Investigating the Design Space for Name Confidentiality in Named Data Networking

Zhiyi Zhang
UCLA
zhiyi@cs.ucla.edu

Su Yong Won
UCLA
swa770@ucla.edu

Lixia Zhang
UCLA
lixia@cs.ucla.edu

MILCOM 2021 - 2021 IEEE Military Communications Conference (MILCOM) | 978-1-6654-3956-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/MILCOM52596.2021.9652892

***Abstract*—As a fundamental departure from the IP design which encodes source and destination addresses in each packet, Named Data Networking (NDN) directly uses application-defined data names for network layer communications. While bringing important data-centric benefits, the semantic richness of NDN names has also raised confidentiality and privacy concerns. In this paper, we first define the problem of name confidentiality, and then investigate the solution space through a comprehensive examination of all the proposed solutions up to date. Our work shows that the proposed solutions are simply different means to hide the actual data names via a layer of translation; they differ in where and how the translation takes place, which lead to different trade-offs in feasibility, efficiency, security, scalability, and different degrees of adherence to NDN's data-centric communications. Our investigation suggests the feasibility of a systematic design that can enable NDN to provide stronger name confidentiality and user privacy as compared to today's TCP/IP Internet.**

***Index Terms*—named data networking, privacy, confidentiality**

I. Introduction

Delivering packets to a specific application via today's Internet requires identifiers at three different protocol layers: (i) an IP address, to which a host is attached, that the packet should be delivered to, (ii) a application process identifier, i.e. the port number, in the host, and (iii) an application layer identifier for the specific resources, e.g. data or services. In addition, each packet also carries the source address and port number as part of the transport connection identifier. In today's Internet protocol stack, these identifiers are defined and used by different protocol layers, with the assumption that a lower layer should not see the identifiers from an upper layer. The wide adoption of TLS further seals the visibility of the application layer identifiers from all the layers below.

Named Data Networking (NDN) [1] uses application layer names for network layer data fetching, integrating all the above identifier information into the data names. In NDN, to fetch a piece of data, an application issues an Interest packet carrying the desired data name to the network. Such a data name is used by network forwarding, by transport demultiplexing inside the host, and by the application to identify the specific resource or data it refers to [2]. Being a data centric design, NDN removes packet source identifiers from all packets.

This work is partially supported by the National Science Foundation under award 1719403.

The work from the author, Su Yong Won, was done prior to joining Amazon.

The use of data names is a key enabler of NDN's data-centric properties. However, exposing semantically meaningful names at the network layer also raises name confidentiality and consequent user privacy concerns. Because a name may reveal information ranging from the content itself to characteristics of packet creators [3], which leads to concerns of user profiling and network censorship. Although a number of solutions have been proposed to hide NDN data names, there is a lack of shared understanding of both the problem and the solution space.

In this paper, we make the following contributions: (i) we compare the information disclosure in TCP/IP architecture versus in NDN, (ii) provide a precise definition of the NDN name confidentiality and the associated user privacy problem, (iii) conduct a systematic examination of the existing literature, classify them based on their design choices, and analyze their effectiveness and limitations, and (iv) discuss the tradeoffs among different approaches and design space for future solutions.

In the rest of the paper, in Section II, we examine the information carried in NDN names and the information disclosure from network identifiers in NDN and TCP/IP, and then define the name confidentiality problem and its derived user privacy concerns in NDN. In Section III, we review, classify, and summarize the existing name obfuscation approaches up to date We discuss the common design patterns and several unique features in NDN that can help preserve confidentiality and user privacy in Section IV and conclude our work in Section V.

II. NDN Name Confidentiality and User Privacy

Assuming that readers are familiar with the basic concepts of NDN [1] and NDN security [4], in this section we first compare the information disclosure by different protocol identifiers, and then define the problems associated with such disclosures.

A. Information Disclosure in TCP/IP and NDN

Compared with TCP/IP, NDN makes two fundamental changes regarding information disclosure by identifiers used at the network layer. First, it exposes semantically meaningful data names at the network layer. Second, it removes the source information of data requests (*i.e.*, NDN packets do not carry information equivalent to source IP addresses). This second change enhances user privacy by making it difficult to associate an Interest packet with its sender from far, although observers that are close to the sender, *e.g.*, sender's ISP, may still be able to make the association. This makes a big difference from TCP/IP where the sender of a packet is directly identified by the source address, unless additional security mechanisms, such as VPN or Onion Routing, are deployed.

It is the first change that raises the name confidentiality and user privacy concerns. More specifically, NDN folds three types of information into a single data name:

- Information that allows the network to identify network node(s) associated with the data name;
- Information that allows demultiplexing among applications within a host; and
- Information that identifies a specific resource served by the application.

We denote name components that carry these pieces of information by N1, N2, and N3, respectively. Note that N1 and N2 can be the same name component(s). Among the three pieces, N1 disclose information similar to what can be inferred from DNS names, and N2 similar to port number in TCP/IP, while N3 carries application information, such as what an HTTP URL encodes. While N1 and N2 are usually encoded into NDN names directly, it is also possible to decouple them from names (§IV-B).

B. The Problem Definition

In NDN, the problem of name confidentiality and the derived user privacy can be described as follows: *The information revealed by NDN names, including N1-3, can potentially reveal information of the carried content and associate the packet to a specific user.* Note that we do not usually consider the identity information leaks of a producer because to publish Data packets, the producer needs to make its name known and reachable.

The potential consequences of undesired information leakage from name can include: (i) Network censorship based on name and the content information by analyzing the content-related information disclosed from names and network censorship based on names and information revealed by names. (ii) Compromise of the user privacy by collecting and observing NDN names sent by a user.

Therefore the goal primarily content confidentiality and user privacy as listed in G1 and G2.

G1: Confidentiality & Resistance to Censorship. Single or multiple NDN names should reveal no information related to the content carried by the packet. This also prevents an adversary from conducting censorship by filtering Interest packets with any characteristics of names. Compared with TCP/IP, the new challenge brought by using NDN is to preserve the confidentiality of N3, as the equivalent information for N1 and N2 is also revealed in TCP/IP (even when secured by TLS). For simplicity, we refer to N3 confidentiality as **G0**. Note that when G1 is realized, G0 is also achieved, but solely realizing G0 does not lead to G1 (as a censorship service can still filter packets based on N1 and N2).

G2: User Privacy. Each observed name contains no information of the Interest sender. With this property, a remote attacker cannot associate multiple Interest packets generated by the same user. Even for an attacker who can associate packets from the same user, for example, by eavesdropping near the consumer (*e.g.* in the local ISP), this property prevents the attacker from guess who the user is.

Besides the basic goals, Ghali *et al.* [5], [6] also argue that strong privacy requires Interest unlinkability about data, meaning that an adversary cannot examine whether two Interest packets refer to the same content, to resist traffic pattern analysis. We list it as **G3: Same-name Unlinkability** for a better coverage. However, in §IV, we will discuss that achieving this goal can degrade NDN's data-centric properties.

C. Design Choices

In order to hide the semantics of the original name or to cut the linkability between names and users, what is needed is another layer of translation. When designing a solution to NDN name confidentiality and privacy, a list of questions regarding this layer of translation need to be answered.

D1: Where (Proxy or not). One option is to directly let communication participants obscure names. The other choice is to rely on single or multiple trusted proxies.

D2: How (Encapsulation or not). One option is to directly obscure the name without changing the packet structure, and the obfuscated name is used for forwarding. The other option is to encrypt the original packet and encapsulate it into another packet. The outer packet's name will be used for forwarding.

D3: What (Full or partial name). Information is not equally important in a name. One can decide to hide different sections of a name. For example, obfuscating the name suffix (N3) is sufficient if the goal is to hide the content identifier.

D4: Name Granularity (Per-component or not). The obfuscation can take place at the name component level so that the original hierarchy in an NDN name can be preserved. Another option is to collapse multiple components into a flat component. The granularity choice for N1 is the most important as it directly affects the network scalability (*e.g.*, a flat N1 can break the hierarchical forwarding strategy).

D5: Entity Granularity (Per-consumer or not). A name obfuscation approach can either choose to (i) obfuscate the same name in the same way for all consumers or (ii) do it differently for each consumer. When the latter is adopted, an Interest packet cannot merge another Interest packet or hit a cached target Data packet as their names are coded differently.

III. Review of Existing Proposals

The goal of reviewing existing name privacy solutions is to understand how existing name obfuscation mechanisms work, what they pay and earn, and whether they retain the fundamental purpose of NDN. Specifically, we collected our corpus from two main sources, (i) papers from Google Scholar search with keywords like "privacy", "confidentiality", "name", "NDN" (or "ICN", "CCN"), etc., and (ii) the referenced related works of these papers. To the best of our effort, we found and surveyed more than 15 works. We believe our corpus includes most of the major name confidentiality and privacy solutions in NDN, providing a good starting point to understand its current landscape. In addition, we categorize the existing works based on their main approaches to facilitate the understanding of commonalities and differences between the existing works. Figure 1 depicts the overview of our classification.

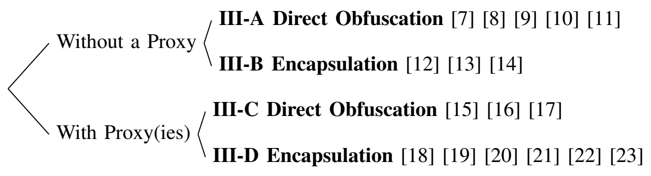
Without a Proxy
III-A Direct Obfuscation [7] [8] [9] [10] [11]
III-B Encapsulation [12] [13] [14]
With Proxy(ies)
III-C Direct Obfuscation [15] [16] [17]
III-D Encapsulation [18] [19] [20] [21] [22] [23]

Fig. 1. Classification of Existing Name Privacy Solutions

A. Direct Obfuscation without Proxies

This type of approach directly lets applications obscure names.

A.1 Partial Name Encryption. NDN confidentiality solutions like Name based Access Control (NAC) [7] not only can be used for content encryption but also for name obfuscation. We use NAC as an example. NAC takes a hybrid encryption scheme where the data is encrypted with symmetric keys generated by the producer and symmetric keys are encrypted with asymmetric keys managed by an access controller. When producing a Data packet, besides encrypting the data content, a producer can also encrypt sensitive name components. Nevertheless, such encryption can only apply to N3 because N1 and N2 are used for networking. For consumers to fetch the encrypted packet, consumers need to learn the encrypted name to construct Interest packets and obtain the decryption keys from the access manager to decrypt the name and the packet.

Therefore, confidentiality of N3 (G0) is accomplished as N3 is encrypted and the user privacy (G2) is also preserved because Interest packets do not carry consumer information. However, partial name encryption approaches do not resist prefix filtering based censorship (G1); we discuss how this can be mitigated with forwarding hints in §IV-B. Since Interest packets are not obfuscated per consumer, same-data unlinkability (G3) is not supported, but on the other hand, the data-centric properties like Interest aggregation and cache are not affected.

A.2 Full Name Homomorphic Encryption. Homomorphic cryptography is used in Ko *et al.*'s work [8], He *et al.*'s work [9], and PATS_NDN [10] to obfuscate full NDN names. A homomorphic algorithm encrypts the name in a way that a router can understand whether two different ciphertexts originated from the same plaintext (Figure 2). More specifically, even if consumers A and B obfuscate the same Interest packet to the different names, a router still knows the two obfuscated

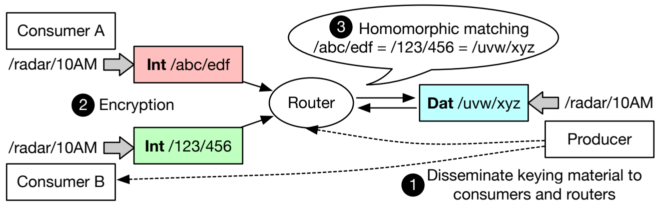

Fig. 2. Homomorphic Obfuscation of Names

names target the same data. For example, Ko *et al.*'s work [8] uses Public-key Encryption with Keyword Search (PEKS) to encrypt each name component where the keyword used for ciphertext search is the name component. To be more specific, a producer first disseminates its public key and a one way function called a trapdoor for each name component to both consumers and routers. Then, consumers can encrypt their Interest packets with PEKS using the producer's public key and a router can compare different encrypted Interest names by matching them on pre-disseminated trapdoor functions. He *et al.*'s work [9] and PATS_NDN [10] took a similar approach but are based on different cryptographic schemes.

Homomorphic approaches provide stronger protection than partial name encryption by fully encrypting the Interest name and thus is resistant to both next hop eavesdropping and censorship (G1, G2). There is a unique obfuscated namespace for each consumer so same-name unlinkability (G3) is also achieved. While achieving G3, homomorphic approaches do not break the data centricity by the homomorphic name comparison at routers.

A.3 Full Name Bloom Filter Obfuscation. Chaabane *et al.* suggest the use of Bloom Filters (BFs) for name obfuscation [11]. For example, to request data named "`/ucla/cs/file1`", the consumer computes separate BFs for "`/ucla`", "`/ucla/cs`", and "`/ucla/cs/file1`". Then the consumer sends out these BFs as the Interest. Since BF allows one to check the member's existence in the set, a router can perform the longest prefix matching by checking the existence of the prefixes in the forwarding table. This solution also obscures the full name while preserving the ability for routers to match names. However, we do not investigate this direction much because there is a lack of systematic design description and implementation, and several issues remain open, *e.g.*, attackers can also perform the same bloom filter queries, false positive results, etc.

Summary and Tradeoffs. Homomorphic solutions seem like an ideal solution to name confidentiality and privacy by providing both full name obfuscation while preserving NDN's data-centric features. However, it comes with costs from various sources, among which, the operational costs at routers can make homomorphic approaches impractical. Specifically, homomorphic matching at routers is much more expensive than regular name matching and routers may need to maintain per-namespace or per-component keying material. In comparison, while only obscuring N3, NAC and other partial name encryption approaches do not require modification at routers and have a better efficiency. It is also noteworthy that both partial name encryption approaches and homomorphic approaches require a trust setup among communication participants, *e.g.*, for key dissemination.

B. Encapsulation without Proxy

B.1 Direct Packet Encapsulation. While not explicitly designed for name obfuscation, several NDN sync protocols [24] can provide name confidentiality and privacy by encapsulating application-generated Data packets into wrapper packets. We use the latest NDN Sync protocol, State Vector Sync (SVS) [12], as an example. In SVS, the wrapper packets are named with a per-producer sequence number, revealing no content information. To fetch the original packets, a consumer will use outer packet names as Interest packets, and multiple original Data packets can be encrypted and embedded into a single NDN sync Data packet. Serban *et al.* [13] also propose a full packet encapsulation mechanism for name privacy. Different from SVS, this approach encrypts and encapsulates both Interest and Data packets into outer packets with a one-to-one mapping. An outer packet carries information such as a key-locator for the packet receiver to decrypt the inner packet.

In both SVS and Serban *et al.*'s work, the original packet is encrypted and encapsulated and Interest packets carry no consumer information, so N3 confidentiality (G0) and user privacy (G2) are preserved. To be forwarded by NDN routers, the outer packets are still named under the producer's prefix (either encrypted or in plaintext), and thus these approaches are subject to prefix based censorship filtering (G1); we discuss how this can be mitigated with forwarding hints in §IV-B. Interests to fetch the same outer packets have the same name, so same-data unlinkability (G3) is not supported, but thus NDN's data-centric properties still hold.

B.2 Content Mixing Based Encapsulation. Arianifar *et al.*'s proposed a system [14] to obfuscate both name and content by mixing the target content's constituent data blocks with the blocks of normal content (called a cover file). To be more specific, a producer (i) splits target files into multiple chunks, (ii) mixes (exclude-or operation) target file chunks and cover file's chunks, and (iii) publish the mixed chunks into Data packets with an obfuscated name. The obfuscated name is generated based on the hash value of the mixed content. Based on metadata about the algorithm, block size, cover file, etc., a legitimate consumer can fetch a purposely-chosen set of the mixed blocks and reconstruct the target file. In contrast, attackers need to pay a higher computational cost to figure out the names as they are not aware of which combinations of blocks are for reconstruction.

Arianifar *et al.*'s work [14] does not guarantee the confidentiality and privacy goals because it leverages computational asymmetry to make it harder for attackers to figure out the original name.

Summary and Tradeoffs. After the encapsulation, the wrapper packets are used for networking forwarding instead of the original packets. Since the wrapper packets are fetched in a normal NDN way, the data-centric properties are not affected. Both NDN Sync based encapsulation and Serban *et al.*'s full packet encapsulation require a trust setup. For example, in SVS, the communication participants need to form a sync group with the group management so that unauthorized parties cannot join. Besides, extra key management (*e.g.* for sync-group-based message encryption) is also needed so that each participant can obtain and renew their keys, and the system can also revoke compromised participants.

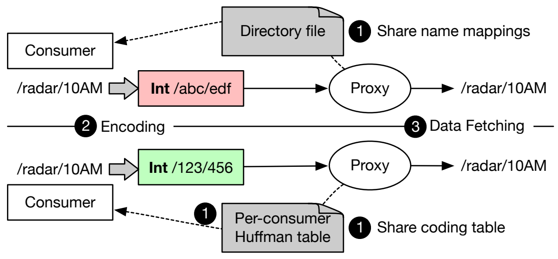

Fig. 3. Proxy based name obfuscation with BEAcM-DP [15] (top) and Tourani *et al.*'s work [16] (bottom)

C. Name Obfuscation via Proxy

This type of approach utilizes one or more proxy servers for name obfuscation by directly updating the packet names.

C.1 Single Proxy. Zhu *et al.* proposed BEAcM-DP [15] to let consumers send Interest packets with obfuscated names and utilize a proxy, called an anonymizer, to translate the obfuscated names back to real names. To be more specific, in BEAcM-DP, the anonymizer provides authorized consumers a directory file for the mapping between original names and obfuscated names. Therefore, the consumers can send obfuscated-named Interest packets to the anonymizer. The anonymizer then retrieves the content with the original name and forwards the replied Data packet back to the consumer using broadcast encryption under the obfuscated name. Tourani *et al.*'s work [16] shares a similar idea, but instead of using a directory file, it let consumers acquire a Huffman table from the proxy and send Huffman-coded Interest packets to the proxy (Figure 3). A different design choice is that in Tourani, obfuscated names are user-specific because each consumer receives a unique Huffman table. When handling a replied Data packet, in both BEAcM-DP and Tourani *et al.*'s approach, the target file will be encrypted, named with the corresponding fake or Huffman-coded Interest name, and sent back to the consumer.

Since the original name is fully hidden, these approaches provide stronger confidentiality (G1) between the consumer and the proxy. The privacy (G2) is also preserved by using the proxy to break the consumer-Interest linkability. Same-data name unlinkability (G3) cannot be achieved internal the proxy because both Huffman encoding and directly based translation are deterministic. For the same reason, NDN's data-centric properties are not degraded.

Work	Layers of translation	Where to add proxy(ies)	When traffic is protected
[15], [16]	Single layer	Near consumer	Consumer-proxy
[17]	Two layers	Near consumer	Consumer-proxy (under per-consumer encryption), proxy-producer (under master key encryption)
[18]	Single layer	Near consumer/producer	Proxy-proxy
[19]	Single layer	Near consumer/producer	Consumer-proxy, proxy-producer
[20], [21], [22], [23]	Multiple layers	Between consumer and producer	Consumer-proxy (the last onion router)

TABLE I
A COMPARISON OF PROXY BASED APPROACHES

C.2 Re-encryption with Proxies. PrivICN [17] utilizes edge routers near consumers as transparent proxies and applies a re-encryption scheme. To start with, a consumer registers with a centralized authority. The authority will generate a pair of a client key C and a proxy key P, and distributes C to the consumer and P to the consumer's edge router. C and P are generated in a way that serial encryption by C and P is the same as a single asymmetric key encryption by the master key M from the authority. In PrivICN, a consumer encrypts an interest packet with C, and the edge router transparently re-encrypts it with P. As a result, all the encrypted packets after the edge routers follow the form of master key encryption. When a master-key-encrypted Data packet comes back, the edge router first decrypts the incoming packet using P to transform it back to the client-specific encryption so that the client can decrypt it with C.

Besides achieving G1-2, since the encryption key C is generated per consumer, same-data name unlinkability (G3) is achieved between consumers and the proxy. Correspondingly, Interest aggregation and cache cannot be used between consumers and the proxy.

Summary and Tradeoffs. All approaches under this category provide strong protection of confidentiality and privacy. Nevertheless, since the proxy will change the packet name, when processing a Data packet or a signed Interest packet, the change of name will invalid the original signatures made by the producer. In addition, all approaches under the category require the deployment of a trusted proxy and when there is a single proxy, it can be a single point of failure. Also, to forward encrypted names in PrivICN, encrypted name prefixes will be used in the routing system correspondingly.

The solutions under this category are also listed in Table I for a comparison of all the proxy based approaches.

D. Encapsulation with Proxy

Several works employ proxies to encrypt and encapsulate original packets for name confidentiality and privacy in NDN.

D.1 VPN based Encapsulation. When a single proxy is used to translate packets for the local network, it is similar to how Virtual Private Network (VPN) [25] works over TCP/IP. For example, NDN-in-NDN [18] provides a solution using the network gateway as a transparent proxy. As the initialization, a centralized authority distributes a shared symmetric key over the network gateways. Assuming the network behind the gateway is secured, each gateway encrypts Interest packets or encapsulates Data packets when they go out to the insecure network. Vice versa, it decrypts or decapsulates them when they enter the secure network. Another work is proposed by Liu *et al.* to provide an identity privacy protection system [19] for vehicular NDN. Nevertheless, instead of assuming a secured local network, a consumer will use the proxy's public key to encrypt and encapsulate the Interest name under the proxy's name prefix. Opposite to NDN-in-NDN, in Liu *et al.*'s design, the packets sent out by the proxy are in plain text.

Both NDN-in-NDN and Liu *et al.*'s work preserve full name confidentiality (G1) and user privacy (G2). However, NDN-in-NDN does not provide full resistance to censorship (G1) because the producer's gateway prefix will be used for forwarding and it can subject to filtering. In both solutions, the NDN data-centric properties are not affected when packets are out of the proxy.

D.2 Onion Routing based Encapsulation. Other approaches employ multiple proxies similar to Onion Routing [26]. Onion routing ensures anonymous communication by forwarding concentrically encrypted packets through a series of ephemeral proxies. As the packet is forwarded, each proxy decrypts a layer of encryption like peeling an onion. ANDaNA [20] is the first work bringing onion routing into NDN. In ANDaNA, a consumer chooses at least two onion routers (*i.e.*, ephemeral proxies) and fetches their public keys from a centralized authority. Then, the consumer (i) generates two symmetric keys, (ii) concentrically encrypts the Interest packet and the two symmetric keys by onion routers' public keys, and (iii) encapsulates the ciphertext into an Interest packet to the first onion router. The first onion router decrypts the packet using its private key, stores the first symmetric key, and forwards the inner packet to the second onion router who will follow the same procedure and send the plain text Interest packet out. When forwarding back the Data packet, each onion router later uses its symmetric key to encrypt the replied data. Figure 4 demonstrates the process in detail. AC3N [21] proposed by Tsudik *et al.* is an evolved version of ANDaNA [20] with improved performance. Later works like Seo *et al.*'s work [22] and Kim *et al.*'s work [23] share a similar idea but are based on variants of the onion routing.

Since onion routing encrypts the entire packet and ensures consumer anonymity, it preserves the content confidentiality and user's privacy (G1-2). Similar to the single proxy encapsulation approaches, when packets leave the last hop of the onion routing network, NDN properties are not affected. In contrast, within the onion network, since each consumer's packet is encrypted differently, same-data unlinkability (G3) is

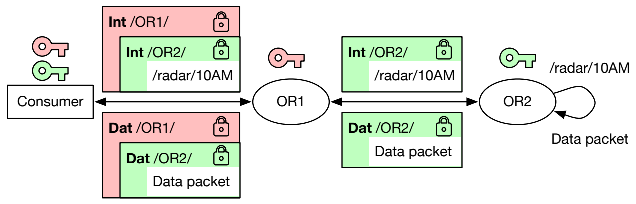

Fig. 4. The use of ephemeral proxies in ANDaNA [20]

achieved but Interest aggregation and cache are not available.

Summary and Tradeoffs. While providing good amount of confidentiality and privacy, the solutions in this category are at the cost to set up and maintain the proxy network. Besides, the cryptographic operations can cause considerable overhead and the packet forwarding path in the onion routing is not optimized for low latency.

The solutions under this category are also listed in Table I.

IV. Discussion

A. Common Design Patterns

We summarize three common design patterns and discuss the tradeoffs made by each. The First common pattern is to employ proxies for packet encapsulation, which provides strong name protection because the entire original name is hidden. The trade off is that trusted proxies must be deployed with associated costs and operational complexity Furthermore, we point out that the existing proxy solutions do not pay adequate attention to security when proposing modification of Data packet names, which break the data integrity and authenticity.

The second, homomorphic approaches appear to provide strong protection of confidentiality and user privacy by encrypting the full name directly at the consumer side, while not breaking data-centric properties by allowing routers to determine whether different obfuscated names target the same data. Unfortunately, this seemingly strong solution carries a prohibitively high cost: each data producer must disseminate its key to all routers in the network, NDN's forwarding pipeline must be changed, and all the routers must support expensive homomorphic operations.

The third pattern is to apply name obfuscation at end-to-end layer, between producers and consumers directly, *e.g.*, NAC based name encryption, NDN Sync based data encapsulation, and encapsulation and encryption of entire NDN packets. Compared with the previous two patterns, this one has some unique advantages. (i) It requires no third party services like proxies nor changes at routers, making the solution highly deployable. (ii) With the encapsulation and encryption of entire NDN packets, the original communication information can be entirely hidden. Even with partial name information hiding, such as provided by NAC and Sync, the prefix information used for packet forwarding is not hidden, the most sensitive information of N3 is obscured. In the latter case, this pattern does not fit use cases where anti-censorship as indicated by G2 is required because the producer prefix is still visible, however as discussed in Section IV-B, when forwarding hints and third party storage services are used, the producer prefix can also be removed.

B. The Decouple of Data Name and Data Containers in NDN

The first two sections of names, N1 and N2, are to identify an application process of a host in the network. However, in NDN, one data name can served by multiple hosts. For example, when the data is cached by storage services (*e.g.*, Content Delivery Networking nodes), these services can also announce the same name prefixes to the network. This is because an NDN name identifies a piece of data instead of specific data containers.

Furthermore, a producer can remove N1 and N2 information from an actual name by letting the consumers use forwarding hints. That is, a producer can name her data purely based on N3 information and encode the topological location(s) of the data into forwarding hints; on the other side, consumers can obtain forwarding hints and send them with the Interest packet when requesting the data. Routers will forward the Interest packet by the forwarding hint to reach the data.

While originally designed for NDN routing scalability support, this mechanism can be utilized to enhance privacy for both consumers and producers. For example, as illustrated in a recent work done by Zhang *et al.* [27], a producer can send the data to a third party data repository for publication and inform the consumer the corresponding forwarding hints for better privacy. The similar strategy can also be used together with mechanisms from (**A.1**) and (**B.1**) to hide producers' prefixes and provide resistances to filtering based censorship. In addition, since forwarding hints are not static like a data name, the forwarding hints can be dynamically changed as the data is replicated by other network nodes, providing even stronger anti-censorship ability.

C. Cache for Strong User Privacy

The ubiquitous adoption of cache in NDN can enhance user privacy against traffic analysis, censorship, and service providers. This is because when data is cached near consumers, the Interest packets can be satisfied before they reach an observation point for traffic analysis, censorship filtering, or service providers.

Compared with TCP/IP where the dedicated cache service needs to be set up at the application layer, NDN makes it intrinsic in the network layer and available to all applications.

D. Future Directions

Compared with TCP/IP connections secured by TLS, an NDN name may leave N3 in clear, without confidentiality protection; the information associated with N1 and N2 is similar to that of a DNS name (and corresponding IP addresses) and port numbers. Therefore, finding solutions to N3 obfuscation is highly desirable for future deployment of NDN.

Duo to the fact that (i) homomorphic solutions are still far away from being practical for deployment in operational networks, and (ii) onion routing and other proxy based solutions do not fit the Internet scale scenarios, future research should prioritize end-to-end name obfuscation solutions. The potential promising directions are proxy-less name encryption (*i.e.* **A.1**) and encapsulation (*i.e.* **B.1**) mechanisms.

As the advance of hardware and cryptography, if there is practical construction of homomorphic encryption that meets the security, efficiency, and scalability requirements. It can be considered as a built-in component of NDN.

V. Conclusion

This paper explores the design space of the NDN name confidentiality and the derived user privacy problem. Throughout the discussion, we have the following arguments.

First, comparing NDN with TCP/IP, from the network layer perspective, N1 and N2 do not expose more information than DNS names. Furthermore, removing source addresses and supporting forwarding hint and cache, NDN enhances consumer privacy and provides more flexibility for applications to improve the privacy of network layer identifiers. At the application layer, when N3 is obscured, TCP/IP with TLS is no better than NDN with regards to data confidentiality and user privacy.

Second, the root cause of the confidentiality and privacy issues is the information revealed by names. Since the information in name is required by the involved applications, the information cannot be removed. Therefore, the solution strategy is to introduce another layer of translation so as to hide the information from unauthorized parties. As such, the name confidentiality and the related name privacy issue are solvable and we have seen different categories of approaches have been proposed for various application requirements.

Last but not the least, among the existing proposals, we observe that obfuscating names at the end host can avoid the deployment of middleboxes that can bring new challenges like trust setup and unclear incentive model.

References

[1] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. Claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named data networking," *ACM SIGCOMM CCR*, vol. 44, no. 3, pp. 66–73, 2014.

[2] H. Zhang, Y. Li, Z. Zhang, A. Afanasyev, and L. Zhang, "Ndn host model," *ACM SIGCOMM CCR*, vol. 48, no. 3, pp. 35–41, 2018.

[3] K. V. Katsaros, L. Saino, I. Psaras, and G. Pavlou, "On information exposure through named content," in *10th International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness*, 2014, pp. 152–157.

[4] Z. Zhang, Y. Yu, H. Zhang, E. Newberry, S. Mastorakis, Y. Li, A. Afanasyev, and L. Zhang, "An overview of security support in named data networking," *IEEE Communications Magazine*, vol. 56, no. 11, pp. 62–68, 2018.

[5] C. Ghali, G. Tsudik, and C. A. Wood, "When encryption is not enough: Privacy attacks in content-centric networking," in *Proceedings of the 4th ACM Conference on Information-Centric Networking (ICN)*. ACM, 2017, p. 1–10.

[6] ——, "(the futility of) data privacy in content-centric networking," in *Proceedings of the 2016 ACM on Workshop on Privacy in the Electronic Society*, ser. WPES '16. ACM, 2016, p. 143–152.

[7] Z. Zhang, Y. Yu, S. K. Ramani, A. Afanasyev, and L. Zhang, "Nac: Automating access control via named data," in *IEEE Military Communications Conference (MILCOM)*. IEEE, 2018, pp. 626–633.

[8] K. T. Ko, H. H. Hlaing, and M. Mambo, "A peks-based ndn strategy for name privacy," *Future Internet*, vol. 12, no. 8, 2020. [Online]. Available: https://www.mdpi.com/1999-5903/12/8/130

[9] H. He and B. Chen, "An elliptic curve based name privacy protection mechanism for sensory data centric named data networking," in *2019 15th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN)*, 2019, pp. 56–62.

[10] "Privacy-aware transmission scheme based on homomorphic proxy re-encryption for ndn," *Int. J. Secur. Netw.*, vol. 13, no. 1, p. 58–70, Jan. 2018. [Online]. Available: https://doi.org/10.1504/IJSN.2018.090646

[11] A. Chaabane, E. D. Cristofaro, M.-A. Kaafar, and E. Uzun, "Privacy in content-oriented networking: Threats and countermeasures," 2013.

[12] P. Moll, V. Patil, N. Sabharwal, and L. Zhang, "A brief introduction to state vector sync," *NDN, Technical Report NDN-0073, Revision 2*, 2021.

[13] C. Serban, F. Douglis, Y. Kim, D. Townley, L. Zhang, A. Afanasyev, and L. Wang, "Ndn full packet security," 2019, named Data Networking Community Meeting. [Online]. Available: https://www.nist.gov/news-events/events/2019/09/ndn-community-meeting

[14] S. Arianfar, T. Koponen, B. Raghavan, and S. Shenker, "On preserving privacy in content-oriented networks," in *Proceedings of the ACM SIGCOMM Workshop on Information Centric Networking (ICN)*. ACM, 2011, p. 19–24.

[15] Y. Zhu, Y. Tao, and R. Huang, "Beacm-dp: A broadcast encryption anti-censorship mechanism based on directory proxy," *Transactions on Emerging Telecommunications Technologies*, vol. 31, no. 2, p. e3794, 2020, e3794 ett.3794.

[16] 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 *Proceedings of the 2nd ACM Conference on Information-Centric Networking (ICN)*. ACM, 2015, p. 167–176.

[17] C. Bernardini, S. Marchal, M. R. Asghar, and B. Crispo, "Privicn: Privacy-preserving content retrieval in information-centric networking," *Computer Networks*, vol. 149, pp. 13–28, 2019.

[18] C. Partridge, S. Nelson, and D. Kong, "Realizing a virtual private network using named data networking," in *Proceedings of the 4th ACM Conference on Information-Centric Networking (ICN)*. ACM, 2017, p. 156–162.

[19] X. Liu, Q. Bing, X. Lu, L. Zhong, D. Wei, and G. Qu, "An identity privacy protection strategy in vehicle named data network," in *IEEE International Conferences on Ubiquitous Computing Communications (IUCC) and Data Science and Computational Intelligence (DSCI) and Smart Computing, Networking and Services (SmartCNS)*, 2019, pp. 818–822.

[20] S. DiBenedetto, P. Gasti, G. Tsudik, and E. Uzun, "Andana: Anonymous named data networking application," 2012.

[21] G. Tsudik, E. Uzun, and C. A. Wood, "Ac3n: Anonymous communication in content-centric networking," in *13th IEEE Annual Consumer Communications Networking Conference (CCNC)*, 2016, pp. 988–991.

[22] S. C. Seo, T. Kim, and M. Jang, "A privacy-preserving approach in content centric," in *IEEE 11th Consumer Communications and Networking Conference (CCNC)*, 2014, pp. 866–871.

[23] Y. Kim, I. J. Kim, and C. Shim, "A strategy for preserving privacy in the ccn," in *International Conference on Information and Communication Technology Convergence (ICTC)*, 2018, pp. 797–800.

[24] P. Moll, W. Shang, Y. Yu, L. Wang, A. Afanasyev, and L. Zhang, "A survey of distributed dataset synchronization in named data networking," *NDN, Technical Report NDN-0053, Revision 2*, 2021.

[25] B. Gleeson, A. Lin, J. Heinanen, G. Armitage, and A. Malis, "A framework for ip based virtual private networks," Internet Requests for Comments, RFC Editor, RFC 2764, February 2000.

[26] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," in *Proceedings of the 13th Conference on USENIX Security Symposium*. USENIX, 2004, p. 21.

[27] Z. Zhang, S. Liu, R. King, and L. Zhang, "Ndn-mps: supporting multiparty authentication over named data networking," in *Proceedings of the 8th ACM Conference on Information-Centric Networking*, 2021, pp. 83–94.