White Paper

Enabling Data Sovereignty: Requirements for User-Controlled Data Management and the Applicability of NDN

January 2026

Research Institute of Advanced Technology
SoftBank Corp.

Internet Research Laboratory
University of California, Los Angeles

SoftBank

UCLA Samueli School of Engineering

1. Introduction

The rapid evolution of internet services has transformed personal data into an indispensable asset underpinning modern social and economic activities. Vast amounts of information—such as medical checkup records, financial records, purchase histories, location data, behavioral logs, and user-generated content—are created daily online. For individuals who produce this data, it should be accessible and usable on their own terms. The significance of this capability becomes most evident in everyday situations—for example, when visiting a hospital, you may need to disclose your past medical records and daily exercise data to the doctor, or when filing taxes or applying for a loan, where one might need to give a preparer purpose-specific, time-limited access to statements. In both cases, we don't want to expose entire accounts. Similarly, when switching between platforms and wanting to retain a complete history of photos, messages, or playlists without losing any, or when consolidating years of health and activity records into a single, comprehensive view controlled by the user. However, people currently cannot do this; the fundamental ability to manage personal data—such as transferring records, granting purpose-specific and time-limited access, and revoking it—remains missing from people's everyday digital experiences.

At the root of these issues lies today's centrally managed data storage. The web is providing a generic platform for modern applications. In today's Web, data is tied to the application that produced it and stored on the cloud platforms on which the applications run on. For example, one's calendar data is locked in the calendar app, while the fitness data is in the fitness app, creating data silos. Consequently, data silos result in a lack of interoperability for data sharing across different applications, as well as a lack of user control over data sharing. Today, we observe that specific platform providers collect and centrally manage user data, leveraging it within their proprietary ecosystems. While this model offers convenience and efficiency, it also entails a structural imbalance: ownership and control of data are separated from individuals, leaving users unable to fully understand or manage how their data is used. This issue has become more pronounced with the rise of hyperscalers in recent years. Many people use free applications, which has led to a personal data management structure that is closed off within the application service. It becomes increasingly difficult for individuals to manage and use their personal data on their own terms.

Addressing these challenges requires a fundamental examination of the principles underlying data management and distribution. Specifically, a paradigm shift is needed toward a user-sovereign data usage model—one in which individuals are recognized as the legitimate right holders of their data, and mechanisms exist to explicitly define, share, and, when necessary, revoke data usage permissions in accordance with clear purposes, scopes, and recipients. Achieving such a model demands not only legal and governance frameworks but also a robust technological foundation to support them.

This white paper identifies the requirements of data sovereignty and articulates the challenges of

meeting them. It begins by analyzing the structural issues inherent in the current data management model and identifies the architectural requirements necessary to enable decentralized data management and user-centric utilization. It then examines a few existing solutions that aim to decentralize data storage, evaluating how well each addresses the identified requirements.

Building on this analysis, the paper explores new directions for putting control of user data into users' hands. To achieve this, we need to separate user data from both the platforms where the apps are hosted and the apps that generated the data, as the data is currently tightly bound to both. Doing so requires data storage space to house the separated data, with users controlling access to it. Such a storage system must ensure authenticity, integrity, and controllability at the network layer, allowing individuals to securely and selectively manage and share their own data. We identify Named Data Networking (NDN) [1] as a potential solution for storing user data under user control. This concept underpins and paves the way for a decentralized data ecosystem that is no longer dependent on centralized control. We conclude by identifying remaining challenges in pursuing this new direction: assigning users provider-independent DNS names and the consequential scalability issues in network delivery; maturing semantic-name-based security and privacy solutions; and the necessity of establishing a competitive market for user-controlled, application-independent storage services.

2. Current system structural issues in user data management

Current user data management faces structural challenges, as we describe below. The existing structure is rooted in centralized service architectures. Application infrastructure providers control personal data, creating silos that hinder portability and interoperability. These issues are further amplified by hyperscaler-driven ecosystems, making user-centric data control increasingly difficult.

2.1. Transfer of Data Control Due to Centralization of Application Services

Within the current hyperscale-driven ecosystem, it has become the norm for service providers to centrally store and manage user data generated by their users. As a result, the authority to manage that data is effectively held by the service providers at the moment of data production. In other words, the act of using a service itself presupposes that users lose control over their personal data to the provider.

This structural characteristic arises from the fact that service providers offer a comprehensive environment that encompasses both the application layer and the underlying operational infrastructure. As part of service delivery, user data is automatically incorporated into the provider's management domain, leaving users with little control over its storage or usage.

Such a structure inherently poses a fundamental problem: users cannot autonomously decide how their data is handled.

2.2. Loss of Portability and Interoperability Due to Data Silos

A siloed data structure causes two issues: portability and interoperability.

Portability Issue: It is technically challenging for users to import their own data from service providers unless appropriate mechanisms for data export are implemented. As a result, users cannot easily migrate their personal data—such as purchase histories, activity records, or settings—when switching to alternative services. For instance, location-based services rarely allow users to move their accumulated history or preferences to another provider, making service transitions cumbersome. Moreover, when a service is discontinued or its specifications are changed, users risk losing access to their accumulated data. This lack of portability prevents users from maintaining long-term control and continuity over their own information assets.

Interoperability Issue: Even when data remains accessible within a particular service, it is often locked into proprietary environments and cannot be referenced or utilized by other services. For example, purchase histories on one e-commerce platform are generally not interoperable with those on different platforms, preventing consistent recommendation or personalization experiences across platforms. Similarly, user data collected for specific applications cannot be flexibly reused in other contexts, limiting the potential for cross-service innovation and value creation.

Siloed data structures severely constrain the ability of personal data to serve as an informational asset. To unlock this potential, we need to redesign both institutional and technical frameworks for data portability and interoperability to enable secure, user-controlled data use across services.

2.3. Expansion of Structural Challenges by Hyperscalers

From the standpoint of development efficiency and market access, developers and small- to medium-sized enterprises are compelled to rely on the cloud infrastructures and API suites provided by hyperscalers. Under this dependency structure, the design, operation, and data management of services are all premised on the hyperscalers' environments. As a result, challenges such as the transfer of control driven by centralized architectures and the siloing of data have been further amplified and entrenched.

Hyperscalers operate an extensive portfolio of services—for example, spanning data management, user-behavior analytics, maps, and calendars. Smaller service providers that rely on these platforms are, in practice, compelled either to provide the data requested by the hyperscaler or to delegate data management to it, thereby remaining bound to the prevailing model of centralized control.

As a result, the concentration and closure inherent in these hyperscaler-centered dependencies are being extended, fostering an ecosystem in which users, developers, and smaller enterprises alike find it increasingly difficult to exercise independent control over data use.

One leading cause of this situation lies in the business models hyperscalers have developed to ensure their economic sustainability. Storing data requires a storage infrastructure, and

maintaining that infrastructure incurs costs. Hyperscalers' business models make their data infrastructures appear free to use while generating revenue through targeted advertisements and other means that leverage user data.

Overcoming the structural dependency on hyperscalers and realizing a decentralized, self-directed model of data management constitutes a key challenge for the next generation of the Internet.

3. Fundamental principles of data sovereignty

At the core of the concept of data sovereignty lies the principle that the rights to personal data should belong to the individual who generates it. Regardless of the system in which the data is stored or the entities that process it, the ultimate authority over the provision, transfer, and use of that data must remain with the individual. This principle redefines user-generated data not as a corporate asset, but as a personal informational resource to be managed in accordance with the individual's own interests and consent.

Under this framework, users must be able to autonomously determine to whom their data is provided and to what extent it is shared. Platform and service providers, in turn, are merely entrusted to use the data temporarily with explicit user authorization; they do not own or have comprehensive rights to the data. This principle represents a structural paradigm shift—from a platform-centric to a user-centric model—aimed at ensuring, both institutionally and technologically, that individuals are the primary agents in the governance of their data.

This concept aligns with the direction articulated in the European Union's General Data Protection Regulation (GDPR) [2], which enshrines the rights of data subjects. However, the data sovereignty model goes further by emphasizing not only legal protection but also the establishment of technical mechanisms that enable individuals to exercise control in practice. In this view, data sovereignty is not merely a normative declaration or regulatory ideal; it is a functional state in which individuals can actively and autonomously make choices regarding the provision and sharing of their own data.

4. Solution Requirements for Enabling Data Sovereignty

Acknowledging users' ownership of data is the first step towards user data sovereignty. The next step is to develop effective solutions to ensure users' rights throughout the data-use lifecycle. To operationalize data sovereignty, individual control and choice must be embedded in the systems that store and manage data.

The most critical component in realizing this objective is the design of the storage layer. To ensure that users retain ownership and control of their data—and to eliminate undue dependence or unauthorized access by third parties—the storage system must satisfy the following key requirements.

4.1. User Control and Decision-Making

Users must possess full authority over the storage, deletion, and sharing of their own data. To ensure this, the storage architecture should provide the following capabilities:

- Users can freely choose the storage provider for their data.
- Users can delete their data at will, without requiring external approval.
- No third party—including the storage provider—can access the data without the user's explicit permission.
- When users grant sharing permissions, they must be provided with tools that can precisely define which data is shared, with whom, and under what conditions.

These capabilities collectively ensure that, at every stage of data storage and circulation, decisions are explicitly made and authorized by the data owner.

4.2. Availability

Sustaining data sovereignty requires continuous and reliable access for users and authorized entities. To achieve this, the storage system must ensure high availability and fault tolerance through verifiable mechanisms that replicate data securely across multiple physical locations or storage providers, maintaining integrity and accessibility even in the event of node or provider failures. These capabilities eliminate single points of failure and establish a trustworthy foundation for autonomous, user-controlled data management.

Availability may be provisioned at different service levels. While some storage providers offer high availability as part of their service, others may offer only a limited degree based on their pricing and SLA, potentially requiring other entities, including users, to implement a complementary layer. Regardless of at which level it is implemented, this property remains crucial for user-controlled data management.

4.3. Independence

The storage infrastructure must be structurally independent from specific applications or service providers. The entity responsible for storing data (the storage provider) should be organizationally and operationally separated from application providers. Users may operate their own storage, or may delegate such operation to a trusted, neutral third-party storage operator.

To achieve this independence, the following conditions should be met. In addition, separating data from applications requires a standardized, unified understanding of how applications access user data.

- The storage service must be designed independently from the application service, allowing multiple services to access the same data resources and preventing data loss even when a service is discontinued.
- APIs and data access protocols should be standardized and publicly documented to enable

external entities to access data.

Through such independence, users can maintain uninterrupted ownership and use of their data, regardless of the application provider or the location where the application runs.

5. Examining the Existing Decentralized Storage Technologies

Many decentralized storage systems have been proposed and implemented recently. To explain their design approaches and understand the design trade-offs, we selected three—Interplanetary File System (IPFS) [3], Authenticated Transfer Protocol (APT) [4], and SOLID [5]—as examples of existing solutions to study. As decentralized storage systems, all three remove the dependency between data and applications and reduce reliance on hyperscaler-provided storage resources. They also differ significantly in their design goals and technical approaches. Below, we explain how each functions, their similarities and differences, and how well they meet the requirements for a decentralized storage service as outlined in the previous section.

5.1. IPFS (Interplanetary File System) and Filecoin

IPFS [3] is an anonymous, distributed storage system that aims to enable an open, peer-to-peer system operating on a voluntary, best-effort basis. Content providers host public data and run the IPFS protocol to enable requesters to fetch desired content. All peers jointly create an indexing and forwarding network to rendezvous content requesters and content providers. Contents are divided into blocks, each identified by a cryptographic hash of its contents, known as a self-certifying content ID (CID). A peer, identified by its public key hash called peerID, can be any machine connected to the Internet that can store and retrieve data blocks. Peer nodes connect through a Distributed Hash Table (DHT) protocol to facilitate CID-to-peerID mapping and forwarding. To publish content, the content provider's local IPFS instance creates a provider record linking CIDs to its own PeerID, then pushes this record to k = 20 closest peers, determined by their PeerIDs' XOR distance from the SHA256 hash of the CID. This high level of duplication ensures record availability even if some of those k peers leave the network. Because content providers may come and go, the provider record includes two parameters: the republish interval (default 12 hours) and the expiry interval (default 24 hours). If the provider remains online, it republishes its record; if not, the record is deleted after the expiry interval.

Filecoin [6] is designed to offer cryptoeconomic incentives to IPFS peers. Filecoin developed its own cryptocurrency, FIL, to encourage parties with storage resources to participate in a global decentralized storage system, such as IPFS, as independent Storage Providers (SPs). The Filecoin network has three types of players.

1. Storage providers (miners) offer hard drive space in exchange for earning FIL by storing data. They must prove continuous data retention through cryptographic proofs.
2. Clients: pay FIL to store and retrieve data. They choose SPs based on price, reliability, and location.

3. Retrieval miners: provide bandwidth and low latency to deliver data to clients.

Filecoin security relies on a blockchain system. The Filecoin blockchain provides an immutable ledger that records all storage agreements, storage proofs, penalties, and verified financial transactions. Filecoin employs two cryptographic proofs, Proof-of-Replication (PoRep) and Proof-of-Spacetime (PoSt), to ensure data is not only stored but also uniquely and continuously maintained. It incentivizes IPFS nodes to become highly motivated, financially responsible, and cryptographically verified IPFS Peers. The IPFS protocol performs the actual transfer and addressing of data, while the Filecoin blockchain enforces service-level agreements.

Note that anything published to IPFS is meant to be public data; therefore, an IPFS system does not support controlled access to user-owned content, which conflicts with the requirements of user-controlled data storage. Since peer nodes are voluntary and offer best-effort service, IPFS also cannot guarantee high content availability or persistence. Furthermore, it seems questionable whether FIL would make IPFS an economically viable storage system.

5.2. AT protocol

ATP [4] assigns each user a Decentralized Identifier (DID) as the permanent identifier. Every user is assigned a DID:plc identifier, and the URL pointing to the user's Personal Data Server (PDS) is stored as the Service Endpoint URL in the DID Document. A user signs all the data records they produce and publishes them to their PDS. Since did:plc is a flat identifier, it cannot be used directly to retrieve content. Instead, a DID:plc serves as an index to a global DID lookup table (the PLC Directory), which maps the flat ID to the user's DID Document. The document contains the user's metadata, including their cryptographic keys and the PDS Service Endpoint URL. When one changes the PDS provider, one will update the Service Endpoint URL in the global PLC Directory while keeping one's DID:plc permanently.

While the use of DID enables user data portability, it also brings the following concerns. First, since did.plc is a flat ID, there is no simple way to build a structured mapping system that indexes all the did.plc for all users. At the moment, the Bluesky company offers a global DID:plc mapping server, creating both a centralization point and a single point of failure. Second, the core mechanism of did:plc is its Operation Log (or sometimes called the "DID history log"), which contains a sequence of entries, signed by the user's private key, that record every action taken by the user regarding their identity, including the creation of the DID, the changes to the associated public key or the handle that points to the user's current PDS which stores the user's data, and voluntary action to shut down the identity. If one ever loses the private key, one can no longer update the Operation Log, rotate one's public key if it is compromised, or deactivate the account, resulting in one permanently losing control of that identity.

Some other design features of ATP also raise concerns about its suitability for supporting user data storage. First, ATP does not provide security for individual data records. Instead, user data records are stored in a Merkle Search Tree (MST); modifications to the tree root are Commit objects signed

by the user's key. Subscribers are supposed to track all changes to the entire MST and fetch all published data records for data authentication. This data authenticity design makes it difficult to use when a user wants to grant access to a specific piece of their data. Second, ATP is yet to provide end-to-end (publisher-to-subscriber) encryption for private content. In its current stage, ATP focuses on user-managed data and public content. In addition, data replication is primarily achieved by caching opportunities on subscribers' PDSs. Since a user's Service Endpoint URL points to only a single PDS, when that PDS is unavailable, access to the user's data would be significantly impacted. Finally, systems running ATP, with Bluesky as the foremost example, have yet to determine an economically viable storage solution. Bluesky currently provides users with free storage, with the eventual goal of users being sovereign over their data and having the choice (and the necessity) to pay for their PDS providers.

5.3. SOLID

The SOLID [5] design framework establishes a Personal Online Data Store (POD) for each user, emphasizing the importance of decoupling data storage from applications, regardless of how the data is generated. Users have the autonomy to select their POD providers and control access to the data they store. Applications can interact with the POD—reading and writing data—only with the user's explicit permission. However, SOLID does not explicitly address the responsibility for ensuring resilient data availability or the methods of data replication to achieve this; instead, these concerns seem to be left to the storage provider.

A commonly recognized issue with the SOLID design is the naming convention associated with users and their data. A user's WebID is derived from the URI of their chosen POD provider, and the data associated with the user falls under the namespace of this WebID. Consequently, when a user switches POD providers, they must not only migrate their data but also update their WebID and the URIs of all their data. This renaming process enables users to be independent of storage providers.

Another important point to consider is that the security of the data does not stem from the data itself; that is, data records are neither signed nor encrypted. Instead, data security depends on the security of the POD and the communication channels used.

5.4. Summary

All the studied designs decouple data from applications and from specific storage platforms. However, their designs differ on three fundamental issues: how to name users and their data, how to secure user data, and who to serve as storage providers.

Regarding naming, IPFS views anonymity as an essential factor of decentralization; thus, it uses anonymous identifiers for peer nodes and data. SOLID, on the other hand, uses semantic names for users and data, which are derived from the names of storage servers. ATP assigns users with

did:plc, a flat, unique identifier[1], which indexes into a global lookup table to find a semantic URL to access users' data. The primary difference between ATP and SOLID lies in their naming conventions for users and data. SOLID adopts URIs from the Web to name both users and data, with those URIs bound to Pod providers. This results in both users and data changing names when users switch Pods, as well as dependency on the old Pod providers for the redirection. Using pod provider names as user and data names violates the independence requirement. ATP avoids user renaming by using DID for the user identifier, with DID also used in the data names. Given that DIDs are flat numeric identifiers, their use creates a dependency on a global lookup table to map a DID to the user's PDS. The dependency on a centrally managed lookup table violates the independence requirement.

Regarding data security, we can further divide it into data authenticity and confidentiality. IPFS provides neither. ATP provides data authenticity via a Merkle Tree that must be maintained, but does not provide end-to-end data confidentiality. SOLID relies solely on existing Web security solutions (secured channels with trustworthy servers) that authenticate storage servers rather than data, and it lacks support for end-to-end data confidentiality.

Regarding storage providers, decentralization in IPFS means an open system with *voluntary participation* built on anonymous identifiers. Both ATP and SOLID let users store their data in personal data storage (Pod and PDS), which can be provided by users themselves or by storage providers. Since not all users may afford to host their own data, this model suggests the existence of a storage service market to meet storage resource needs; in fact, ATP explicitly identifies paid storage services as an eventual necessity.

Let's evaluate the above three designs based on the storage requirements for supporting user data sovereignty described in Section 4, including user-controlled decisions about data storage and access, persistent data availability, and independence of storage services from applications, centralized platforms, and provider-specific APIs. We observe that IPFS does not fully meet any of these criteria. ATP lets users choose PDS, but it primarily supports public data and lacks specific mechanisms to ensure persistent data availability. SOLID offers users choices for their PODs, but its design also lacks mechanisms for persistent data availability, and its data security relies on existing web security solutions, such as encrypted channels and trusted servers, which are inadequate in supporting fine-grained access to user data.

In the next section, we introduce NDN Repo, a new data storage service built on Named Data Networking, and demonstrate how it fulfills all three requirements for supporting user data sovereignty.

[1] Note that, although both IPFS anonymous ID and did:plc are flat identifiers, they are different. The former is a key or key hash, which is also called a self-certify identifier as the owner can prove the possession of the corresponding private key; the latter is just a unique identifier which has an associated public key storage, and the actual key pair which may change over time.

6. NDN Repo Overview

In this section, we first provide a brief overview of Named Data Networking (NDN), then explain how NDN integrates networking with storage. We then describe Repo as a realization of such storage, along with its usage. Finally, we demonstrate why NDN Repo [7] naturally meets the solution requirements of user-controlled data sovereignty.

6.1. NDN Overview

In a nutshell, a network transmits data bits. Today's TCP/IP networks identify data containers (hosts) by their addresses and deliver data by establishing connections between hosts and sending packets to destination addresses. NDN identifies data objects by their semantic names and uses those names to retrieve data at the network layer. Fetching named data enables NDN to apply security protections directly to the data, rather than to node-to-node connections. Each data item carries its semantically meaningful name in the DNS namespace, a directly encrypted payload, and a cryptographic signature by its producer, which points to its certificate by name. Data producers cryptographically sign each piece of data they generate and, if needed, encrypt it. Data consumers can verify the authenticity of all received data by checking the signature against the producer's certificate and decrypting the content with the appropriate key. NDN also maintains data immutability by making the version number the last component of the object's name; an altered data object results in a new version with a different name.

As an example, consider a calendar app running on users' local devices. A user owns her DNS name alice.me, and one of her calendar events is named alice.me/calendar/events/ndn-weekly-call/v=2, which identifies the second version of the calendar event "ndn-weekly-call" owned by alice.me. The event details are encrypted with a key managed by Alice herself, and the event data (including the name and encrypted content) is signed with her private key. This secured, named data can be stored anywhere, regardless of the trustworthiness of the underlying storage, such as either Alice's local device or her ISP-provided in-network storage. Data consumers, like the calendar event invitees, can retrieve the data from any location, verify it by checking the signature with the alice.me certificate, confirm the event's legitimacy against pre-installed policies (e.g., whether Alice is authorized to create the event), and decrypt the event details using the corresponding decryption key.

Semantically named, secured data form the fundamental building blocks of all NDN-based networked systems. Along with application content, keys, certificates, and security policies are also semantically named and secured, and can be retrieved by name just like any other data object. For example, one can retrieve Alice's certificate using the name embedded in Alice's signature. This data-centric networking approach offers many advantages, such as enabling routers to cache data as it passes through to satisfy future requests for the same data.

At the same time, networking by data fetching introduces a new problem: how can calendar invitees be notified of a new invitation so they can retrieve it? NDN addresses this problem by

developing a dataset synchronization protocol dubbed Sync, which uses network multicast to inform participants in a multi-party application of new data generations in a timely and resilient manner.

In addition to Sync, data-centric networking also requires persistent data availability. For example, to verify received data, one needs to fetch the data producer's certificate, even if the producer is offline at the time. NDN addresses this need by deploying persistent data repositories, or Repos for short. Today's applications are built on the client-server model, where servers hold all the data and are always online. NDN Repos make data available at all times, with the data decoupled from specific services, and can be used by all authorized parties. Unlike router caches, which are opportunistic data storage, Repos are managed data storage that can be installed on NDN routers or as independent storage nodes. Since NDN names and secures data directly, an NDN network can supply requested data from anywhere, whether from the original producer, a router cache, or other types of in-network storage such as Repos. NDN integrates networking with storage and alleviates the scalability challenges in content distribution in the absence of Content Distribution Network (CDN) services.

6.2. Repo Design

Repo is a managed, in-network storage with the advantages mentioned above. It adheres to the trust policies set by its manager and interacts with authorized data producers and data consumers in the network.

Overview: Four types of parties participate in the Repo operation: a Repo Manager, multiple Repo instances, data producers, and data consumers. The Repo Manager establishes the security policies for data storage in the Repo by defining acceptable name patterns for signing certificates. The multiple Repo instances work together to provide a distributed storage service, and an authorized data producer can upload data to any of them. Note that all data sent to the Repo is signed and encrypted; therefore, the Repo cannot access its content. A Repo instance stores data from its producer and replicates it across several other instances (Figure 1). Later, an authorized data consumer, equipped with the corresponding decryption key, can request the desired data by its name, and the network will fetch it from the Repo if that is the best way to access it (the producer may be offline or have worse connectivity than the Repo).

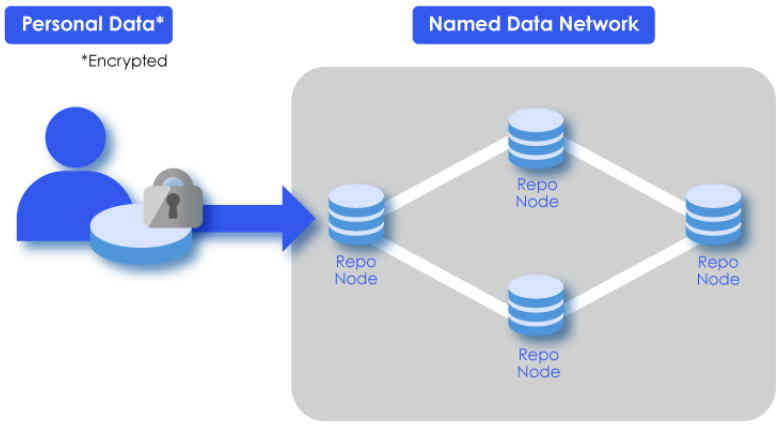


Figure 1: A User inserts encrypted personal data into the NDN Repo.

A producer invokes the service by creating a command — a named, secured data item that specifies the name of the data to be stored — and informs the Repo to retrieve the command. The closest Repo instance to the producer fetches the command, verifies the signature to authenticate the requester, and checks the command's legitimacy against the policies set by the manager (Figure 2).

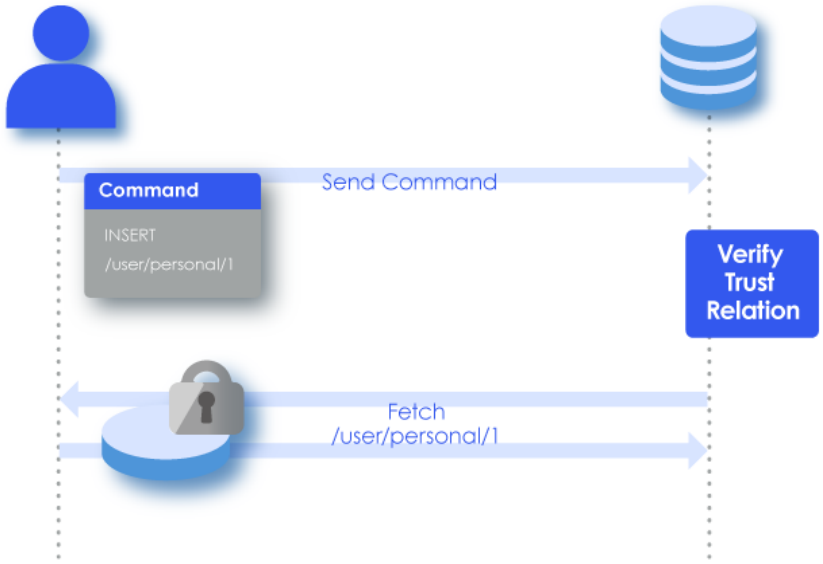


Figure 2: The flow of inserting data into the Repo.

All Repo instances belong to an NDN Sync group and collectively manage the replication state. After an instance successfully processes the command and retrieves data from the producer, it uses NDN Sync to inform all peer instances that a new data replication task is available. Then, instances will do the requested storage task until the desired level of replication is reached. When a consumer requests data by name, the network forwards the request to the closest Repo instance that has the corresponding data to satisfy it (Figure 3).

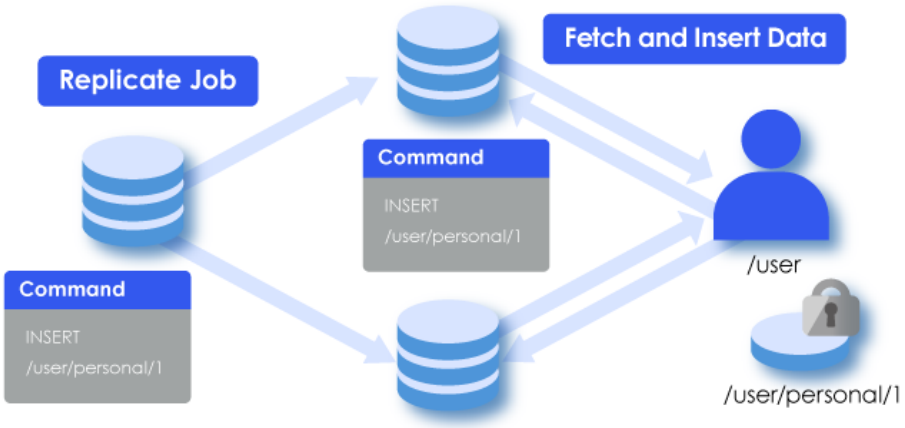


Figure 3: A storage task for a user's personal data is published to Repo's NDN Sync group. Multiple Repo instances then fetch and insert the data.

Repo Usage: Repo supports four types of commands from data producers. Besides the insert and delete verbs used in traditional key-value storage, it also accepts join and leave commands for multi-party applications that use NDN Sync. For joining, one member of the multiparty application generates a command that requests that Repo join the application's Sync group. After joining, Repo learns the group's new data generation and retrieves all data into the Repo. When the application is finished, a member issues a leave command to Repo. This provides a nearly transparent data storage solution.

6.3. How NDN Repo Meets the Requirements

User Control and Decision-Making

Users can choose which Repo service providers to store their data. They encrypt data at the desired level of granularity based on their security policies, using different encryption keys for each

data item, and share the decryption key for each data item only with its authorized consumers. This enables precise access control and safeguards against unauthorized access, even by the storage provider. Additionally, cryptographic signatures ensure the storage provider cannot tamper with the data.

Availability

Repo copies data to multiple nodes. If a node becomes unreachable, the system re-replicates its storage tasks to ensure the data is replicated to the desired level.

Independence

Repo, as a dedicated data storage service, is designed to be application-agnostic. From Repo's perspective, data produced by different applications are simply different datasets with distinct names. Repo's security policies are defined in terms of users and their certificate names; it does not understand the semantics of either the application or the data names. Repo can be run by anyone with storage resources and used by any entities that speak named, secured data.

7. Example Use Case: User-Controlled Personal Data Access to AI

Today, e-commerce sites generate recommendations from browsing and purchase histories collected within their own domains. Because providers partition this data, behavior on other sites is not taken into account. As a result, systems tend to re-suggest items that have already been purchased or fail to reflect shifts in price sensitivity and preferences.

These issues can be mitigated if browsing and purchase histories, along with other personal data, are managed on the user side rather than by services and are provided at the user’s discretion. When a user aggregates histories from multiple sites and shares only the necessary scope, AI can generate recommendations that reflect the user’s overall purchasing behavior. In addition, if the user chooses to provide daily search history and—where desired—social media interactions, responses can better account for preferences, budget considerations, and decision-making tendencies. The user retains complete control over which portion of the data to provide, can refuse unwanted sharing, and can revoke previously granted access. The concept is illustrated in Figure 4.

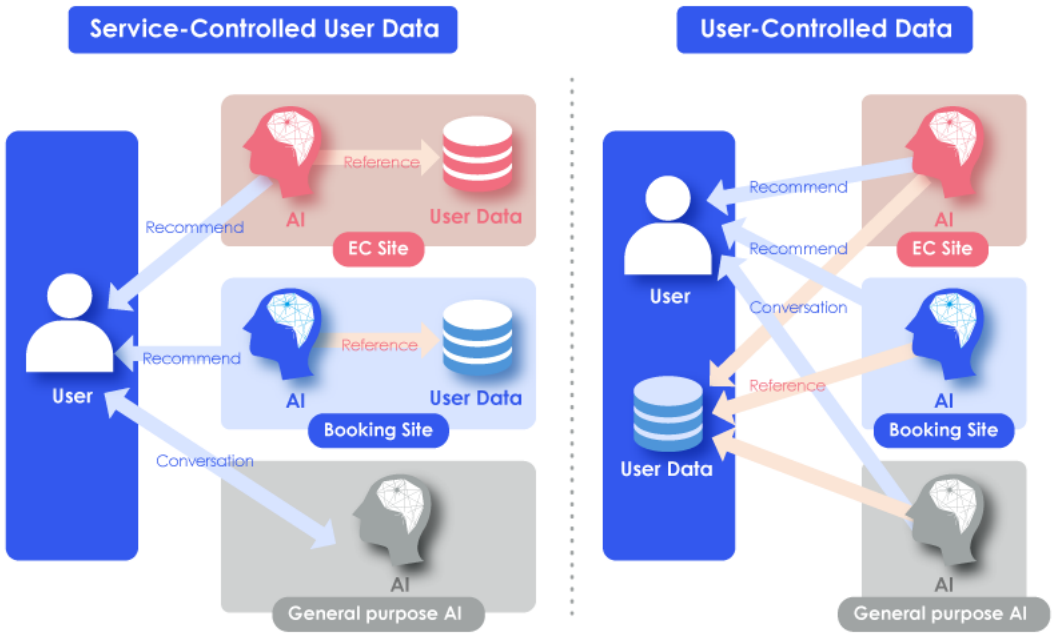


Figure 4: The concept of shifting from service-controlled data to user-controlled data management mechanism

We envision that the above objective can be effectively implemented by leveraging the NDN Repo as follows. A user, or the user's utility application, collects their own data and stores it in an NDN Repo, which can be run on the user's own storage or run by a storage provider, with the storage content managed by the user. The AI agents running on E-commerce sites can access specific user data, with the user's permission, to generate the most relevant product recommendations.

Applications utilizing user-collected and managed data access are not limited to purchasing. For travel planning, authorized browsing, and search histories can be referenced across sources to propose itinerary candidates that reflect past booking patterns and typical spending levels, while avoiding conflicts with confirmed reservations and enabling comparison of transportation, lodging, and on-site activities. For household financial management, authorized purchase histories can be aggregated to support tracking spending trends and recurring expenses, detecting potential budget overruns, and short-term spending forecasts.

By shifting the data management structure in this way, user-generated data becomes assets that users can store and share at their discretion. Letting users control their own data removes dependence on specific services or applications. It enables user-controlled AI access to personal data, maximizing benefits for users while minimizing the loss of user privacy.

The core enabler in this new direction of personal data sovereignty in the age of AI is NDN's solution for securing data directly, rather than relying on secure servers and channels. The NDN primitives provide signing, authentication, and access control for individual data objects; they

collectively constitute an effective foundation for building a decentralized, user-directed AI data ecosystem.

8. Market Changes Driven by the Separation of Data and Services

Efforts to separate applications from data and return data sovereignty to users cannot be realized without a new economic model that sustains such a concept. At the NDN Workshop 2024, Wilfred Pinfold suggested that mechanisms enabling individuals to make use of their own data could create new economic ecosystems [8]. For example, by voluntarily providing one's online shopping history, a user could more easily find desired products across multiple online shopping sites. This user-controlled personal data sharing could improve convenience for small online retailers and revitalize the online marketplace as a whole.

Similarly, in the smart mobility domain, users currently need to subscribe to multiple services, such as LimeBike, Bird, Skip, and others. If users could manage their personal and payment information, they could access various innovative mobility services seamlessly, while service providers could reach users who need their offerings. As the separation of services and data advances, demand for data management operators is expected to grow, leading to competition in the field and the emergence of storage providers specialized for various use cases.

This economic shift is crucial for fostering a competitive market for storage services, enabling users to finally realize a decentralized, user-controlled data management model finally.

9. Identified Challenges and Future Directions

During the development of this whitepaper, we identified several remaining challenges to enabling user data sovereignty and outlined future directions to address them.

The first and foremost challenge is user and data naming. The independence requirement means that user data names should persist over storage provider changes. None of the three examined designs fully solved this problem. The NDN Repo design addressed it by directly routing/forwarding on data names, which leads to the following two remaining issues. First, users need to obtain DNS names as their own identities, independent of service providers. These user names are also used to derive the names of user-produced data. Although it is feasible for users to obtain DNS names, and many do, making it a common practice requires an investigation into the tools needed to provide all Internet users with DNS names and to enable users to manage their names. The second challenge is the scalability in network routing and forwarding, given that the number of names is orders of magnitude larger than the number of IP addresses. The NDN team has developed an initial solution to address this scalability challenge by separating name prefixes from network topological connectivity [9].

The next identified challenge is the security for both user data and the overall user-controlled data management system. The NDN Repo design can provide end-to-end data authenticity and

confidentiality. However these solutions and their implementations need to be hardened via large scale trial deployment.

The third challenge is developing a new, economically viable storage service model. Hardware technologies for large-scale, high-performance storage systems are well developed and used in centrally controlled cloud storage services. What is needed is a competitive market for storage services where users can purchase reliable, cost-effective services.

10. Conclusion

This white paper analyzed the core structural challenges of centralized data management: (1) loss of user control, (2) data siloing, and (3) lack of portability, and defined the essential requirements for data sovereignty: (a) User Control, (b)Availability, and (c)Independence. Our examination of existing decentralized technologies (IPFS, AT Protocol, SOLID) found them lacking.

NDN Repo, built on Named Data Networking (NDN), is one of the possible technologies to approach the goal. NDN secures data directly with semantic names, signatures, and encryption, making data storage location-transparent and application-agnostic. This inherent security enables fine-grained access control, technically enforcing “User Control and Decision-Making” and repositioning user-generated data as a personal, shareable asset.

NDN Repo enables novel use cases, such as personalized AI, by allowing users to integrate and share their data across services at their discretion. Key challenges remain in realizing this vision, including: establishing provider-independent naming, hardening security solutions, and developing an economically viable, competitive storage service market. Addressing these challenges is vital for establishing a self-directed data management model for the future Internet. SoftBank and UCLA will continue to focus on developing this field.

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 Computer Communication Review, Volume 44, Issue 3, https://doi.org/10.1145/2656877.2656887, 2014.

[2] European Commission, "Data protection," [Online]. Available: https://commission.europa.eu/law/law-topic/data-protection_en.

[3] Interplanetary File System, "IPFS: Building blocks for a better web," [Online]. Available: https://ipfs.tech/. [Accessed November 2025].

[4] Bluesky, "AT Protocol," [Online]. Available: https://atproto.com/. [Accessed November 2025].

[5] Solid Team, "Solid," [Online]. Available: https://solidproject.org/. [Accessed November 2025].

[6] Filecoin, "A decentralized Storage Network for the World's Information," [Online]. Available:

https://filecoin.io/. [Accessed November 2025].

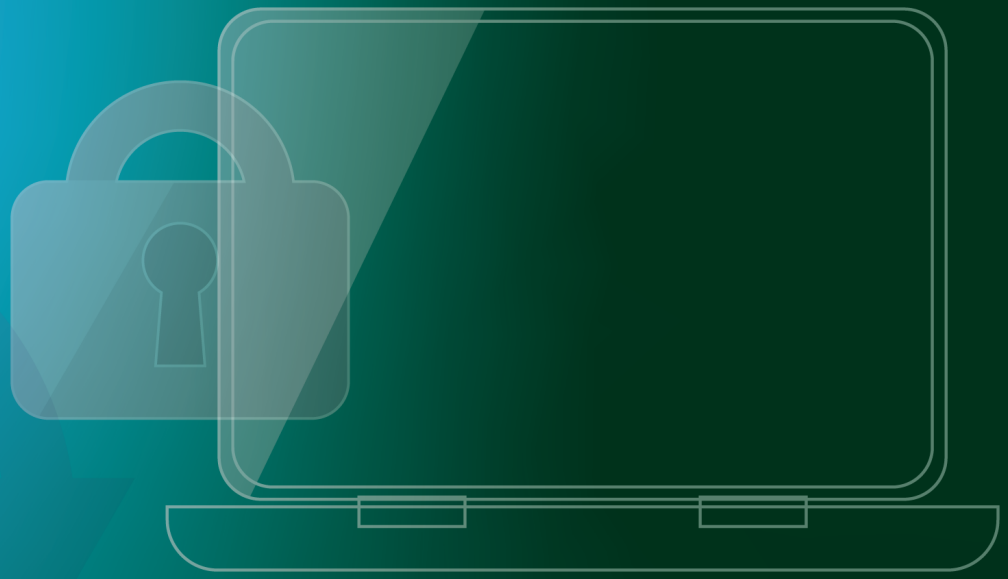
[7] T. Yu, J. Zhi, X. Ma, Y. Kocaogullar, V. Patil, R. Wakikawa and L. Zhang, "Repo: Application Agnostic and Oblivious In-Network Data Store," 2024 IEEE International Conference on Metaverse Computing, Networking, and Applications (MetaCom), https://doi.org/10.1109/MetaCom62920.2024.00052, 2024.

[8] NDN Workshop 2024, "Panel Discussion," [Online]. Available: https://cdnapisec.kaltura.com/index.php/extwidget/preview/partner_id/684682/uiconf_id/33598632/entry_id/1_yxz7cuen/embed/dynamic#t=01:46:40. [Accessed November 2025].

[9] V. Patil and L. Zhang, "Enabling Decentralized Applications: A Transport Perspective," UCLA, ProQuest ID: Patil_ucla_0031D_24256. Merritt ID: ark:/13030/m5bh4d4x. Retrieved from https://escholarship.org/uc/item/03r09055, 2025.

Acknowledgement

This white paper was collaboratively developed by the Research Institute of Advanced Technology at SoftBank and the Internet Research Laboratory at the University of California, Los Angeles, with contributions from Ryuji Wakikawa, Yuto Sekiya, Hidehiko Kawahara, Sota Sugimura, and Keiichi Shima of SoftBank, and Lixia Zhang, Tianyuan Yu, Adam Thieme, and Xinyu Ma of UCLA.



SoftBank

The Research Institute of Advanced Technology at SoftBank, established in April 2022, is committed to advancing cutting-edge technologies that drive societal progress. As an *activator* for innovation, it focuses on research in AI-RAN, 6G, HAPS (High Altitude Platform Station), autonomous driving, quantum technologies, and other forward-looking fields.

Our mission is to harness the power of technology to create solutions that address real-world challenges and shape a better future. Guided by SoftBank's corporate philosophy, "Information Revolution - Happiness for everyone," we strive to lead the next wave of advancements that improve lives and inspire progress.

For more details about our latest technological developments and initiatives, please visit our website: https://www.softbank.jp/en/corp/technology/research/