Decentralized Photo Sharing via Named Data Networking

Varun Patil, Tianyuan Yu, Xinyu Ma, Lixia Zhang
{varunpatil,tianyuan,xinyu.ma,lixia}@cs.ucla.edu
UCLA
Los Angeles, USA

ABSTRACT

The semantic naming and data-centric security paradigm of Named Data Networking opens up new possibilities for building decentralized applications. In this poster, we identify some factors that lead to centralization and non-interoperability of cloud services. We present the preliminary design of a photo sharing application as a simple example to articulate how NDN can overcome the lack of cloud-independent user identities, illustrating the process and considerations involved in designing decentralized applications that can provide the features offered by existing cloud services.

CCS CONCEPTS

• **Security and privacy** → **Systems security**; • **Information systems**; • **Networks**;

KEYWORDS

Named Data Networking, decentralization, application design, distributed system, file sharing

ACM Reference Format:
Varun Patil, Tianyuan Yu, Xinyu Ma, Lixia Zhang. 2023. Decentralized Photo Sharing via Named Data Networking. In *ACM ICN 2023 Posters and Demos (ACM ICN '23), October 9–10, 2023, Reykjavik, Iceland*. ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3623565.3623755

1 INTRODUCTION

Most user-facing applications today are cloud-based, owned and managed by private corporations. The increasing reliance of the society on a relatively small number of application service providers has led to concerns regarding centralization of control, user privacy, and data sovereignty. Although several existing packages, such as Nextcloud [1], ownCloud [2], Seafile [7], etc., allow end-users to deploy cloud-equivalent services, these packages simply replicate the cloud model, resulting in isolated small cloud instances hosted by individual users. Similar to the existing centralized clouds, there is no interoperability between multiple instances.

In this poster, we take a first step towards building decentralized apps, which is to understand what are the functions in today's applications that are provided by cloud providers. We then take a simple photo sharing app as a case study to understand on how to support the same set of functions using decentralized solutions. More specifically, we sketch a design of decentralized photo sharing running over Named Data Networking (NDN) [3]. We hope this design exercise can serve as an illustration to shed insights on what are the critical components that enable application decentralization.

2 WHAT CLOUDS OFFER FOR APPS

We identify three major functions that today's clouds offer: identity based security, rendezvous of users, and resources including computation and storage. Clouds authenticate individual users and assign them identifiers. This tight coupling of user identity to the cloud is the core reason for lock-in and centralization [4]. Because of cloud-based identities, users necessarily discover each other *inside* the clouds, and data access control are naturally handled by the clouds. Clouds also offer secured data storage, helping applications with data availability.

3 PHOTO SHARING DESIGN

From the above reasoning, we can see that, to build a simple, decentralized photo sharing app among a group of friends, one must support the following functions:

1. User naming Each user needs a unique and semantically meaningful name that can be recognized by others in the group.
2. Photo discovery In place of cloud's rendezvous function, the app needs an effective mechanism to inform everyone about new photos published by the others.
3. Photo storage The shared photos need to be stored somewhere to allow follower users to fetch them.
4. Photo access control The photos should be accessible only by authorized followers of the publisher.

3.1 Identity and Trust Relations

The photo sharing app is a social application where users form follower–followee relations managed by themselves. Therefore, a web of trust model should be used for this application. However, existing identities can still be used for bootstrapping and name assignment. For example, we can leverage the widely deployed distributed global naming system, DNS, to offer such an Internet-wide unique NDN name. If a user owns a DNS name (as many users do), then he/she can obtain a certificate of domain ownership that may be verified through any existing PKI systems, such as DNSSEC or web PKI. LetsEncrypt, for instance, provides a free and automated service to verify domain ownership which we can utilize for bootstrapping. For users who do not own a DNS name, he/she may obtain a unique identity, such as an email address, from another individual or organization that owns a DNS name. The owner of a domain name may issue the user a certificate either out-of-band or using a challenge response mechanism, which is then similarly verifiable. After certificates are obtained, users can further establish trust relations (e.g., scanning QR code) with their NDN identities.

We note here that our application only utilizes DNS and the global PKI for initial bootstrapping, as a practical means to assign a user a globally unique identity. In particular, the identities and trust relations of users are completely managed by the users themselves, and do not depend on the PKI in any way.

3.2 Discovery

The requirement of data discovery in a photo sharing app primarily arises from the requirement of a shared collection, i.e. an album of photos. Such a shared album may be updated over time with new photos. Thus, the followers of a shared album must be notified that newly shared data is available. In practice, a particular photo shared in an album can generally only be updated by its owner. We continue with this assumption for our initial design; thus we can sidestep the complicated consensus problem and focus on data availability and notification.

Distributed Dataset Synchronization (Sync) [5] serves as the transport layer of NDN and provides the function of data production notification. Sync notifies every participant of a "Sync group" about the existence of newly produced data by another group member. The problem of shared albums and discovery can be directly mapped to Sync, wherein each album forms a Sync group. Such a Sync group has the following features.

(1) All group members (i.e. users with whom the album is shared) are notified whenever any group member shares a new photo with the group. Such a notification contains the name of the newly generated data.
(2) For access control, all data published to the group are encrypted. Only the users in the group authorized by the particular data producer can decrypt the data.

We note that since all data packets produced in the photo sharing group can be saved in NDN repos, learning the name of the newly produced data suffices for consumers interested in fetching the data, and the data producer does not need to stay online.

3.3 Storage

Since data in NDN are immutable and semantically secured, the storage layer can be decoupled from the application. Further, since the network layer has multicast data delivery and in-network caching intrinsically designed in, the high performance requirement of storage is reduced by a significant degree. Several storage systems have been built over NDN including single instance repositories such as repo-ng [9], fast-repo [10] and NDNFS [8] as well as distributed storage systems such as Kua [6]. However, there are still some design questions worth looking into.

(1) Who maintains repositories? In a completely distributed system, users have full control over their data. Therefore, repositories should not be operated by the application developers, but rather be treated as infrastructure that users pay for.
(2) How do repositories interact with our system? Since we do not require synchronous communication, repositories needs to actively fetch shared data, which requires repositories to join Sync groups. This needs a new repo implementation.
(3) How to authenticate repositories? A requirement for the repo participating in the Sync group is that users should be able to authenticate the repo. This may require the repo to have an application level group-specific identity.

3.4 Access Control

Data in NDN is secured and encrypted directly, eliminating the dependency on the security of communication channels. To control the access to data, data packets are encrypted by a secret encryption key. This approach changes the problem of controlling the access to data to the problem of how to distribute the data decryption key to authorized parties only. We use Name-based Access Control (NAC) [11] as a framework in our application as follows.

(1) Each user acts as the access manager of their own photos, or any other confidential data produced by the user.
(2) All confidential Data are encrypted with a symmetric Content Key (CK), which in turn is encrypted using an asymmetric Key Encryption Key (KEK).
(3) The Key Decryption Key (KDK), i.e. the private counterpart of the KEK, is shared with all authorized users in the group by encrypting it individually with each user's public key.
(4) The encrypted CK and the encrypted KDK for each authorized user are stored in the repo as soon as it is generated, thus allowing asynchronous access.

We omit details on the naming scheme of NAC due to the lack of space, but note that it is easily extensible to multiple users. We also note that new questions arise when using access control in groups of users, such as an individual user's ability to authorize additional users to access a piece of data not, allowing newly joined users to access previously published data, and the handling of membership and access revocation.

4 SUMMARY AND NEXT STEPS

In this poster, we briefly discussed the design and process of building a real-world user-facing decentralized NDN application. With the specific example of the traditionally cloud-based photo sharing app, we noted how separating identity from cloud providers is the key requirement for the decentralized design and implementation, and how such an application can use functions such as Sync, storage and access control.

As a simple app, photo sharing only requires shared storage resources. However, implementing such an application still requires integration of multiple NDN components, which is not trivial. If one extends decentralized apps to other types, that may need processing resources as well. As our future efforts, we will explore the implementation of this photo sharing application.

REFERENCES

[1] 2023. Nextcloud. https://github.com/nextcloud/server.
[2] 2023. ownCloud. https://github.com/owncloud.
[3] Alexander Afanasyev, Tamer Refaei, Lan Wang, and Lixia Zhang. 2018. A Brief Introduction to Named Data Networking. In *Proc. of MILCOM*.
[4] Christian Huitema, Geoff Huston, Dirk Kutscher, and Lixia Zhang. 2023. Report of 2021 DINRG Workshop on Centralization in the Internet. *SIGCOMM Comput. Commun. Rev.* 53, 2 (jul 2023), 31–39. https://doi.org/10.1145/3610381.3610386
[5] Philipp Moll, Varun Patil, Lan Wang, and Lixia Zhang. 2022. SoK: The Evolution of Distributed Dataset Synchronization Solutions in NDN. In *Proceedings of the 9th ACM Conference on Information-Centric Networking* (Osaka, Japan) *(ICN*

'22). Association for Computing Machinery, New York, NY, USA, 33–44. https://doi.org/10.1145/3517212.3558092

[6] Varun Patil, Hemil Desai, and Lixia Zhang. 2022. Kua: A Distributed Object Store over Named Data Networking. In *Proceedings of the 9th ACM Conference on Information-Centric Networking* (Osaka, Japan) *(ICN '22)*. Association for Computing Machinery, New York, NY, USA, 56–66. https://doi.org/10.1145/3517212.3558083

[7] Seafile. 2023. Seafile. https://github.com/haiwen/seafile.

[8] Wentao Shang, Zhe Wen, Qiuhan Ding, Alexander Afanasyev, and Lixia Zhang. 2014. *NDNFS: An NDN-friendly File System.* Technical Report NDN-0027. NDN.

[9] NDN Project Team. 2014. *repo-ng.* https://github.com/named-data/repo-ng

[10] NDN Project Team. 2018. *Fast Repo.* https://github.com/remap/fast-repo

[11] Zhiyi Zhang, Yingdi Yu, Sanjeev Kaushik Ramani, Alex Afanasyev, and Lixia Zhang. 2018. NAC: Automating Access Control via Named Data. In *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)*. 626–633. https://doi.org/10.1109/MILCOM.2018.8599774