendezvous point and the first half of keying materials, e.g., those in the Diffie-Hellman key agreement protocol. The receiver establishes a connection to the sender through the rendezvous point while sending the second half of keying materials. The sender and the receiver derive a shared secret key used to encrypt and authenticate packets from these key materials. At this time, the sender can send packets to the receiver through the rendezvous point without knowing the IP address of the receiver.

B. ANDāNA

In this subsection, we describe ANDāNA, which is an initial attempt to adapt Tor to NDN to achieve consumer anonymity [10]. Because we design a system to achieve producer anonymity based on hidden service deployed on Tor, our system has affinities with ANDāNA.

Similar to sender anonymity in IP, consumer anonymity can be defined as request-consumer unlinkability, i.e., unlinkability of a plaintext Interest packet and a consumer who sends

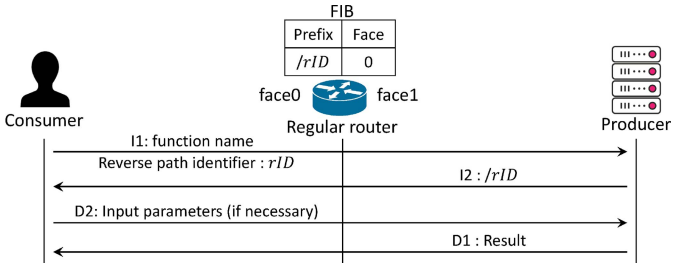

Fig. 1. Overview of the RICE protocol.

it. The content-oriented design of NDN is compatible with consumer anonymity. In particular, consumer anonymity is naturally achieved against adversaries on core networks because each Interest packet carries information about which Data packet is being requested but not about who is requesting it. However, this kind of consumer anonymity is insufficient against adversaries on edge networks because they can directly observe who sends a specific Interest packet.

To solve this issue, ANDāNA [10] has been designed. ANDāNA has the advantage that it achieves a level of anonymity comparable to that of Tor with one fewer anonymizing router. By the term "level of anonymity", we mean the number of anonymizing and regular routers an adversary must compromise to break anonymity by tracking packets throughout a circuit. For example, in the case that three anonymizing routers are included in a sender's circuit in Tor, the level of anonymity is three. This is because an adversary can learn the sender's IP address by compromising the first-hop anonymizing router and track packets from the sender throughout the circuit by compromising the rest of the anonymizing routers. The above advantage is thanks to NDN networks inherently providing a level of consumer anonymity equivalent to that achieved by passing through one anonymizing router. More specifically, only the first-hop regular routers of consumers can learn their identities in ANDāNA, whereas the first-hop anonymizing routers can learn senders' IP addresses in Tor. Thus, the adversary must compromise the first-hop regular routers in addition to the anonymizing routers in circuits to break consumer anonymity.

C. RICE

Our system runs on RICE [15], which is a communication protocol having different features from the original Interest-Data exchanges in NDN. RICE is originally designed to enable consumers to delegate computation to remote entities. The overview of the RICE protocol is illustrated in Fig. 1. A consumer first issues an Interest packet (called an I1 packet) specifying the name of a function the consumer asks to execute. This Interest packet also carries a consumer-chosen *reverse path identifier*. When each intermediate regular router receives the I1 packet, it creates an ephemeral FIB entry to forward other Interest packets (called I2 packets) specifying the reverse path identifier to the interface from which the I1 packet came. In Fig. 1, a regular router creates a FIB entry to forward I2 packets specifying /rID as their name prefixes to face0. As a result, a sequence of FIB entries on the intermediate regular routers, called a *reverse path*, is created by the I1 packet. If a producer who has the capability to execute the function receives the I1 packet, it sends back I2 packet(s) along the reverse path to let the consumer return some input parameters for execution with the corresponding Data packet(s) (called D2 packet(s)). Upon receiving the D2 packet(s), the producer executes the function and returns its result with the Data packet (called a D1 packet) corresponding to the I1 packet or in another Interest/Data exchanges. Specifically, our system leverages the feature of RICE that senders of I1 packets enable remote entities to send I2 packets back to them without advertising their routable names to achieve producer anonymity efficiently.

III. Producer Anonymity

In this section, we identify issues of naive NDN that make producer anonymity difficult to achieve, and then present our adversarial model and rigorously define producer anonymity.

A. Issues Regarding Producer Anonymity

We first illustrate two typical scenarios where producer anonymity is required. In the descriptions, we indicate a producer and a consumer by (P) and (C), respectively. 1) Assuming that Alice (P) wishes to launch a website that provides people (C) with information about fraud by some companies or governments, she may lose her job or be punished if she is not anonymous. 2) Assume that Bob (P) agrees to offer his health information, such as his age, weight, and blood pressure value, to a server for statistical surveys (C). However, he might wish to hide his identity from the server for his privacy.

In contrast to consumer anonymity, producer anonymity has not been thoroughly studied in NDN. Producer anonymity should be defined as being somewhat different from receiver anonymity, whereas consumer anonymity can be regarded as a type of sender anonymity. If the notion of receiver anonymity is applied to NDN, it can be defined as request-producer unlinkability, i.e., unlinkability of a plaintext Interest packet and a producer who receives it. However, we are rather interested in content-producer unlinkability, i.e., unlinkability of a plaintext Data packet and its producer. The difference lies in the communication features of IP and NDN.

Receiver anonymity is originally defined for host-oriented IP architecture [8], in which two hosts communicate with each other based on connections/sessions established between them. In contrast, content-oriented NDN architecture does not assume such end-to-end connections/sessions. Indeed, every content can occasionally be returned from any intermediate entity caching it. In this case, producers do not receive any packets, however, content-producer unlinkability can still be violated because Data packets are strongly bound to their producers by their two components, producer names and signatures. First, a producer name is a human-readable globally-routable name prefix of every content name of a producer. Each producer name plays the dual roles of the identifier and the locator of a producer simultaneously. This implies that one producer name reveals a similar amount of information to a pair of an IP address and a domain

TABLE I
SUMMARY OF NOTATION

Notation	Description
$\mathbb{C}$	Set of all consumers
$\mathbb{P}$	Set of all producers
$\mathbb{A}$	Set of all anonymizing routers
$\mathbb{R}$	Set of all regular routers
$\mathbb{D}$	Set of all valid Data packets
$\mathcal{A}dv$	Adversary
κ	Security parameter
CF	Configuration
$\perp$	Special symbol meaning no circuit
(pk_{id}, sk_{id})	Identity key pair of a producer
k_i	Secret key exchanged between a producer and the i-th anonymizing router
sID_i	Session identifier exchanged between a producer and the i-th anonymizing router
rID_i	Reverse path identifier assigned for the i-th link between anonymizing routers by a producer
H	Cryptographic hash function
$\mathsf{Cert}(pk)$	Public key certificate of public key pk
Enc_{k_i}	Secret key encryption algorithm with secret key k_i
Dec_{k_i}	Secret key decryption algorithm with secret key k_i
σ_{sk}	Signature generated with private key sk
t_{k_i}	MAC tag generated with secret key k_i
$f_{\mathbb{A}}$	Fractions of compromised anonymizing routers
$f_{\mathbb{R}}$	Fractions of compromised regular routers
q	Probability that an anonymizing router becomes unavailable in a round
m	Number of rounds

name, which is often encrypted by using cryptographic protocols such as TLS. Thus, each producer name carries enough information to uniquely identify the producer and to obtain more meaningful information, such as the producer's locations and affiliations [18]. Second, the signature carried in each Data packet is also regarded as the producer's identifier because it is publicly verifiable with her/his unique public key. These features make producer anonymity difficult to achieve, in contrast to consumer anonymity being naturally achieved.

B. Adversarial Model

Table I summarizes the notation used in this article. We define $\mathbb{C}$, $\mathbb{P}$, $\mathbb{A}$, and $\mathbb{R}$ as the sets of all consumers, producers, anonymizing routers, and regular routers, respectively. Each intersection of these sets can be non-empty.

We assume that the goal of an adversary $\mathcal{A}dv$ is to identify who publishes what content. Following the adversarial model in Tor and ANDāNA [10], [12], we assume $\mathcal{A}dv$ that is 1) non-global, 2) active, and 3) efficient. First, we assume that $\mathcal{A}dv$ compromises only a proper subset of entities. Then, $\mathcal{A}dv$ can be represented as a 4-tuple: $\mathcal{A}dv = (\mathbb{C}_{\mathcal{A}dv}, \mathbb{P}_{\mathcal{A}dv}, \mathbb{A}_{\mathcal{A}dv}, \mathbb{R}_{\mathcal{A}dv}) \subset (\mathbb{C}, \mathbb{P}, \mathbb{A}, \mathbb{R})$, where $\mathbb{C}_{\mathcal{A}dv}$, $\mathbb{P}_{\mathcal{A}dv}$, $\mathbb{A}_{\mathcal{A}dv}$, and $\mathbb{R}_{\mathcal{A}dv}$ are the sets of compromised consumers, producers, anonymizing routers, and regular routers, respectively. This assumption is reasonable because these entities are assumed to be distributed throughout the networks. Second, $\mathcal{A}dv$ is capable of performing any action that the compromised entities can perform, such as observing, altering, and dropping packets. Third, $\mathcal{A}dv$ can run any algorithms only in time polynomial in a security parameter κ. This is a fundamental assumption for almost all of modern cryptographic protocols [19].

Note that we do not assume $\mathcal{A}dv$ who aims to block some specific content throughout the networks, such as worldwide censorship authorities. To evade such censorship, we must encrypt Interest and Data packets in an end-to-end manner in exchange for the advantage of content caching because censorship can be enforced by simply dropping Interest/Data packets which contain some censored keywords even if their origins are anonymous [20], [21]. In addition, we do not aim to achieve consumer anonymity against $\mathcal{A}dv$, whereas hidden service is designed so that it provides both sender and receiver anonymity. This enables our system to leverage cached content close to consumers because circuits are not needed between consumers and rendezvous points. Moreover, leveraging cached content improves producer anonymity in our system as shown in Section V.

C. Anonymity Definition

We present a formal definition of producer anonymity in terms of content-producer unlinkability. In this article, we define producer anonymity by using the notion of indistinguishable configurations [10], [22]. In brief, a *configuration* consists of packets and network entities forwarding them and represents the packet flow in a *round* of communication. We define a round as a series of content publishing of producers performed without changing their circuits. For the sake of simplicity, we assume that each producer is requested at most one piece of content by a consumer in each round. This assumption does not affect the definition of producer anonymity because producer anonymity is broken if a producer is linked to even a piece of content.

Formally, we define a configuration CF as a mapping which associates producers with established circuits, consumers who issue Interest packets, and the corresponding plaintext Data packets, as follows:

Definition 1 (Configuration):

$$\mathsf{CF} : \mathbb{P} \to \mathbb{A}^n \cup \{\perp\} \times \mathbb{C} \times \mathbb{D},$$

where $\perp$ is a special symbol to represent the case where content is returned to a consumer from a cache on a regular router without using a circuit and $\mathbb{D}$ is the set of all the Data packets which follow the prescribed packet format.

For convenience of explanation, we also define the following four mappings which represent elements in a configuration CF; $\mathsf{CF}_{\mathbb{A}} : \mathbb{P} \to \mathbb{A}^n \cup \{\perp\}$ (selections of *n* anonymizing routers in circuits), $\mathsf{CF}_{\mathbb{A}_i} : \mathbb{P} \to \mathbb{A}$ (selections of *i*-th anonymizing routers in circuits), $\mathsf{CF}_{\mathbb{C}} : \mathbb{P} \to \mathbb{C}$ (associations between producers and consumers), and $\mathsf{CF}_{\mathbb{D}} : \mathbb{P} \to \mathbb{D}$ (selections of Data packets to publish). For example, if a producer $p \in \mathbb{P}$ publishes a Data packet $dat \in \mathbb{D}$ along a circuit consisting of *n* anonymizing routers $\{a_1, \ldots, a_n\} \in \mathbb{A}^n$ to a consumer $c \in \mathbb{C}$ in a configuration CF, then $\mathsf{CF}(p) = \{a_1, \ldots, a_n, c, dat\}$, $\mathsf{CF}_{\mathbb{A}}(p) = \{a_1, \ldots, a_n\}$, $\mathsf{CF}_{\mathbb{A}_i}(p) = a_i$, $\mathsf{CF}_{\mathbb{C}}(p) = c$, and $\mathsf{CF}_{\mathbb{D}}(p) = dat$. For another example, if c receives dat of p from a cache on a regular router in CF, then $\mathsf{CF}(p) = \{\perp, c, dat\}$. In this case, *p* does not send/receive any packets.

Because $\mathcal{A}dv$ can eavesdrop packets only at a portion of entities and Data packets are encrypted throughout circuits, for

a configuration CF, there can exist another possible configuration CF' which yields packet flow that seem to be consistent with those yielded in CF from the viewpoint of $\mathcal{A}dv$. In this case, $\mathcal{A}dv$ cannot differentiate CF from CF', i.e., CF and CF' are indistinguishable with respect to $\mathcal{A}dv$. To formalize this notion, let $\mathcal{A}dv(1^\kappa, \widehat{\mathsf{CF}})$ denote any probabilistic algorithm run by $\mathcal{A}dv$ in time polynomial in the security parameter κ such that, given $\widehat{\mathsf{CF}}$ chosen uniformly at random from a known set $\{\mathsf{CF}, \mathsf{CF}'\}$, it outputs 1 if it deduces that $\widehat{\mathsf{CF}}$ corresponds to CF. From this definition, $\mathsf{Pr}[\mathcal{A}dv(1^\kappa, \mathsf{CF}) = 1]$ and $\mathsf{Pr}[\mathcal{A}dv(1^\kappa, \mathsf{CF}') = 1]$ represent the probability that $\mathcal{A}dv$ correctly identifies inputted CF as CF and the probability that $\mathcal{A}dv$ incorrectly identifies inputted CF' as CF, respectively. By using $\mathcal{A}dv(1^\kappa, \widehat{\mathsf{CF}})$, indistinguishable configurations can be defined as follows.

Definition 2 (Indistinguishable Configurations): Two configurations CF and CF' are said to be indistinguishable with respect to $\mathcal{A}dv$, denoted as $\mathsf{CF} \equiv_{\mathcal{A}dv} \mathsf{CF}'$, if for $\mathcal{A}dv$ there exists a negligible function $\epsilon(\cdot)$, such that

$$\left|\mathsf{Pr}[\mathcal{A}dv(1^\kappa, \mathsf{CF}) = 1] - \mathsf{Pr}\left[\mathcal{A}dv\left(1^\kappa, \mathsf{CF}'\right) = 1\right]\right| \leq \epsilon(\kappa),$$

for the security parameter κ.

From the definition, the left side of the inequality in definition 2 represents the probability that $\mathcal{A}dv$ can distinguish CF from CF'. A function is said to be negligible if it is asymptotically smaller than an inverse function of any positive polynomial. Therefore, $\mathsf{CF} \equiv_{\mathcal{A}dv} \mathsf{CF}'$ implies that $\mathcal{A}dv$ can correctly differentiate CF and CF' only with negligible probability.

Content-producer unlinkability is achieved with respect to $\mathcal{A}dv$ if $\mathcal{A}dv$ can determine neither which content a producer $p \in \mathbb{P}$ is providing nor whether p or another producer $p' \in \mathbb{P}$ $(p' \neq p)$ is publishing particular content. This notion can be formalized by using indistinguishable configurations as follows: given the actual configuration CF (i.e., the configuration which reflects the actual network activities $\mathcal{A}dv$ is observing) in which p publishes a Data packet *dat*, content-producer unlinkability is achieved if there exist another imaginary but possible configuration CF' in which p' publishes *dat* and p publishes another Data packet, and $\mathcal{A}dv$ cannot determine whether s/he is observing either CF or CF'. This implies that content publishing of p and p' causes only a negligible difference in $\mathcal{A}dv$'s observation. We define producer anonymity in terms of content-producer unlinkability as follows:

Definition 3 (Producer Anonymity): $p \in (\mathbb{P} \setminus \mathbb{P}_{\mathcal{A}dv})$ has producer anonymity in configuration CF with respect to $\mathcal{A}dv$ if $\exists \mathsf{CF}' \equiv_{\mathcal{A}dv} \mathsf{CF}$ such that $\exists p' \in (\mathbb{P} \setminus \mathbb{P}_{\mathcal{A}dv})$, $\mathsf{CF}'_{\mathbb{D}}(p') = \mathsf{CF}_{\mathbb{D}}(p) \neq \mathsf{CF}_{\mathbb{D}}(p') = \mathsf{CF}'_{\mathbb{D}}(p)$ and $p' \neq p$.

From the perspective of an anonymity set, which is generally defined as the set of all possible subjects that might cause an action [8], the anonymity set with respect to p's content publishing consists of p and all the producers who satisfy the requirements for p' in the Definition 3. As the number of producers in the anonymity set increases, p is hidden in the larger crowd, and the anonymity degree increases.

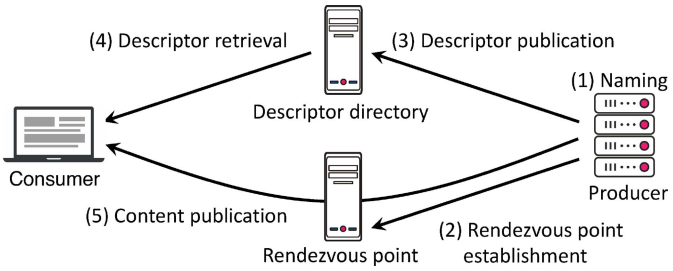

Fig. 2. Overview of the system to achieve producer anonymity.

IV. Design

In this section, we first provide an overview of our system, and then describe its protocols in detail.

A. System Model

One of the key constraints in designing a system to achieve producer anonymity is that producers cannot initiate content publishing without first receiving Interest packets from consumers. This is due to the feature of NDN that Data packets are published only when the corresponding Interest packets are issued to maintain one-to-one flow balance of Interest and Data packets. Taking this constraint into account, our system uses pseudonyms of producers, called onion names, and rendezvous points. Onion names are used to distinguish services provided by anonymous producers. However, consumers cannot send Interest packets toward producers only by using onion names because onion names are designed so that their producers cannot be identified and located as described in Section IV-B. Thus, rendezvous points accept Interest packets specifying onion names under their globally routable names and forward them to anonymous producers.

The overview of our system is illustrated in Fig. 2. We assume that every anonymizing router advertises its routable name and its public key certificate via directory nodes like Tor and hidden service. 1) A producer who wishes to anonymously publish content generates a long-term public/private key pair and derives an onion name from the public key. 2) The producer asks an anonymizing router to act as a rendezvous point through a circuit. If the anonymizing router accepts it, the producer waits for content requests while maintaining the circuit. We assume that the producer changes circuits periodically like Tor and hidden service, however, the rendezvous point is used for a longer period of time until it becomes unavailable. 3) The producer uploads her/his descriptor to some of the *descriptor directories*, which are anonymizing routers for publishing descriptors, through another circuit. A descriptor contains the producer's public key certificate, the rendezvous point's routable name, and the rendezvous point's public key certificate. The name of the descriptor is derived from the corresponding onion name. 4) A consumer who learns the onion name in some out-of-band way downloads the descriptor from one of the responsible descriptor directories by specifying the descriptor name. 5) To obtain content, the consumer issues a request toward one of the rendezvous points. The corresponding content is returned by an intermediate regular router between the consumer and the rendezvous point if it

has cached the content. Otherwise, the rendezvous point forwards the request to the producer through the circuit. In this procedure, the producer can publish content without revealing anything more than the onion name.

Note that the consumer builds circuits neither to the descriptor directory nor the rendezvous point since we focus solely on producer anonymity in this article. In addition, hidden service uses introduction points to enable senders to send connection requests to receivers and exchange secret keys with them as described in Section II-A. This additional communication phase is required to establish connections and to encrypt packets in an end-to-end manner with the secret keys to evade censorship enforced throughout the networks. However, our system does not use introduction points because we do not assume such worldwide censorship as described in Section III-B.

B. Naming

In our system, an onion name is used as the prefix of every content name of a producer instead of her/his producer name. There are several requirements for onion names that are different from those for producer names. It must be ensured that every onion name is 1) non-routable, 2) non-human-readable, and 3) unique and securely bound to both its producer and her/his public key without relying on any authorities. First, if an onion name is advertised as the routable name of a producer, $\mathcal{A}dv$ can easily correlate the onion name with the producer, and thus producer anonymity cannot be achieved. Second, onion names should not be human-readable to prevent information leakage from themselves. Third, in the case of producer names, uniqueness of names and bindings between a producer, her/his name, and her/his public key are established by trusted authorities, such as ICANN and CAs [23], [24]. However, our system does not leverage such authorities to avoid any single point of failure in terms of anonymity.

Taking these requirements into account, producers generate their onion names from fresh public keys. A producer first generates a long-term public/private key pair (pk_{id}, sk_{id}) called an *identity key pair*, and the corresponding self-signed public key certificate $\mathsf{Cert}(pk_{id})$ signed with sk_{id}. Note that $\mathsf{Cert}(pk_{id})$ must be generated so that it does not contain its producer's identifiers except the public key. We assume the length of this key pair is a function of the security parameter κ. The producer uses "onion" as the top component and the hash of pk_{id} as the second component of the onion name, respectively. By using the onion name, content names of the producer are represented as follows:

$$/\mathsf{onion}/H(pk_{id})/\langle\mathsf{suffix}\rangle,$$

where $\langle\mathsf{suffix}\rangle$ denotes the name suffixes determined by the producer, e.g., $\langle\mathsf{suffix}\rangle = \mathsf{article/xyz/html}$.

The onion names satisfy the three requirements. Onion names are not routable because they are just hashes of public keys. Thus, consumers cannot send Interest packets directly to producers. For the same reason, onion names are non-human-readable and a collision of onion names occurs only with negligible probability. Finally, the bindings between a producer, her/his onion name, and her/his public key are securely established as follows: the producer is bound with the onion name and the public key because their ownership can be proved with a signature which can be generated only with the producer's private key corresponding to the public key pk_{id}, and the onion name is bound with the public key because the onion name is self-certifying, i.e., the onion name contains the hash of the public key. In addition, no authorities are required while establishing these bindings because the public key certificate $\mathsf{Cert}(pk_{id})$ and onion names are locally generated.

In terms of trust of producers, there is an inherent conflict between producer anonymity and the naive NDN's trust mechanism, in which a consumer of a piece of content verifies its producer's certificate along the trust chain, and trusts the producer if the trust chain reaches one of the consumer's trust anchors. In contrast, our system has no built-in mechanisms to provide consumers with the information they need to decide onion names to trust since self-signed certificates are used to keep private the bindings between producers, their names, and their public keys. Therefore, if necessary, trust should be established to producers' onion names instead of their identities by using reputation systems or in some ad-hoc manners [25], [26], [27]. For example, there are several websites publishing the lists of what kind of services are offered under some onion addresses in the current hidden service. We believe that such mechanisms help consumers decide onion names whose content they retrieve and encourages anonymous producers to behave trustworthily.

Similarly, consumers cannot directly authenticate anonymous producers. Thus, consumers authenticate onion names instead. When a consumer receives a Data packet belonging to an onion name of her/his interest, the consumer confirms that it is certainly created by the correct owner of the onion name by verifying whether the signature is valid for the public key corresponding to the onion name. Since onion names and public key certificates reveal nothing about producers' identities, producer anonymity cannot be broken in this kind of authentication process.

C. Rendezvous Point Establishment

Next, the producer asks an anonymizing router to act as a rendezvous point by sending it the onion name and the public key certificate $\mathsf{Cert}(pk_{id})$. Hereinafter, we refer to this anonymizing router simply as a rendezvous point. To prevent $\mathcal{A}dv$ from impersonating the producer, this request should contain the signature generated with sk_{id}, denoted by $\sigma_{sk_{id}}$. The rendezvous point accepts it only if both the onion name and $\sigma_{sk_{id}}$ are valid for pk_{id} obtained from $\mathsf{Cert}(pk_{id})$.

The important point is how to send them to the rendezvous point. It might be problematic to send them with Interest packets because they are not designed to carry much data. A straightforward way to send them with Data packets is that the producer advertises her/his producer name to the first-hop anonymizing router to have it forward Interest packets from the rendezvous point requesting them. This is the same approach as hidden service, in which receivers' IP addresses are given to the first-hop anonymizing routers as source addresses. In

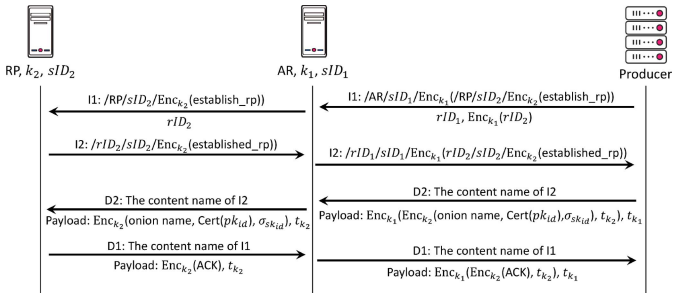

Fig. 3. Anonymous rendezvous point establishment.

contrast, the producer has the first-hop anonymizing router forward such Interest packets along a reverse path used in RICE in our approach. This enables the producer to send Data packets without revealing her/his identity even to the first-hop anonymizing router.

In the following description, we explain how the producer can establish the rendezvous point through a circuit built on reverse paths. Fig. 3 shows the communication sequence, where a circuit includes one anonymizing router other than the rendezvous point. Let AR and RP denote the anonymizing router and the rendezvous point, and /AR and /RP denote their routable names, respectively. We use any CCA-secure secret key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, where Gen is a key generation algorithm which generates a secret key according to inputted security parameter κ and Enc and Dec are an encryption and a decryption algorithm with the secret key, respectively. CCA-secure encryption schemes are probabilistic and non-malleable: 1) ciphertexts are randomized so that $\mathcal{A}dv$ cannot gain any partial information on the plaintexts, and 2) given a ciphertext, $\mathcal{A}dv$ cannot generate a different ciphertext such that their plaintexts are somehow related [19]. We assume that the producer has built a circuit anonymously by exchanging secret key $k_i \leftarrow \mathsf{Gen}(\kappa)$ and session identifier sID_i chosen uniformly and independently at random from $\{0,1\}^\kappa$ with AR and RP. This can be done with standard Interest/Data packets exchanges [28].

The producer first issues an I1 packet which includes establish_rp as a name component to RP to request rendezvous point establishment. The I1 packet is encapsulated in multi-layers of encryption by using Enc. We assume that Interest packets from the producer always pass through one regular router offered by an ISP as the first-hop regular router, hereinafter. AR and RP remove the top layers of the received I1 packets by using Dec with the secret keys k_1 and k_2 corresponding to the session identifiers sID_1 and sID_2 specified in the I1 packet, respectively. The I1 packet also carries reverse path identifiers rID_1 and rID_2 in each layer to create two reverse paths between the producer and AR and between AR and RP. rID_1 and rID_2 are chosen uniformly and independently at random from $\{0,1\}^\kappa$ to prevent $\mathcal{A}dv$ from linking the incoming and the outgoing packet at a non-compromised anonymizing router. On the receipt of the I1 packet, RP issues an I2 packet specifying the content name $/rID_2/sID_2/\mathsf{Enc}_{k_2}$ (established_rp) along the reverse path. This I2 packet notifies the producer that RP has agreed to act as the rendezvous point and is requesting the D2 packet containing the onion name, $\mathsf{Cert}(pk_{id})$, and $\sigma_{sk_{id}}$. AR encrypts the entire content name of the I2 packet with k_1 and appends rID_1 as the new name prefix to forward it along the reverse path to the producer. The D2 packet is transported by using PIT entries created by the I2 packet, while being decrypted with k_1 and k_2. The D2 packet also contains MAC tags generated with k_1 and k_2, denoted by t_{k_1} and t_{k_2}, to enable each anonymizing router to verify the origin of the D2 packet. Finally, after receiving the D2 packet, the rendezvous point returns the D1 packet corresponding the I1 packet to notify that the D2 packet has been received.

In these Interest/Data packet exchanges, AR and RP cannot learn their predecessors, i.e., they cannot learn the producer and AR, respectively. This is because the I1/D2 packets do not carry their senders' identities. In contrast, the first-hop regular router can learn the MAC addresses of the producer.

The rendezvous point establishment protocol requires the intermediate regular routers to maintain ephemeral FIB entries, each of them contains a unique reverse path identifier, to ensure the reachability to producers. If there are so many producers who wish to enjoy anonymity, the FIB size on each regular router might exceed its capability. To solve this issue, producers can leverage aggregatable reverse path identifiers by appending topological prefixes to the reverse path identifiers : $/\langle\text{topological-prefix}\rangle/rID$, where $/\langle\text{topological-prefix}\rangle$ denotes (maybe hierarchical) name prefixes which represent topological information of producers. However, the topological information should be carefully controlled because the use of such prefixes can degrade anonymity.

D. Descriptor Publication/Retrieval

In order to advertise the existence, the producer uploads the descriptor to several descriptor directories in the same way as the rendezvous point establishment phase. The descriptor is used by consumers to find the established rendezvous point corresponding to the onion name of their interests. Concretely, the descriptor is a type of content generated by the producer containing the producer's public key certificate $\mathsf{Cert}(pk_{id})$, the routable name of the rendezvous point, its public key certificate, and the signature $\sigma_{sk_{id}}$. We assume that the selection of responsible descriptor directories follows previous studies on hidden service [14], [29]. In short, the descriptor directories are managed by a scheme based on a distributed hash table (DHT) and the responsible directories are determined by the content name of the descriptor (called *descriptor name*) and current timestamp. Similar to onion names, descriptor names are derived as follows:

$$/\mathsf{onion}/H(pk_{id})/\mathsf{descriptor}.$$

A consumer who learns the onion name derives the descriptor name, finds the responsible descriptor directories determined by the descriptor name, and downloads the descriptor from one of them. The consumer accepts the descriptor only if $\sigma_{sk_{id}}$ is valid for pk_{id} obtained from $\mathsf{Cert}(pk_{id})$ in the descriptor.

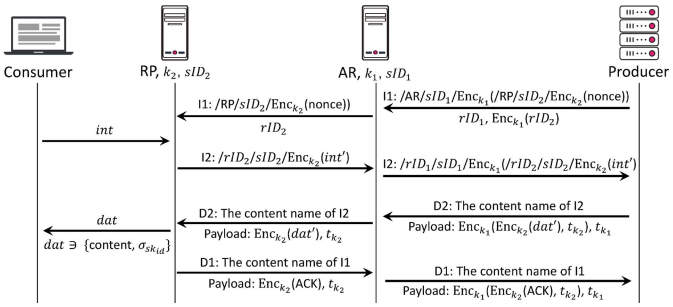

Fig. 4. Anonymous content publication.

E. Content Publication

After uploading the descriptor, the producer waits for content requests from consumers. Because reverse paths expire after a certain amount of time has elapsed, the producer updates them by issuing I1 packets carrying nonces to RP. These I1 packets also have the role of keeping the circuit alive by sending packets periodically, similar to PADDING cells in Tor [30]. RP waits for content requests for a certain time period T determined according to the reverse path expiry time and RTT between the producer and RP. If no content request from consumers has arrived within T, RP returns a Data packet to the producer to inform that there is no request. Suppose that the expiry time of FIB entries on reverse paths is set to t_{FIB}, then $T \leq t_{FIB} - RTT$ should hold, where RTT is the estimated RTT between the producer and RP.

Fig. 4 illustrates the flow of Interest/Data packets in the case where there is a content request from a consumer. Since the onion name is not routable, the consumer issues an Interest packet int requesting content through RP by appending its routable name as the content name prefix. For example, int carries the content name /RP/onion/encode(pk_{id})/article/xyz/html. int can be satisfied by any cache on the regular routers between the consumer and RP because it is not encrypted. If int reaches RP without being satisfied by the caches, RP first removes the name component /RP from int and then forwards such a new Interest packet int' along the reverse path associated with the onion name specified in int'. In the reverse paths, int' is treated as an I2 packet. The corresponding Data packet dat' containing the requested content and $\sigma_{sk_{id}}$ is returned from the producer to RP as the D2 packet by using PIT entries. Then, RP generates a Data packet *dat* which has the same content name as int by encapsulating dat' (without encryption). After sending *dat* to the consumer, RP returns the D1 packet to acknowledge the D2 packet. Suppose that the PIT entries expiry time is set to t_{PIT}, then $T \leq t_{PIT} - 2RTT$ should also hold to transport the D1 packet to the producer.

The consumer verifies that *dat* has certainly been generated by the intended producer advertising the onion name by verifying $\sigma_{sk_{id}}$ in *dat* with the public key *pk* corresponding to the onion name of her/his interest. In addition, each anonymizing router cannot learn its predecessor in these Interest/Data packet exchanges for the same reason as the rendezvous point establishment phase.

V. Anonymity Analysis

In this section, we provide an analysis of our system on producer anonymity against 1) $\mathcal{A}dv$ who just observes bit patterns of packets passing through compromised entities; and 2) $\mathcal{A}dv$ who also observes other sources of information, such as timing and volume of packets. We call the former a *weak adversary*, denoted as $\mathcal{A}dv^w$, and the latter a *strong adversary*, denoted as $\mathcal{A}dv^s$, respectively ($\mathcal{A}dv \in \{\mathcal{A}dv^w, \mathcal{A}dv^s\}$).

A. Notation

In the following discussion, we focus on the content publication phase described in Section IV-E because producer anonymity is achieved in other phases in the same way. In terms of linkability of packets, I1, I2, D1, and D2 packets traverse the same route by using PIT or reverse paths and are easily linkable by observing their content names and the reverse path identifiers. Consequently, in terms of producer anonymity, it is sufficient to focus on linkability between producers and one of these packets. In addition, I2 and D2 packets between a RP and a producer are just encrypted forms of Interest and Data packets between a consumer and the PR, respectively. From these observations, we hereinafter focus only on linkability between a producer and a Data packet.

Let $\mathcal{E}_k$ denote an operation in which a Data packet is encrypted once by using an encryption algorithm Enc with a secret key $k \leftarrow \mathsf{Gen}(\kappa)$ and a reverse path identifier and a session identifier chosen uniformly and independently at random from $\{0,1\}^\kappa$ are appended to the content name as described in Section IV. We represent a Data packet *dat* which has gone through $\mathcal{E}$ for a sequence of l secret keys $\mathcal{K}_l = (k_1, \ldots, k_l)$ (if $l = 0$, then $\mathcal{K}_l = \emptyset$) in this order as follows:

$$dat_{\mathcal{K}_l} = \begin{cases} dat & (l = 0) \\ \mathcal{E}_{k_l}\Big(\mathcal{E}_{k_{l-1}}\big(\cdots\big(\mathcal{E}_{k_1}(dat)\big)\cdots\big)\Big) & (l \geq 1) \end{cases}$$

Obviously, producer anonymity is broken if $\mathcal{A}dv$ can correctly correlate an outgoing Data packet at the producer, i.e., $\mathcal{E}_{k_n}(\mathcal{E}_{k_{n-1}}(\cdots(\mathcal{E}_{k_1}(dat))\cdots))$, and an outgoing Data packet at a rendezvous point, i.e., *dat*, because it implies that $\mathcal{A}dv$ can correlate the input and output of a circuit. For the sake of simplicity of notation, we define that the circuit is said to be compromised by $\mathcal{A}dv$ in such a case.

In our system, the sender of each Data packet might be included in an anonymity set, i.e., there might exist several possible senders, from the viewpoint of $\mathcal{A}dv$. This is because Data packets do not carry any identifiers of their senders and we assume that $\mathcal{A}dv$ does not compromise all the entities. Note that the term "sender" does not always correspond to the producer; for example, if an anonymizing router forwards a Data packet after removing the top layer of encryption, its sender is the anonymizing router. We define a *sender anonymity set* of a Data packet $dat_{\mathcal{K}_l}$ with respect to $\mathcal{A}dv$ as follows.

Definition 4 (Sender Anonymity Set):

$$\mathsf{AS}_{\mathcal{A}dv}^{dat_{\mathcal{K}_l}} = \{e \in \mathbb{P} \cup \mathbb{A} \mid \mathsf{Pr}[\mathcal{A}dv \text{ infers that } e \text{ has sent } dat_{\mathcal{K}_l} \mid \mathcal{A}dv \text{ observes } dat_{\mathcal{K}_l}] > 0\}.$$

This implies that the sender anonymity set of $dat_{\mathcal{K}_l}$ with respect to $\mathcal{A}dv$ contains all the entities which seem to have

sent it with non-zero probability from the perspective of $\mathcal{A}dv$. When the producer sends a Data packet, $\mathcal{A}dv$ on the first-hop regular router can identify the producer (i.e., the sender), however, the sender anonymity set will grow as it is transported toward regular routers on core networks because packets from more senders can pass through them. We assume that the anonymity degree of all the possible senders of $dat_{\mathcal{K}_l}$ equals $|\mathsf{AS}_{\mathcal{A}dv}^{dat_{\mathcal{K}_l}}|^{-1}$, where $|\cdot|$ represents the size of a set.

B. Anonymity Against $\mathcal{A}dv^w$

Since any CCA-secure encryption scheme is used and reverse path identifiers and session identifiers are chosen uniformly and independently at random in our system, the following theorem holds.

Theorem 1: $\mathcal{A}dv^w$ can correctly correlate incoming Data packets from non-compromised producers with the outgoing counterparts at a non-compromised anonymizing router only with negligible probability.

We provide the proof of Theorem 1 in the Appendix.

Next, we show the requirement to achieve producer anonymity in our system.

Theorem 2: $p \in \mathbb{P} \setminus \mathbb{P}_{\mathcal{A}dv^w}$ has producer anonymity in a configuration CF with respect to $\mathcal{A}dv^w$ if $\exists p' \in \mathbb{P} \setminus \mathbb{P}_{\mathcal{A}dv^w}$ such that $\mathsf{CF}_{\mathbb{D}}(p') \neq \mathsf{CF}_{\mathbb{D}}(p)$, $p' \neq p$, and any of the following conditions holds:

1) $\mathsf{CF}_{\mathbb{A}}(p) = \mathsf{CF}_{\mathbb{A}}(p') = \perp$.
2) $p, p' \in \mathsf{AS}_{\mathcal{A}dv^w}^{\mathsf{CF}_{\mathbb{D}}(p)_{\mathcal{K}_n}}$.
3) $\exists i \in \{1, \ldots, n\}$, $\mathsf{CF}_{\mathbb{A}_i}(p) = \mathsf{CF}_{\mathbb{A}_i}(p') \in \mathbb{A} \setminus \mathbb{A}_{\mathcal{A}dv^w}$.
4) $\exists i \in \{1, \ldots, n\}$, $\mathsf{CF}_{\mathbb{A}_i}(p), \mathsf{CF}_{\mathbb{A}_i}(p') \in \mathbb{A} \setminus \mathbb{A}_{\mathcal{A}dv^w}$ and $\mathsf{CF}_{\mathbb{A}_i}(p), \mathsf{CF}_{\mathbb{A}_i}(p') \in \mathsf{AS}_{\mathcal{A}dv^w}^{\mathsf{CF}_{\mathbb{D}}(p)_{\mathcal{K}_{n-i}}}$.

In other words, p can anonymously publish her/his Data packet if there exists another producer p' who publishes another Data packet and any of the following conditions holds: 1) the Data packets of p and p' are returned from caches; 2) p and p' are included in the same sender anonymity set of the encrypted Data packets from p and p' with respect to $\mathcal{A}dv^w$; 3) p and p' share the same i-th anonymizing router which is not compromised by $\mathcal{A}dv^w$; or 4) The i-th anonymizing routers of p and p' are not compromised by $\mathcal{A}dv^w$ and these anonymizing routers are included in the same sender anonymity set of their output Data packets with respect to $\mathcal{A}dv^w$.

Theorem 2 is also proven in the Appendix.

In hidden service, receiver anonymity is achieved against $\mathcal{A}dv^w$ only if a circuit includes at least one non-compromised anonymizing router. Similarly, our system achieves producer anonymity in such a case as described in the third condition. The difference between our system and hidden service is that our system can achieve anonymity even if all the anonymizing routers in a circuit are compromised. This is because a producer can be included in a sender anonymity set with respect to $\mathcal{A}dv^w$ who does not compromise the producer's first-hop regular router. This corresponds to the second condition. Putting together the second and the third condition, our system achieves a level of anonymity with one fewer anonymizing router than hidden service thanks to the first-hop regular router providing anonymity at the network layer, instead of an anonymizing router. The fourth condition is similar to the second condition. In contrast to the fact that each anonymizing router can learn its predecessor from the source address in a received packet in hidden service, each anonymizing router cannot learn its predecessor form a Data packet in our system. Thus, the fourth condition implies that it is more difficult for $\mathcal{A}dv^w$ to compromise a circuit in our system than in hidden service. The first condition implies that producer anonymity can be achieved by leveraging in-network caching. This is because producers do not send/receive any packets when cache hits occur.

The inverse of Theorem 2 also holds: producer p does not have producer anonymity if none of the conditions of Theorem 2 hold, i.e., Data packet *dat* of p is not returned from caches and $\mathcal{A}dv^w$ compromises the first-hop regular router of p and all the anonymizing routers in p's circuit. This is because it implies that $\mathcal{A}dv$ can track *dat* throughout the circuit, i.e., the circuit is compromised by $\mathcal{A}dv^w$. Let $f_{\mathbb{A}} = |\mathbb{A}_{\mathcal{A}dv}|/|\mathbb{A}|$ and $f_{\mathbb{R}} = |\mathbb{R}_{\mathcal{A}dv}|/|\mathbb{R}|$ ($0 \leq f_{\mathbb{A}}, f_{\mathbb{R}} < 1$, $\mathcal{A}dv \in \{\mathcal{A}dv^w, \mathcal{A}dv^s\}$), i.e., $f_{\mathbb{A}}$ and $f_{\mathbb{R}}$ represent the fractions of compromised entities in the sets of all the anonymizing routers and the regular routers, respectively. Hereinafter, we suppose that each anonymizing router in a circuit is compromised independently with probability $f_{\mathbb{A}}$. This gives a good approximation in the realistic model, in which a large number of anonymizing routers exist (e.g., $|\mathbb{A}| \approx 6000$ in current Tor [31]). Then, $\mathcal{A}dv^w$ can break producer anonymity with probability

$$(1 - p_c) \cdot f_{\mathbb{R}} \cdot f_{\mathbb{A}}^n, \tag{1}$$

where p_c is the probability that a cache hit occurs ($0 \leq p_c \leq 1$). Intuitively, larger n, i.e., longer circuits, contributes to achieve producer anonymity with more confidence.

C. Anonymity Against $\mathcal{A}dv^s$

$\mathcal{A}dv^s$ can launch more sophisticated attacks called *traffic analysis attacks* [16], [32], [33], [34], [35] by using other sources of information, such as timing and volume of packets. In particular, we focus on *traffic confirmation attacks*, which are a type of traffic analysis attacks aiming to correlate two entities included in the same circuit. To mitigate traffic confirmation attacks, several schemes like packet batching and reordering have been proposed [35], [36], however, they are not implemented in Tor and hidden service due to their high cost in terms of the delay and the load at each anonymizing router. Thus, our system does not explicitly employ such schemes, whereas we believe that they could also be incorporated into our system almost without modification.

In this subsection, rather than discussing how to deal with traffic confirmation attacks, we analyze the probability that producer anonymity is broken with the predecessor attack [16], [17], assuming that $\mathcal{A}dv^s$ launches successful traffic confirmation attacks. The predecessor attack is a substantial threat to anonymity systems based on onion routing because it is difficult to detect, relatively easy to launch, and the probability of its success increases to 1.0 with time [16]. With the predecessor attack, $\mathcal{A}dv^s$ can break producer anonymity even when $\mathcal{A}dv^w$ cannot.

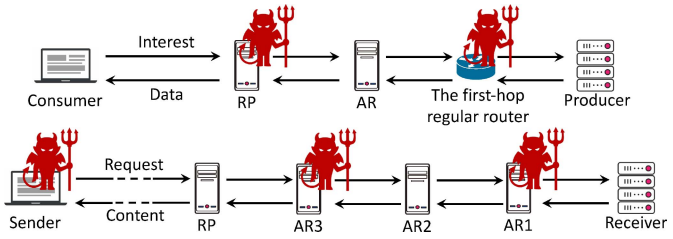

Fig. 5. The cases in which receiver anonymity and producer anonymity are broken in hidden service and in our system with traffic confirmation attacks.

1) Predecessor Attack in Static Model: Before describing the predecessor attack, we describe the cases in which producer anonymity and receiver anonymity are broken with traffic confirmation attacks and show the probabilities of their occurrence in a single round. Hereinafter, we assume that traffic confirmation attacks always succeeds when two entities on the same circuit are compromised by $\mathcal{A}dv^s$. In the current implementation of hidden service, each circuit of receivers includes three anonymizing routers by default. To match the level of anonymity against $\mathcal{A}dv^w$, we assume that two anonymizing routers are included in each circuit in our system.

The upper illustration in Fig. 5 depicts the case in which anonymity of a producer is broken in our system. AR and RP are an anonymizing router and a rendezvous point included in the producer's circuit, respectively. Hereinafter, we focus only on the case in which Interest packets are not satisfied by caches because $\mathcal{A}dv^s$ cannot launch traffic confirmation attacks if cache hits occur. In our system, producer anonymity is broken with traffic confirmation attacks if both the first-hop regular router and the rendezvous point of the producer's circuit are compromised by $\mathcal{A}dv^s$. This is because it implies that $\mathcal{A}dv^s$ can correlate the producer's MAC address obtained at the first-hop regular router and plaintext Data packets outputted by the rendezvous point. Note that $\mathcal{A}dv^s$ can make sure that s/he has compromised a rendezvous point among anonymizing routers because packets are plaintext only between a consumer and a rendezvous point. Similarly, $\mathcal{A}dv^s$ can make sure that a compromised regular router is the first-hop of a producer by checking whether the packets received on the regular router are the same as those received on the compromised rendezvous point. If not the same, there is an anonymizing router between the regular router and the rendezvous point, and thus the regular router is the first-hop of a producer. Thus, producer anonymity is broken in our system with probability $f_{\mathbb{R}} \cdot f_{\mathbb{A}}$.

Similarly, the lower illustration in Fig. 5 depicts one of the cases in which anonymity of a receiver is broken in hidden service. Note that the last-hop anonymizing router of a receiver's circuit is not the rendezvous point in hidden service. Instead, the last-hop anonymizing router of a sender's circuit is the rendezvous point. Traffic confirmation attacks against hidden service looks a little more complicated due to end-to-end encryption between senders and receivers: $\mathcal{A}dv^s$ requires to act as a sender to observe packets from a receiver in plaintext. However, this requirement is easily satisfied because anyone can act as a sender, and thus we ignore the probability that a sender is compromised, hereinafter. The goal of $\mathcal{A}dv^s$ acting as a sender is to learn a receiver's IP address at the first-hop anonymizing router. Even if $\mathcal{A}dv^s$ finds that a compromised anonymizing router is included in a receiver's circuit, however, $\mathcal{A}dv^s$ cannot confirm its position in the circuit. Thus, in addition to the first-hop anonymizing router (AR1), $\mathcal{A}dv^s$ must compromise another anonymizing router in the same circuit (AR2 or AR3). Then, $\mathcal{A}dv^s$ makes sure of their positions in a circuit based on the IP addresses of the compromised anonymizing routers and the rendezvous point known to the sender (i.e., $\mathcal{A}dv^s$). For example, if $\mathcal{A}dv^s$ learns that two compromised anonymizing routers in the same circuit are not adjacent to each other and one of them is adjacent to the rendezvous point, then these anonymizing routers are AR1 and AR3. Therefore, receiver anonymity is broken in hidden service with probability $f_{\mathbb{A}} \cdot (1 - (1 - f_{\mathbb{A}})^2) = 2f_{\mathbb{A}}^2 - f_{\mathbb{A}}^3$.

In the predecessor attack, $\mathcal{A}dv^s$ just acts like legitimate entities, i.e., follows the prescribed protocols for the compromised entities, and continually performs a succession of traffic confirmation attacks against the packets passing through them [16]. In this subsection, we consider the predecessor attack in the static model, where the entities in $\mathbb{A}$ and $\mathbb{R}$ do not change throughout rounds. In each round, a circuit is compromised by $\mathcal{A}dv^s$ in hidden service and in our system with probabilities of $2f_{\mathbb{A}}^2 - f_{\mathbb{A}}^3$ and $f_{\mathbb{R}} \cdot f_{\mathbb{A}}$, respectively as described above. These are the lower bound of the probabilities of a circuit being compromised with the predecessor attack. Comparing to Eq. (1), we can see that the predecessor attack has a greater probability of success in many cases. Because receiver and producer anonymity are broken if one circuit is compromised, we focus on the probability that at least one circuit is compromised by $\mathcal{A}dv^s$ in m rounds, hereinafter.

If all the anonymizing routers in circuits are chosen uniformly at random in each round, the probability that at least one circuit of a receiver is compromised in m rounds in hidden service is derived as follows:

$$1 - \left(1 - 2f_{\mathbb{A}}^2 + f_{\mathbb{A}}^3\right)^m. \tag{2}$$

As the number of rounds increases ($m \to \infty$), the probability grows to 1.0.

To mitigate the predecessor attack, entry guards are introduced to hidden service. Because a receiver repeatedly uses the first-hop anonymizing router called an entry guard, $\mathcal{A}dv^s$ must compromise it in addition to either the second-hop or the third-hop anonymizing router. The probability that neither the second-hop nor the third-hop anonymizing router is compromised even once in m rounds is $(1 - f_{\mathbb{A}})^{2m}$. Therefore, the probability that at least one circuit of a receiver is compromised is derived as follows:

$$f_{\mathbb{A}} \cdot \left\{1 - (1 - f_{\mathbb{A}})^{2m}\right\}. \tag{3}$$

As the number of rounds increases ($m \to \infty$), the probability grows to $f_{\mathbb{A}}$ (< 1).

In our system, the last-hop anonymizing router in circuits of a producer, i.e., a rendezvous point, is fixed. Such an anonymizing router is called an exit guard. In addition, the first-hop regular router plays the role of an entry guard because the producer's identities, such as MAC addresses, are revealed

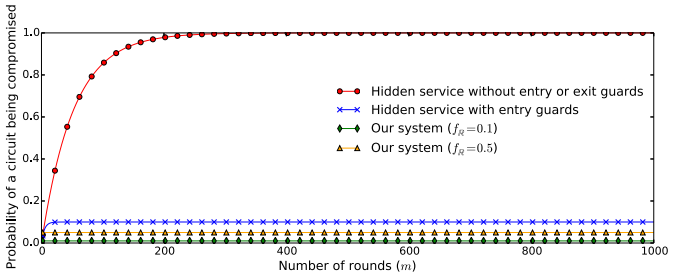

Fig. 6. The probability that at least one circuit is compromised with the predecessor attack in the static model ($f_{\mathbb{A}} = 0.1$).

only to it. Thus, $\mathcal{A}dv^s$ must compromise both of them to compromise a circuit. Therefore, the probability that at least one circuit of a producer is compromised is derived as follows:

$$f_{\mathbb{R}} \cdot f_{\mathbb{A}}, \tag{4}$$

regardless of the number of rounds ($f_{\mathbb{R}} \cdot f_{\mathbb{A}} < f_{\mathbb{A}} < 1.0$). This probability is equal to the lower bound of probability of a circuit being compromised by the predecessor attack in the static model.

Fig. 6 shows the changes in Eq. (2), Eq. (3), and Eq. (4) as the number of rounds increases when $f_{\mathbb{A}} = 0.1$. It is shown that our system offers the best security even if $f_{\mathbb{R}}$ is much larger than $f_{\mathbb{A}}$.

2) Predecessor Attack in Dynamic Model: In this subsection, we consider the predecessor attack in the dynamic model, where the members in $\mathbb{A}$ and $\mathbb{R}$ change over rounds. As distinct from the static model, changes of an entry and an exit guard are caused if one of them becomes unavailable, and this gives $\mathcal{A}dv^s$ further opportunities to compromise circuits in addition to those in the static model. The dynamic model is worth considering because it has been shown that only about half of the entry guards in hidden service remain available for the intended period of time (e.g., 720-1440 hours), and thus changes of guards occur when they are undesirable [37]. Since anonymizing routers are operated by unreliable volunteers, we believe that this is a inherent problem for hidden service and our system.

It is possible to use both entry and exit guards in hidden service to improve security against the predecessor attack in the static model. However, this causes a problem in the dynamic model because the receiver changes entry and exit guards if one of them becomes unavailable, and thus changes of guards are presumed to occur more frequently in the case where both entry and exit guards are used than in the case where only entry guards are used [17]. Although our system employs both entry and exit guards, our system mitigates this problem by having the first-hop regular routers act as entry guards instead of the first-hop anonymizing routers. Because (carrier-grade) regular routers managed by ISPs are intended to provide higher availability, e.g., the five nines available requirement [38], than anonymizing routers operated by voluntary hosts, changes of guards are affected almost exclusively by the availability of anonymizing routers chosen as exit guards. This implies that our system provides a degree of security against the predecessor attack in the dynamic model that is equivalent to that provided by hidden service, in which only entry guards are employed.

Hereinafter, we compare the case where anonymizing routers are used as an entry and an exit guard in hidden service with our system, where a regular router and an anonymizing router are used, in terms of the probability that at least one circuit is compromised by $\mathcal{A}dv^s$ in m rounds. Our goal is to show how the probability decreases by substituting regular routers for anonymizing routers. We assume that each anonymizing router becomes unavailable in $\mathbb{A}$ independently with probability q ($0 < q < 1$) at the end of each round. Because we are interested in how the change in q affects the probability of a circuit being compromised, we assume $f = f_{\mathbb{A}} = f_{\mathbb{R}}$ ($0 \leq f < 1$), i.e., the same fraction of anonymizing routers and regular routers are compromised by $\mathcal{A}dv$.

When anonymizing routers are used as both entry and exit guards in hidden service, the probability that the producer changes guards i times in m rounds is $\{1-(1-q)^2\}^i\{(1-q)^2\}^{m-i}\binom{m}{i}$. For i changes of guards, the probability that $\mathcal{A}dv^s$ succeeds in compromising at least one circuit of the producer is $1-(1-2f^2+f^3)^i$. Thus, the probability that $\mathcal{A}dv^s$ succeeds in compromising at least one circuit of the producer in m rounds is derived as follows:

$$\sum_{i=0}^{m}\left\{1-\left(1-2f^2+f^3\right)^i\right\}\left\{1-(1-q)^2\right\}^i \times\left\{(1-q)^2\right\}^{m-i}\binom{m}{i}. \tag{5}$$

Next, we consider our system, in which regular routers and anonymizing routers are used as entry and exit guards, respectively. We assume that the probability that each regular router becomes unavailable in $\mathbb{R}$ at the end of each round is sufficiently close to 0.0. Therefore, in our system, the probability that the producer changes guards i times in m rounds is $q^i(1-q)^{m-i}\binom{m}{i}$. For i changes of guards, the probability that $\mathcal{A}dv^s$ succeeds in compromising at least one circuit of the producer is $1-(1-f^2)^i$ since $\mathcal{A}dv^s$ must compromise the first-hop regular router and the last-hop anonymizing router. Thus, the probability that $\mathcal{A}dv^s$ succeeds in compromising at least one circuit of the producer in m rounds is derived as follows:

$$\sum_{i=0}^{m}\left\{1-\left(1-f^2\right)^i\right\}q^i(1-q)^{m-i}\binom{m}{i}. \tag{6}$$

Fig. 7 shows the change in Eq. (5) and Eq. (6) for several pairs of f and q. It is shown that the probabilities of a circuit being compromised are sufficiently small in our system for all the pairs of f and q and the differences increase as q increases.

From the analysis of the both models, the security of our system against the predecessor attack can be summarized as follows: First, our system provides the better security in the static model because both entry and exit guards are used. Second, our system provides the security comparable to that of hidden service in the dynamic model due to the use of regular routers as entry guards. Because $\mathcal{A}dv^s$ can launch the predecessor attack in the static model and the dynamic model

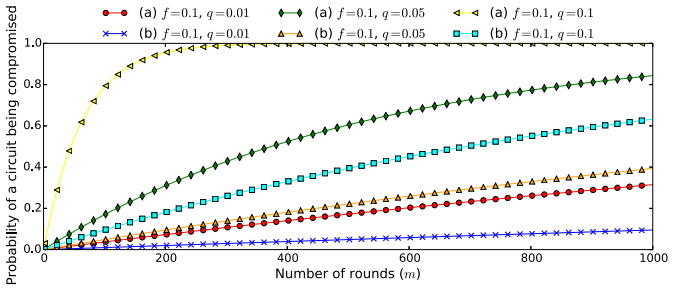

Fig. 7. Probabilities that at least one circuit is compromised with the predecessor attack in the dynamic model. (a) represents the case where both entry and exit guards are used in hidden service (Eq. (5)), and (b) represents our system (Eq. (6)).

at the same time, we conclude that our system provides better security against the predecessor attack than hidden service.

VI. Evaluation

In this section, we first evaluate the performance of our system compared to hidden service in terms of RTT, defined as the time it takes for a consumer to send a request and then receive the corresponding the content, and throughput by implementing prototypes with minimum functions. We compare our system only with hidden service because no other anonymity systems are proposed for NDN which assume the same adversarial model as our system as described in Section VII. Then, we assess the probability of the successful predecessor attack assuming $\mathcal{A}dv^s$ with the knowledge of the underlying network topology to address an issue arisen in realistic networks.

A. Implementation and Performance

In this subsection, we focus only on the content publication phase because the same style of communication is used in the other phases and they have little effect on producers' long-term activities thanks to the fact that they are performed only at the first set up time. In addition, we assume that Interest packets issued by a consumer are forwarded to a producer without being satisfied by any intermediate router's cache. This is the worst case scenario in terms of RTT and throughput.

We implemented our system as applications that run on producers and anonymizing routers (including rendezvous points) by using the ndn-cxx library, which is a C++ library implementing NDN primitives. These applications implement the functions required in the content publication phase, such as encryption and decryption of packets, described in Section IV-E. We used AES-128 as a secret key encryption/decryption algorithm and HMAC with SHA-256 as a message authentication code generation/verification algorithm. These cryptographic functions were implemented by using OpenSSL. To compare our system with hidden service, we also implemented simple hidden service applications which work as receivers and anonymizing routers because the current hidden service implementation includes many functions not required for comparison. In hidden service, we assume that a sender issues an IP packet requesting content through a circuit built by a receiver without using a circuit between a sender and a rendezvous point so that the only receiver anonymity comparable to producer anonymity is provided.

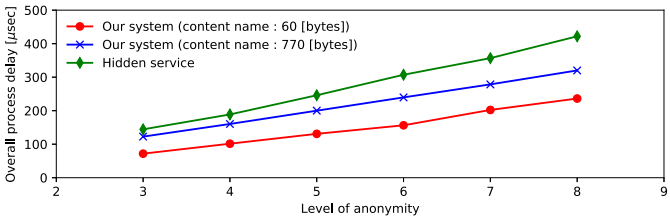

Fig. 8. Process delay measurement.

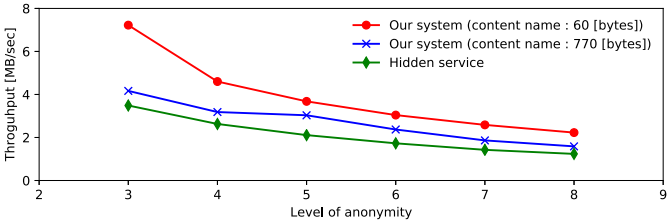

Fig. 9. Throughput measurement.

We focus mainly on the process delay taken by the applications to derive RTT. This is because current NDN runs as a application on top of TCP/IP, and thus an unavoidable overhead will be incurred in our system, which is implemented over NDN, compared to hidden service, which is implemented over TCP/IP, when producers/receivers and anonymizing routers communicate with each other as described in [10], [39]. Thus, we first show the overall process delay taken by the applications of our system and hidden service, and then show the RTT for various average end-to-end delay in communication between each pair of entities assuming a simple line topology. The overall process delay is defined as the total time it takes for the applications to process a packet by using cryptographic functions, not including the time it take to transport the packet between them. All experiments were conducted on a machine with an Intel Xeon E5-2620 v4 processor (2.10 GHz) with eight DDR4 16GB DRAM devices. The operating system on the machine is Ubuntu 18.04 LTS.

Fig. 8 shows the overall process delay as a function of the achieved level of anonymity against $\mathcal{A}dv^w$, and similarly Fig. 9 shows the throughput of the applications. Since a level of anonymity represents the number of anonymizing routers and regular routers $\mathcal{A}dv$ must compromise to trace packets throughput a circuit, for example, if the level of anonymity is three, it means that three anonymizing routers are used in hidden service and that two anonymizing router is used in our system. Because the size of application data is set to 512 bytes in the current hidden service implementation, we set the size of each Data packet payload to the same size. Regarding the size of content name in Interest/Data packets, Ghali *et al.* have generated realistic NDN compatible names according to the URLs in the Unibas dataset from The Content Name Collection [20] and found that the average and the maximum size of the names are approximately 60 bytes and 770 bytes, respectively. According to this result, we evaluated the cases where content names are set to these sizes.

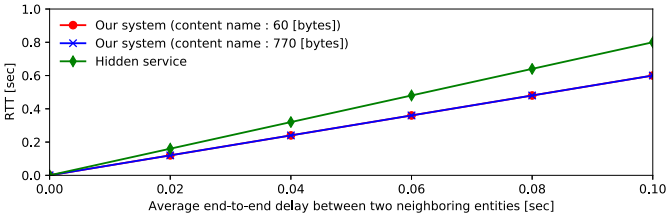

Fig. 10. RTT measurement (level of anonymity = 3).

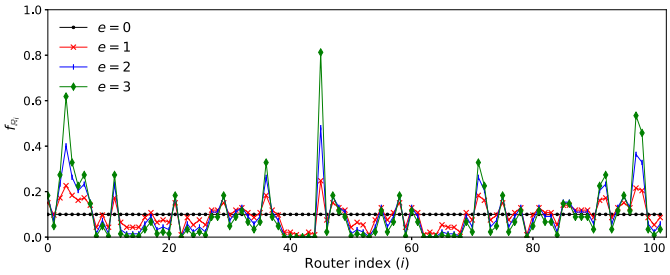

Fig. 11. Probability that the *i*-th first-hop regular router is compromised.

As expected, our system has better performance (i.e., less process delay and more throughput) when content names are shorter. This is mainly because the smaller content names result in shorter process delay for performing cryptographic functions and packet generation. This implies that producers can improve efficiency of content publishing by carefully naming their content so that the content names are small in our system. In comparison with hidden service, our system has better performance because our system reduces the number of required anonymizing routers in a circuit by one while still achieving a comparable level of anonymity to hidden service. More specifically, our system reduces the number of cryptographic operations performed by a producer compared to those performed by a receiver in hidden service, in addition to that an anonymizing router becomes unnecessary.

Fig. 10 shows how end-to-end delay between neighboring entities influences the RTT. We assumed that the level of anonymity is three, i.e., hidden service uses three anonymizing routers and our system uses two anonymizing routers. It is shown that the overhead due to the process delay is sufficiently small as we can see that RTT approximately equals to 0.0 when the end-to-end delay equals to 0.0. Thus, we conclude that RTT is predominantly determined by end-to-end delay required to transport packets around geographically distributed anonymizing routers. Therefore, we argue that such overhead is unavoidable to achieve producer anonymity. However, it is also shown that our system sufficiently prunes such overhead ($0.05 \sim 0.2$ [sec]) by reducing the number of anonymizing routers by one.

B. Anonymity Analysis Under a Realistic Network Topology

We have analyzed the security of our system against the predecessor attack launched by $\mathcal{A}dv^s$ in Section V-C, assuming the adversarial model where $\mathcal{A}dv^s$ chooses regular routers to compromise uniformly at random. In this subsection, we analyze the security against the predecessor attack against $\mathcal{A}dv^s$ who chooses anonymizing routers to compromise taking the underlying network topology into account. Because the security of our system depends on the first-hop regular routers of producers, $\mathcal{A}dv^s$ can break anonymity of many producers if it preferentially compromises regular routers connecting to many ones. Thus, the expected number of such producers can depend on an underlying network topology. To confirm this, we evaluate the expected value of the number of producers being broken anonymity with the predecessor attack in the static model based on the real network topology and population in Tokyo, Japan.

The target area is a 32 km square part of Tokyo. We construct a router-level topology defined as a tree of depth 3 based on the positions and coverage areas of telephone exchange buildings of NTT East Corporation [40]. The first-level regular router is placed in Otemachi, which is one of the most thickly populated areas in Japan. The second level regular routers are placed near the 6 terminal stations on the Yamanote line. The third-level regular routers are placed so that the target area is uniformly covered and connected to the closest second level regular routers. The fourth level regular routers connects to the hosts within some specific municipalities, and thus plays the role of the first-hop regular routers of producers. We call the fourth-level regular routers the first-hop regular routers, hereinafter. The number of the first-hop regular routers is $M = 102$. We assume that the number of producers connected to each of the first-hop regular routers is proportional to the population within its coverage area, and the number of all producers is 927 (1/10000 of the total population in the target area).

Let n_i and $f_{\mathbb{R}_i}$ denote the number of producers connecting to the *i*-th first-hop regular router and the probability of the *i*-th first-hop regular router being compromised, respectively. We set $f_{\mathbb{R}_i}$ to be proportional to $n_i^e / \sum_{j=1}^{M} n_j^e$ so that first-hop regular routers with a larger number of producers are more likely to be compromised. e is a constant that determines how much priority is given to the regular routers with a larger number of producers. We assume that $\mathcal{A}dv^s$ compromises ten of the first-hop regular routers (i.e., approximately 10% of all the first-hop regular routers), and thus $f_{\mathbb{R}_i} = 0.1$ for all the first-hop regular routers when $e = 0$. By setting $f_{\mathbb{R}_i}$ so that $\sum_{i=1}^{M} f_{\mathbb{R}_i}$ is constant for $e \geq 1$, the larger e becomes, the larger $f_{\mathbb{R}_i}$ for the first-hop regular routers with larger n_i becomes and the smaller $f_{\mathbb{R}_i}$ for those with smaller n_i becomes. Fig. 11 shows the values of $f_{\mathbb{R}_i}$ when $e = \{0, 1, 2, 3\}$. For example, the 45-th first-hop regular router is compromised with higher probability than others when $e \geq 1$ because it connects to more producers.

As described in Section V-B, the probability that anonymity of each producer connected to the *i*-th first-hop regular router is broken with the predecessor attack in the static model is $f_{\mathbb{R}_i} \cdot f_{\mathbb{A}}$, where $f_{\mathbb{A}}$ is the probability of each anonymizing routers being compromised defined in Section V-B. Thus, the expected value of the number of producers of which anonymity is broken among all the producers connected to the *i*-th first-hop regular router is $\sum_{j=1}^{n_j} j \cdot f_{\mathbb{R}_i} \cdot \binom{m}{i} \cdot f_{\mathbb{A}}^j \cdot (1 - f_{\mathbb{A}})^{n_i - j}$. By using linearity of expectation, the expected value of the total number of producers who are broken anonymity is computed

TABLE II
THE EXPECTED VALUE OF THE NUMBER OF PRODUCERS BEING BROKEN ANONYMITY AND ITS RATIO

	$e=0$	$e=1$	$e=2$	$e=3$
E	9.01	11.84	13.63	15.06
$ratio\ (=E/927)$	0.0097	0.0128	0.0147	0.0162

as follows:

$$E = \sum_{i=1}^{M} \sum_{j=1}^{n_j} j \cdot f_{\mathbb{R}_i} \cdot \binom{m}{i} \cdot f_{\mathbb{A}}^{j} \cdot (1 - f_{\mathbb{A}})^{n_i - j}. \tag{7}$$

The actual values of E and its ratio among all the producers are summarized in Table II. The result shows that $\mathcal{A}dv^s$ can break anonymity of more producers by selectively compromising the first-hop regular routers connecting to a large number of producers. Thus, it is important for producers who wish to enjoy anonymity to contract ISPs with a small number of hosts if $\mathcal{A}dv^s$ with the knowledge of the underlying network topology should be contemplated. However, if there are too few hosts in the same network, anonymity is difficult to achieve in principle. Therefore, producers should choose ISPs to contract while simultaneously considering these two factors.

VII. RELATED WORK

The design of onion routing is derived from Chaum's Mix-Net [41], which aims to hide who sends an e-mail to whom. In Mix-Net, e-mails are encapsulated in layers of public-key encryption and forwarded through several nodes called mixes. After decrypting received packets, each mix stores them for a certain period of time and forwards them in random order. Since we describe Tor, hidden service, and ANDāNA as existing onion routing-based systems in detail in Section II, we summarize other kinds of anonymity systems here. Note that producer anonymity is briefly mentioned in ANDāNA paper [10], however, we presented a stronger and more rigorous definition in this article. Intuitively, the definition of ANDāNA considers producer anonymity only against adversaries on anonymizing routers and regular routers, whereas our definition also considers producer anonymity against consumers.

Kaushik *et al.* have proposed an attribute-based signature scheme suitable for NDN, called NDN-ABS (NDN Attribute-Based Signature), to enable privacy-preserving signature verification [42]. With NDN-ABS, consumers cannot identify a single producer among a set of producers with the same attribute from a signature. However, NDN-ABS cannot provide producer anonymity against adversaries eavesdropping packets on the networks because it focuses only on the information leakage from signatures. In addition, the attribute authority, which managed producers' secret keys used to generate signatures, is a single point of failure in terms of anonymity.

Next, we summarize systems for consumer anonymity. Tourani *et al.* have proposed a lightweight mechanism to hide content names based on Huffman coding [43] and Kurihara *et al.* have proposed an approach employing consumer-driven access control on content names, while enabling content caching on trusted entities [44]. These mechanisms use trusted proxies called anonymizers to allow consumers to anonymously retrieve content. Arianfer *et al.* have proposed a covered content-based approach, in which producers mix censored content with other content based on keywords contained in it to prevent information leakage from content names [45].

Crowds is a P2P-based anonymity system without packet encapsulation in IP [46]. In Crowds, packets are sent around peers to hide their origins. Each peer probabilistically decides whether to forward received packets to the intended destinations or to relay them to other peers. Thus, adversaries far away from an initiator of a packet cannot determine whether a particular peer is the initiator or not. Inspired by Crowds, a similar system called CRISP is proposed for NDN to achieve consumer anonymity [9]. In CRISP, instead of peers, each regular router probabilistically determines whether to forward a received Interest packet toward the specified producer or toward another cooperative regular router. This process makes it difficult for adversaries to trace back an Interest packet to its origin (i.e., a consumer). However, these systems have a drawback that anonymity is broken if the first-hop entities, i.e., peers or regular routers, are compromised by adversaries because such adversaries can immediately learn who initiates a particular packet.

In Section V, we specifically focus on the traffic confirmation attacks among traffic analysis attacks [16], [32], [33], [34], [35]. To mitigate traffic analysis attacks, several schemes have been proposed. For example, Gulcu and Tsudik have proposed a batching and reordering scheme in which each anonymizing router stores arrival packets until the number of packets reaches a threshold and forwards them in random order [36]. Regarding padding schemes, Shmatikov and Wang, have proposed an adaptive padding scheme, in which each anonymizing router inserts dummy packets in original packet flows when it is difficult to prevent $\mathcal{A}dv$ from correlating two links using inter-packet time intervals [35]. These schemes can be leveraged in our system almost without modification, however, our system does not explicitly employ them due to their high cost. Rather, we have analyzed the security of our system under the assumption that $\mathcal{A}dv$ succeeds in launching traffic analysis attacks.

VIII. CONCLUSION AND FUTURE WORK

This article first defined producer anonymity in NDN in terms of content-producer unlinkability, and then designed a system to achieve producer anonymity based on hidden service. Concretely, we leveraged onion names and rendezvous points of hidden service to address the NDN issue that every content is inherently tied to it producer. Moreover, we improved hidden service in terms of efficiency and security by incorporating RICE in onion routing. Our experiments showed that our system definitely reduces RTT and improves throughput in content retrieval by reducing the number of anonymizing routers required to achieve a certain level of anonymity by one.

Our plans for future work include the implementation of all the phases of our system, such as the rendezvous point establishment phase and the descriptor publication/retrieval phase, and more performance evaluations under various scenarios, e.g., mobile wireless networks and congested networks. In addition, in terms of security of our system, integrating several DoS mitigation mechanisms into our system is also one of our future research plans. For instance, requiring producers to solve puzzles, which cost a lot of CPU cycles or memory before establishing reverse paths and circuits, can hinder adversaries from making regular routers and anonymizing routers unavailable by establishing many reverse paths and circuits through them.

APPENDIX
ANONYMITY PROOFS

Proof of Theorem 1: Suppose that two non-compromised producers $p, p' \in \mathbb{P} \setminus \mathbb{P}_{\mathcal{A}dv^w}$ independently exchange sequences of secret keys $\mathcal{K}_n = (k_1, \ldots, k_i, \ldots, k_n)$ and $\mathcal{K}'_n = (k'_1, \ldots, k'_i, \ldots, k'_n)$ with their chosen anonymizing routers to build circuits, and $a_i \in \mathbb{A} \setminus \mathbb{A}_{\mathcal{A}dv^w}$ is used as the *i*-th anonymizing router in both circuits. Two encrypted Data packets from p and p' received by a_i are denoted by $dat_{\mathcal{K}_{n-i+1}}$ and $dat_{\mathcal{K}'_{n-i+1}}$, respectively ($dat, dat' \in \mathbb{D}$). Suppose that $\mathcal{A}dv^w$ attempts to correlate these incoming Data packets with the corresponding outgoing Data packets, i.e., $dat_{\mathcal{K}_{n-i}}$ and $dat_{\mathcal{K}'_{n-i}}$. Suppose that an encryption algorithm Enc of any CCA-secure secret key encryption scheme Π is used in the encapsulation algorithm $\mathcal{E}$ defined in Section V-A. First, by the definition of CCA-secure encryption schemes, $\mathcal{A}dv^w$ can correctly correlate them by observing the changes in their bit patterns due to decryption at a_i only with negligible probability. This is because $\mathcal{A}dv^w$ who does not compromise a_i cannot learn secret keys k_i, k'_i. Second, in our system, reverse path identifiers and session identifiers, which are the unencrypted parts of content names on Data packets, can be considered as the other source of information to correlate Data packets. We assume that $dat_{\mathcal{K}_{n-i+1}}$ and $dat_{\mathcal{K}_{n-i}}$ carry reverse path identifiers rID_i, rID_{i-1} and session identifiers sID_i, sID_{i-1}, respectively. Similarly, $dat_{\mathcal{K}'_{n-i+1}}$ and $dat_{\mathcal{K}_{n-i}}$ are assumed to carry reverse path identifiers rID'_i, rID'_{i-1} and session identifiers sID'_i, sID'_{i-1}, respectively. $\mathcal{A}dv^w$ can correlate these Data packets if $\mathcal{A}dv^w$ can correlate any pair of these identifiers on the input and the output Data packet. However, it is infeasible for $\mathcal{A}dv^w$ because they are chosen uniformly and independently at random by p and p' from $\{0,1\}^\kappa$ as described in Section IV-C. ∎

Proof of Theorem 2 - (1): Without loss of generality, we assume CF such that $\mathsf{CF}(p) = (\perp, c, dat)$ and $\mathsf{CF}(p') = (\perp, c', dat')$, and $\mathcal{A}dv^w$ such that $\mathbb{P}_{\mathcal{A}dv^w} = \mathbb{P} \setminus \{p, p'\}$, $\mathbb{A}_{\mathcal{A}dv^w} = \mathbb{A}$, $\mathbb{R}_{\mathcal{A}dv^w} = \mathbb{R}$, and $\mathbb{C}_{\mathcal{A}dv^w} = \mathbb{C}$, i.e., $\mathcal{A}dv^w$ compromises all the producers except for p and p' and all the anonymizing routers, regular routers, and consumers. This CF satisfies the first condition in Theorem 2. In this case, *dat* and dat' are published without using circuits. Supposing another configuration CF' which is identical to CF except that $\mathsf{CF}(p) = (\perp, c', dat')$ and $\mathsf{CF}(p') = (\perp, c, dat)$, the only sources of information to distinguish CF and CF' are onion names and signatures on *dat* and dat'. However, $\mathsf{CF}' \equiv_{\mathcal{A}dv^w} \mathsf{CF}$ holds because the onion names and signatures are generated from identity key pairs chosen independently and randomly by p and p'. Therefore, from the definition 3, p has producer anonymity. ∎

Proof of Theorem 2 - (2): Without loss of generality, we assume CF such that $\mathsf{CF}(p) = (a_1, \ldots, a_n, c, dat)$ and $\mathsf{CF}(p') = (a'_1, \ldots, a'_n, c', dat')$, and $\mathcal{A}dv^w$ such that $\mathbb{P}_{\mathcal{A}dv^w} = \mathbb{P} \setminus \{p, p'\}$, $\mathbb{A}_{\mathcal{A}dv^w} = \mathbb{A}$, and $\mathbb{C}_{\mathcal{A}dv^w} = \mathbb{C}$, i.e., $\mathcal{A}dv^w$ compromises all the producers except for p and p' and all the anonymizing routers and consumers. We also assume that $p, p' \in \mathsf{AS}^{dat_{\mathcal{K}_n}}_{\mathcal{A}dv^w}$, i.e., p and p' are included in the same sender anonymity set with respect to $\mathcal{A}dv^w$. This CF satisfies the second condition in Theorem 2. We suppose another configuration CF' which is identical to CF except that $\mathsf{CF}'(p) = (a'_1, \ldots, a'_n, c', dat')$ and $\mathsf{CF}'(p') = (a_1, \ldots, a_n, c, dat)$. In this case, $\mathcal{A}dv^w$ can track *dat* and dat' from a_1 to a_n and from a'_1 to a'_n for CF and CF' because all the anonymizing routers are compromised. Thus, $\mathcal{A}dv^w$ can distinguish CF from CF' only if $\mathcal{A}dv^w$ can correctly correlate the input Data packets at a_1 and a'_1 with p and p'. However, such $\mathcal{A}dv^w$ cannot exist from the definition of the sender anonymity set. Therefore, $\mathsf{CF}' \equiv_{\mathcal{A}dv^w} \mathsf{CF}$ holds for such CF', and p has producer anonymity. ∎

Proof of Theorem 2 - (3): Without loss of generality, we assume CF such that $\mathsf{CF}(p) = (a_1, \ldots, a_{i-1}, a_i, a_{i+1}, \ldots, a_n, c, dat)$ and $\mathsf{CF}(p') = (a'_1, \ldots, a'_{i-1}, a_i, a'_{i+1}, \ldots, a'_n, c', dat')$, and $\mathcal{A}dv^w$ such that $\mathbb{P}_{\mathcal{A}dv^w} = \mathbb{P} \setminus \{p, p'\}$ and $\mathbb{A}_{\mathcal{A}dv^w} = \mathbb{A} \setminus \{a_i\}$, $\mathbb{R}_{\mathcal{A}dv^w} = \mathbb{R}$, and $\mathbb{C}_{\mathcal{A}dv^w} = \mathbb{C}$, i.e., $\mathcal{A}dv^w$ compromises all the producers except for p, p', all the anonymizing routers except for a_i, and all the regular routers and consumers. This CF satisfies the third condition in Theorem 2. We assume another configuration CF' which is identical to CF except that $\mathsf{CF}'(p) = (a_1, \ldots, a_{i-1}, a_i, a'_{i+1}, \ldots, a'_n, c', dat')$ and $\mathsf{CF}'(p') = (a'_1, \ldots, a'_{i-1}, a_i, a_{i+1}, \ldots, a_n, c, dat)$. In this case, $\mathcal{A}dv^2$ can track *dat* and dat' from a_1 to a_{i-1}, from a_{i+1} to a_n, from a'_1 to a'_{i-1}, and from a'_{i+1} to a'_n for CF and CF'. Thus, $\mathcal{A}dv^w$ can distinguish CF from CF' only if $\mathcal{A}dv^w$ can correctly correlate the inputs and outputs at a_i. However, such $\mathcal{A}dv^w$ does not exist because it contradicts Theorem 1. Therefore, $\mathsf{CF}' \equiv_{\mathcal{A}dv^w} \mathsf{CF}$ holds for such CF', and p has producer anonymity. ∎

Proof of Theorem 2 - (4): Without loss of generality, we assume CF such that $\mathsf{CF}(p) = (a_1, \ldots, a_{i-1}, a_i, a_{i+1}, \ldots, a_n, c, dat)$ and $\mathsf{CF}(p') = (a'_1, \ldots, a'_{i-1}, a'_i, a'_{i+1}, \ldots, a'_n, c', dat')$, and $\mathcal{A}dv^w$ such that $\mathbb{P}_{\mathcal{A}dv^w} = \mathbb{P} \setminus \{p, p'\}$ and $\mathbb{A}_{\mathcal{A}dv^w} = \mathbb{A} \setminus \{a_i, a'_i\}$, and $\mathbb{C}_{\mathcal{A}dv^w} = \mathbb{C}$, i.e., $\mathcal{A}dv^w$ compromises all the producers except for p, p', all the anonymizing routers except for a_i, a'_i, and all the consumers. We also assume that $a_i, a'_i \in \mathsf{AS}^{dat_{\mathcal{K}_{n-i}}}_{\mathcal{A}dv^w}$, i.e., a_i and a'_i are included in the same sender anonymity set of their output Data packets with respect to $\mathcal{A}dv^w$. This CF satisfies the fourth condition in Theorem 2. We assume another configuration CF' which is identical to CF except that $\mathsf{CF}'(p) = (a_1, \ldots, a_{i-1}, a_i, a'_{i+1}, \ldots, a'_n, c', dat')$ and $\mathsf{CF}'(p') = (a'_1, \ldots, a'_{i-1}, a'_i, a_{i+1}, \ldots, a_n, c, dat)$. In this case, $\mathcal{A}dv^w$ can track *dat* and

dat' from a_1 to a_{i-1}, from a_{i+1} to a_n, from a'_1 to a'_{i-1}, and from a'_{i+1} to a'_n for CF and CF'. Thus, $\mathcal{A}dv^w$ can distinguish CF and CF' only if $\mathcal{A}dv^w$ can correctly correlate a_i, a'_i with their output Data packets. However, such $\mathcal{A}dv^w$ cannot exist from the definition of the sender anonymity set. Therefore, $\mathsf{CF}' \equiv_{\mathcal{A}dv^w} \mathsf{CF}$ holds for such CF', and p has producer anonymity. ■

REFERENCES

[1] L. Zhang *et al.*, "Named data networking," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 3, pp. 66–73, Jul. 2014.

[2] S. Farrell and H. Tschofenig, "Pervasive monitoring is an attack," Internet Eng. Task Force, RFC 7258, May 2014. [Online]. Available: https://tools.ietf.org/html/rfc7258

[3] J. Griffiths. (2018). *China is Exporting The Great Firewall as Internet Freedom Declines Around the World*. [Online]. Available: https://edition.cnn.com/2018/11/01/asia/internet-freedom-china-censorship-intl/index.html

[4] M. Kim, J. Lim, H. Yu, K. Kim, Y. Kim, and S.-B. Lee, "ViewMap: Sharing private in-vehicle dashcam videos," in *Proc. 14th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, Mar. 2017, pp. 163–176.

[5] K. Kita, Y. Kurihara, Y. Koizumi, and T. Hasegawa, "Location privacy protection with a semi-honest anonymizer in information centric networking," in *Proc. ACM Conf. Inform. Centric Netw.*, 2018, pp. 95–105.

[6] I. Miers, C. Garman, M. Green, and A. D. Rubin, "Zerocoin: Anonymous distributed E-cash from bitcoin," in *Proc. IEEE Symp. Security Privacy*, Berkeley, CA, USA, May 2013, pp. 397–411.

[7] J. Bonneau, A. Narayanan, A. Miller, J. Clark, J. A. Kroll, and E. W. Felten, "Mixcoin: Anonymity for bitcoin with accountable mixes," in *Financial Cryptography and Data Security*. Heidelberg, Germany: Springer, 2014, pp. 486–504.

[8] A. Pfitzmann and M. Hansen. (Feb. 2008). *Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management—A Consolidated Proposal for Terminology, Version V0.31*. [Online]. Available: http://dud.inf.tu-dresden.de/Anon_Terminology.shtml

[9] P. Zhang, Q. Li, and P. P. C. Lee, "Achieving content-oriented anonymity with crisp," *IEEE Trans. Depend. Secure Comput.*, vol. 14, no. 6, pp. 578–590, Nov./Dec. 2017.

[10] S. DiBenedetto, P. Gasti, G. Tsudik, and E. Uzun, "ANDaNA: Anonymous named data networking application," Dec. 2011. [Online]. Available: https://arxiv.org/abs/1112.2205.

[11] M. G. Reed, P. F. Syverson, and D. M. Goldschlag, "Anonymous connections and onion routing," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 4, pp. 482–494, May 1998.

[12] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," in *Proc. 13th Conf. USENIX Security Symp.*, Aug. 2004, p. 21.

[13] C. Díaz, S. Seys, J. Claessens, and B. Preneel, "Towards measuring anonymity," in *Privacy Enhancing Technologies*, R. Dingledine and P. Syverson, Eds. Heidelberg, Germany: Springer, 2003, pp. 54–68.

[14] *Tor Rendezvous Specification—Version 3*, Tor Project, Cambridge, MA, USA, 2017. [Online]. Available: http://gitweb.torproject.org/torspec.git/tree/rend-spec-v3.txt

[15] M. Krol, K. Habak, D. Oran, D. Kutsher, and P. Ioannis, "RICE: Remote method invocation in ICN," in *Proc. 5th ACM Conf. Inf. Centric Netw.*, 2018, pp. 1–11.

[16] L. Overlier and P. Syverson, "Locating hidden servers," in *Proc. IEEE Symp. Security Privacy (SP'06)*, Oakland, CA, USA, May 2006, pp. 100–114.

[17] M. Wright, M. Adler, B. N. Levine, and C. Shields, "Defending anonymous communications against passive logging attacks," in *Proc. Symp. Security Privacy*, Berkeley, CA, USA, May 2003, pp. 28–41.

[18] P. Gasti and G. Tsudik, "Content-centric and named-data networking security: The good, the bad and the rest," in *Proc. IEEE Int. Symp. Local Metropolitan Area Netw. (LANMAN)*, Washington, DC, USA, Jun. 2018, pp. 1–6.

[19] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*. Boca Raton, FL, USA: CRC Press, 2007.

[20] C. Ghali, G. Tsudik, and C. A. Wood, "(The futility of) data privacy in content-centric networking," in *Proc. ACM on Workshop Privacy Electron. Soc.*, 2016, pp. 143–152.

[21] X. Cui, Y. H. Tsang, L. C. K. Hui, S. M. Yiu, and B. Luo, "Defend against Internet censorship in named data networking," in *Proc. 18th Int. Conf. Adv. Commun. Technol. (ICACT)*, Pyeongchang, South Korea, Jan. 2016, pp. 300–305.

[22] J. Feigenbaum, A. Johnson, and P. Syverson, "A model of onion routing with provable anonymity," in *Financial Cryptography and Data Security*. Heidelberg, Germany: Springer, 2007, pp. 57–71.

[23] A. Ghodsi, T. Koponen, J. Rajahalme, P. Sarolahti, and S. Shenker, "Naming in content-oriented architectures," in *Proc. ACM SIGCOMM Workshop Inf. Centric Netw.*, 2011, pp. 1–6.

[24] Z. Zhang *et al.*, "An overview of security support in named data networking," *IEEE Commun. Mag.*, vol. 56, no. 11, pp. 62–68, Nov. 2018.

[25] P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman, "Reputation systems," *Commun. ACM*, vol. 43, no. 12, pp. 45–48, Dec. 2000.

[26] *Web of Trust*, WOT Serv., Helsinki, Finland, 2007. [Online]. Available: http://www.mywot.com

[27] Y. Yu, A. Afanasyev, Z. Zhu, and L. Zhang, "An endorsement-based key management system for decentralized NDN chat application," Univ. California, Los Angeles, CA, USA, Rep. NDN-0023, 2014. [Online]. Available: https://named-data.net/publications/techreports/ndn-tr-23-chronochat-security/

[28] M. Mosko, E. Uzun, and C. A. Wood, "Mobile sessions in content-centric networks," in *Proc. IFIP Netw. Conf. Workshops*, Stockholm, Sweden, Jun. 2017, pp. 1–9.

[29] A. Biryukov, I. Pustogarov, and R.-P. Weinmann, "Trawling for Tor hidden services: Detection, measurement, deanonymization," in *Proc. IEEE Symp. Security Privacy*, Berkeley, CA, USA, May 2013, pp. 80–94.

[30] *Tor Protocol Specification*, Tor Project, Cambridge, MA, USA, 2003. [Online]. Available: http://github.com/torproject/torspec/blob/master/torspec.txt

[31] *Tor Metrics Portal*, Tor Project, Cambridge, MA, USA, 2019. [Online]. Available: https://metrics.torproject.org

[32] S. J. Murdoch and G. Danezis, "Low-cost traffic analysis of Tor," in *Proc. IEEE Symp. Security Privacy (SP'05)*, Oakland, CA, USA, May 2005, pp. 183–195.

[33] S. Chakravarty, A. Stavrou, and A. D. Keromytis, "Identifying proxy nodes in a Tor anonymization circuit," in *Proc. IEEE Int. Conf. Signal Image Technol. Internet Based Syst.*, Bali, Indonesia, Nov. 2008, pp. 633–639.

[34] P. Mittal, A. Khurshid, J. Juen, M. Caesar, and N. Borisov, "Stealthy traffic analysis of low-latency anonymous communication using throughput fingerprinting," in *Proc. 18th ACM Conf. Comput. Commun. Security*, 2011, pp. 215–226.

[35] V. Shmatikov and M.-H. Wang, "Timing analysis in low-latency mix networks: Attacks and defenses," in *Computer Security—ESORICS 2006*. Heidelberg, Germany: Springer, 2006, pp. 18–33.

[36] C. Gulcu and G. Tsudik, "Mixing E-mail with Babel," in *Proc. Internet Soc. Symp. Netw. Distrib. Syst. Security*, San Diego, CA, USA, Feb. 1996, pp. 2–16.

[37] T. Elahi, K. Bauer, M. AlSabah, R. Dingledine, and I. Goldberg, "Changing of the guards: A framework for understanding and improving entry guard selection in Tor," in *Proc. ACM Workshop Privacy Electron. Soc.*, 2012, pp. 43–54.

[38] W. Greene and B. Lancaster, *Carrier-Grade: Five Nines, The Myth and The Reality*, vol. 3, Pipeline, London, U.K., 2006.

[39] G. Tsudik, E. Uzun, and C. A. Wood, "AC3N: Anonymous communication in content-centric networking," in *Proc. 13th IEEE Annu. Consum. Commun. Netw. Conf.*, Las Vegas, NV, USA, Jan. 2016, pp. 988–991.

[40] *Coverage Areas of Telephone Exchange Buildings*, NTT East Corp., Chiyoda City, Japan, 2016. [Online]. Available: https://www.ntt-east.co.jp/info-st/info_dsl/area.html

[41] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Commun. ACM*, vol. 24, no. 2, pp. 84–90, Feb. 1981.

[42] S. Kaushik, R. Tourani, G. Torres, S. Misra, and A. Afanasyev, "NDN-ABS: Attribute-based signature scheme for named data networking," in *Proc. ACM Conf. Inf. Centric Netw.*, 2019, pp. 123–133.

[43] R. Tourani, S. Misra, J. Kliewer, S. Ortegel, and T. Mick, "Catch me if you can: A practical framework to evade censorship in information-centric networks," in *Proc. 2nd ACM Conf. Inf. Centric Netw*, 2015, pp. 167–176.

[44] J. Kurihara, K. Yokota, and A. Tagami, "A consumer-driven access control approach to censorship circumvention in content-centric networking," in *Proc. 3rd ACM Confe. Inf. Centric Netw.*, 2016, pp. 186–194.

[45] S. Arianfar, T. Koponen, B. Raghavan, and S. Shenker, "On preserving privacy in content-oriented networks," in *Proc. ACM SIGCOMM Workshop Inf. Centric Netw*, 2011, pp. 19–24.
[46] M. K. Reiter and A. D. Rubin, "Crowds: Anonymity for Web transactions," *ACM Trans. Inf. Syst. Security*, vol. 1, no. 1, pp. 66–92, Nov. 1998.

Kentaro Kita (Student Member, IEEE) received the master's degree in information science from Osaka University, Japan, in 2020, where he is currently pursuing the Ph.D. degree with the Graduate School of Information Science and Technology. His research interests include privacy, anonymity, security, and information centric networking. He is a member of the IEICE.

Yuki Koizumi (Member, IEEE) received the master's and Ph.D. degrees in information science from Osaka University, Japan, in 2006 and 2009, respectively, where he is an Associate Professor with the Graduate School of Information Science and Technology. His research interests include information-centric networking. He is a member of ACM and IEICE.

Toru Hasegawa (Member, IEEE) received the B.E., M.E., and Dr. Informatics degrees in information engineering from Kyoto University, Japan, in 1982, 1984 and 2000, respectively. After receiving the master degree, he worked as a Research Engineer with KDDI Research and Development Labs (formerly, KDD R&D Labs.) for 29 years and moved to Osaka University, where he is a Professor with the Graduate school of Information and Science. He has published over 100 papers in peer-reviewed journals and international conference proceedings, including MobiCom, ICNP, IEEE/ACM TRANSACTIONS ON NETWORKING, *Computer Networks*. His current interests are future Internet, information centric networking, and mobile computing. He received the Meritorious Award on Radio of ARIB in 2003, the Best Tutorial Paper Award in 2014 from IEICE, and the Best Paper Award in 2015 from IEICE. He has served on the program or organization committees of several networking conferences, such as ICNP, P2P, ICN, CloudNet, ICC, and Globecom, and as a TPC Co-Chair of Testcom/Fates in 2008, ICNP in 2010, P2P in 2011, and Global Internet Symposium in 2014. He is a fellow of IPSJ and IEICE.

Onur Ascigil (Member, IEEE) received the Ph.D. degree from the Computer Science Department, University of Kentucky, USA. He is currently a Research Associate with the Department of Electronic and Electrical Engineering, University College London, U.K. His research interests include information-centric networking, routing, edge computing, and peer-to-peer networks.

Ioannis Psaras (Member, IEEE) is a Research Scientist with Protocol Labs and a University Lecturer (Assistant Professor) with University College London. He is interested in resource management techniques for current and future networking architectures with particular focus on routing, caching and congestion control. Over the last few years he has focused on function-centric networks to realise distributed and decentralised edge computing, also referred to as "computing in the network." He holds a prestigious EPSRC Early Career Fellowship in the area of "content-oriented and service-centric edge-computing architectures." He has been heavily involved in the effort to shift the Internet towards an Information-Centric Networking environment and lately has been focusing on the application of blockchain technologies to decentralised, permissionless storage networks.