The Barriers to Overthrowing Internet Feudalism

Tai Liu
NYU-AD
tai.liu@nyu.edu

Zain Tariq
NYU-AD
zain.tariq@nyu.edu

Jay Chen
NYU-AD
jay.chen@nyu.edu

Barath Raghavan
ICSI / Nefeli Networks / USC
barath@icsi.berkeley.edu

ABSTRACT

Today's Internet scarcely resembles the mythological image of it as a fundamentally democratic system. Instead, users are at the whims of a small number of providers who control nearly everything about users' experiences on the Internet. In response, researchers and engineers have proposed, over the past decade, many systems to re-democratize the Internet, pushing control over data and systems back to the users. Yet nearly all such projects have failed. In this paper we explore why: what are the goals of such systems and what has caused them to run aground?

1 INTRODUCTION

Five years ago, Bruce Schneier noticed something curious about the state of the user-facing Internet [42]:

> Some of us have pledged our allegiance to Google: We have Gmail accounts, we use Google Calendar and Google Docs, and we have Android phones. Others have pledged allegiance to Apple: We have Macintosh laptops, iPhones, and iPads; and we let iCloud automatically synchronize and back up everything... Some of us have pretty much abandoned e-mail altogether... for Facebook.

He observed that our data, our communication tools, and, increasingly, our hardware is controlled by five companies, which he analogized to feudal lords. In pledging our allegiance, we get distinct benefits [42]:

> We choose to do it because of the convenience, redundancy, automation, and shareability. We like it when we can access our e-mail anywhere, from any computer. We like it when we can restore our contact lists after we've lost our phones. We want our calendar entries to automatically appear on all of our devices. These cloud storage sites do a better job of backing up our photos and files than we would manage by ourselves...

The de facto reality of the Internet of the 1990s and early 2000s matched its de jure architecture: a federated network of many autonomous providers with little centralized control of services or infrastructure. Today's Internet, while governed by many of the same protocols, scarcely resembles its past.

None of this is news to networking researchers and engineers who, unsettled at the notion of becoming vassals to powerful companies, have designed and built numerous systems that aim to upset this power balance and re-democratize the Internet. Over the past decade many such systems have been developed, and there has been increased interest in just the last few years.[1] While these diverse efforts do not have a unified objective, they have largely aimed to overcome the privacy, security, and reliability challenges of a feudal internet. Yet, to date, nearly all of these efforts have failed. In this paper, we seek to understand why.

What barriers remain to overthrowing the current structure of the Internet? We begin by considering the benefits and drawbacks of today's architecture. We then coalesce the objectives of various projects to identify requisite properties and fundamental components of a re-democratized Internet. We also examine how existing efforts aim to satisfy these properties and the mechanisms used to do so. Finally, we discuss what is missing both technically and otherwise, and suggest directions for future research.

2 A FEUDAL INTERNET

What are the common sought-after features of a democratized Internet, one that restores the structure of the 1980s or 1990s? The discussion in this space has become confused, so we begin by defining our terms.

Changes in the Internet's structure have taken place along two orthogonal axes over the past few decades, yet they are often conflated. The first axis concerns distribution—centralized vs. distributed—and whether the physical resources being accessed for some service are located at a single machine (at

HotNets-XVI, November 30–December 1, 2017, Palo Alto, CA, USA

ACM ISBN 978-1-4503-5569-8/17/11…$15.00
https://doi.org/10.1145/3152434.3152454

[1] The notion of building a re-decentralized Internet has become popular: it was a central plot device in the TV show *Silicon Valley*, though their system was made possible by a magical compression algorithm.

one extreme) or dispersed across many machines all over the planet (at the other). The second axis concerns control—democratic vs. feudal—and whether the authority over the service and the machines providing a service is spread across many individuals or organizations or held by a few.

The Internet of today is quite different from that of a few decades past along both axes: it has gone from partially-centralized and democratic to distributed and feudal.[2] Our aim is to move towards distributed and democratic: not to undo the necessary trend towards wide distribution, but to disperse control. Put another way, the scale-out design philosophy that has served us well in the design of systems over the past two decades must now be applied to the control of systems as well [39].

2.1 Feudal Internet Features

Our primary focus is decentralizing administrative control of various systems. Before we do so hastily, we believe it is important to understand the reasons for today's centralized administrative control (which leads to a feudal Internet). Centralized systems are attractive to users due to:

Convenience. Compared to running one's own service, cloud services are always on, accessible, fast, secure, scalable, sharable, and easy to setup, use, and maintain.

Homogeneity. Most users are on the same set of platforms so users are familiar with how to use them, and have a reasonable understanding about how they work. Homogeneity also produces important network effects, particularly in social applications.

Cost. Cloud services are cheap; most services are offered for free to end users, at least initially.

These considerable benefits are what make cloud services so popular and difficult for users to escape from despite their negative consequences to user's freedom and privacy. Eventually, sufficient users will have bought into the platform that inertia (e.g., losing/migrating all personal data, network effects) makes the cost of opting out or switching services impractical. Once users are effectively locked in, they are easily monetized.

However, centralized systems are attractive not only to users but also to systems designers and operators because of:

Performance. It has proved easier to scale Internet services when they are under centralized administrative control. New protocols and architectures, hardware and software—all of which can be designed in concert to improve performance—can be rolled out systematically. This has proved to be the case in datacenter design, where a single organization can tailor its systems to the needs of the services it runs.

[2]It may seem strange to describe the Internet of the past as partially centralized, but numerous services were indeed centralized. Consider the way the Web and FTP services were run in the 1990s when the average person didn't and couldn't afford to operate an always-on server. These servers were generally hosted by ISPs (centralized), of which there were hundreds to thousands (semi-democratized).

Security. Centralized authority simplifies security policies: instead of data coming from other (distributed) nodes that are assumed to be untrustworthy, other nodes can be assumed to be cooperative and trustworthy, making the design of distributed algorithms and systems simpler. In addition, it is easier to uniformly and rapidly secure systems under centralized command.

Financing. The cost of aggregated infrastructure can benefit from economies of scale, which potentially lowers the cost of providing a service. Today's financial incentives encourage the aggregation and monetization of users and their data.

2.2 Goals

A democratized Internet would need to provide the benefits that users have come to expect of today's Internet: the convenience of not having to maintain one's own infrastructure, the homogeneity of use across many devices and from different locations, and the low costs of both of these. On the other hand, there is little need to meet the needs of researchers and engineers who design and build such infrastructure—they (we) are responsible for designing systems and we can choose to build alternative systems. However, financial constraints are a key limiting factor for democratized Internet service architectures, something we return to later.

3 RE-DEMOCRATIZING THE INTERNET

A slew of recent systems have been developed in academia, industry, and as open source projects in attempts to democratize key Internet services [2, 4, 6, 7, 14, 16, 18, 18, 22, 23, 27, 27–30, 34, 35, 40, 41, 43, 44, 47, 50, 52, 54]. In Table 1, we roughly categorize these recent systems by the central problem(s) they aim to solve. This list is by no means exhaustive, but does represent a set of relatively well-known projects. There is some overlap, but the core problems they tackle fall into four categories: naming, group communication, data storage, and serverless web applications. In this section, we describe how recent efforts approach these problems, how they compare with past systems, and what pieces are missing.

3.1 Name Registration

Three mechanisms are commonly used to represent user identities on the Internet: public keys, personal information, and pseudonyms. Public-key-based identities consisting of opaque strings help preserve privacy and are considered relatively secure; however, such identities have faced usability barriers for as long as public-key cryptography has existed. Since none of these three basic mechanisms are simultaneously usable, secure, and privacy preserving by themselves, a name (or pseudonym) is combined with a public-key to yield a secure, human-meaningful identity. Centralized public key infrastructures (PKIs) rely on trusted certification authorities (CAs) or a Web of Trust (WoT) to issue and authenticate digital certificates. Existing PKIs relying on CAs or a WoT suffer from well-known security, trust, and revocation weaknesses (e.g., centralized administrative control, CA compromises,

Decentralization Problem	Recent Projects
Naming	Namecoin, Emercoin, Blockstack
Group Communication (e.g., public messaging and social networking)	Matrix, Riot, Ring, Nextcloud, GNU social, Mastodon, Friendica, Identi.ca
Data storage	IPFS, Blockstack, Maidsafe, Secure-scuttlebutt, Nextcloud, Sia, Storj, Swarm, Filecoin
Web applications	Beaker, ZeroNet, Freedom.js

Table 1: Decentralization problems and examples of recent projects

WoT Sybil attacks, etc.). The literature is replete with research on identity management, naming systems, PKIs and the fundamental tradeoffs that exist [17, 21, 24, 46, 53].

Motivated by these problems and by the rise of Bitcoin [33], several recent decentralized alternatives to centralized PKIs have been developed that leverage advances in blockchain technology, including: Namecoin [34], Emercoin [14], and Blockstack [2, 3]. By providing cryptographically auditable, append-only ledgers, blockchains allow users to publicly register a name with associated metadata in a decentralized manner. Compared to centralized systems, blockchains essentially trade scalability and performance for global consensus and security. These blockchain-based naming schemes manage to resolve Zooko's Triangle [25] by providing, simultaneously, human-meaningful, secure, and decentralized names. However, blockchains suffer from some well-known problems, including: the 51% attack, limits on data storage, wasteful mining computation, the endless ledger problem, and many others (see [2, 31]). Since name registration does not require high bandwidth or large amounts of data to be stored, those two weaknesses are mitigated in this use case, but the other challenges remain.

3.2 Group Communication

For the purposes of our discussion, we consider both group messaging and online social networking as group communication problems since they are roughly analogous and share several requirements. Public messaging or publicly-accessible social networks (e.g., Reddit, Twitter) exacerbate scalability, security, and privacy challenges.

Group communication and sharing platforms have been around for decades. Usenet, one of the oldest messaging platforms on the Internet, offered a decentralized (federated), distributed online forum with some privacy. No one other than a user's service provider knew the user's real identity and there was no single authoritative entity that owned the network or controlled its content. Usenet eventually collapsed under its own traffic load and suffered from issues relating to the storage of undesirable content.[3] The shift toward centralized systems managed to resolve performance and abuse-related problems, but have created a new set of challenges. In addition to the features from Section 2.1, messaging and social networking systems should provide the following communication-specific features:

[3]This relates to an ethical debate that we do not explore in this paper.

Connectedness. Users should be able to communicate with others in the face of node failures, node attrition, loss of communication channels, etc.

Abuse Prevention. Platforms should have mechanisms that handle abuse, however abuse is defined (e.g., spam, hate speech, brigading, etc.). This property becomes more salient as the scope increases and becomes more public.

Privacy. No identifying information about users should be revealed to an unauthorized entity.

Today's most popular messaging and online social networks (OSN) can achieve the first two properties due to their centralized infrastructural and administrative nature. However, the connectedness of centralized platforms depends completely on the prerogative of platform operators. For example, if it is no longer profitable to provide service to a user or they "misbehave", access to the platform can be unequivocally revoked and personal data rendered inaccessible. The norms for "good behavior" and the definitions of abuse are dictated by platform operators or their delegates. One way messaging and OSNs cope with abuse is through moderation. However, moderation is often in direct tension with freedom of expression, and can be influenced by governments, other powerful organizations, or individuals in positions of authority. Finally, it is well-known that due to the profit motive of these platforms, service is provided in exchange for user's private data and attention.

Centralized platforms have made some progress keeping user data private from other *users*, but they have simultaneously continued to violate user privacy as their monetization strategies grow more sophisticated. For example, user data today is mined for social profiling, monetized directly (e.g., via advertising), or sold to third-party organizations.

These numerous challenges have sparked many efforts to consider democratized communication platforms while attempting to maintain a comparable level of service. We categorize these systems into two network models: socially-aware P2P and federated.

Socially-aware P2P Systems

Academic systems like PrPl [45], Persona [5], Lockr [49] and OTR (Off The Record messaging) [9] have mainly focused on protecting user privacy. Rather than uploading data to geographically distributed servers as in Usenet—which also comes at a cost of service availability as servers often

face huge number of connections and temporarily refuse new connections—PrPl [45] allows users to retain ownership over their data by storing it on home servers or in encrypted form on public storage providers. PrPl [45] lets users define access levels, i.e, some users (trusted nodes or "friends") are allowed to access private data while others only have access to public data. Persona [5] and Lockr [49] provide similar functionality, but take this further by allowing users to define relationships with other users and ensuring that relationships are not exploited. OTR [9] introduces the concepts of repudiability and forgeability to the discussion.

Since privacy is the main focus of these systems, they do not emphasize high service availability or controlling abuse. In PrPl, trusted users can obtain certificates to access the data directly from the storage, without the need to go through "Butlers" to maintain accessibility. Persona [5] claims to provide a relatively high level of service availability, but data and metadata are not coupled together, which can harm availability in the event of node failures. Lockr [49] and OTR [9] are designed to maintain user ownership and privacy of data at the cost of service availability.

We call these systems socially-aware peer-to-peer (P2P) systems because they require users to define social trust relationships with other users. Compared with peer-to-peer (P2P) systems, centralized systems have the performance, security, and financial benefits described in Section 2.1. Socially-aware P2P networks improve some of the security and privacy drawbacks of traditional P2P networks since users are communicating with other users that they trust. However, this comes at a price of reduced availability since nodes accept connections only from socially-trusted peers. Furthermore, establishing these networks can be tedious for the user as it can be challenging to quantify social index value of the relationships between nodes.

Federated Systems

Many recent non-academic projects that decentralize group communication place more emphasis on freedom rather than solely focusing on privacy. These applications are nearly all built on a federated model.

Riot [41] is a chat application which is based on Matrix, a federated network protocol. Matrix [30] provides high availability by replicating data over the entire network and ensures privacy by using end-to-end encryption techniques like the double ratchet algorithm [37]. Although messages are encrypted, metadata is still accessible and readable by the Matrix server that stores it, which slightly compromises the level of privacy by revealing the identities of the participants of an exchange. Unlike centrally administered systems, every application built on Matrix can define its own abuse moderation policies and implement them on the application level.

GNU Social [18] is another federation-based social networking application that relies on OStatus [36] for federation. OStatus allows real-time exchange of messages between nodes, but there are no intrinsic privacy mechanisms; privacy must be implemented at the application level. Mastodon [29] also runs on OStatus [36] and provides similar functionality to GNU Social, but also allows federations to define their own rules on abuse (e.g., racism, sexism, xenophobia, violence, gender discrimination, etc.). Unlike Matrix [30], OStatus-based applications are bottlenecked by single servers that can cause entire instances to be inaccessible if they fail.

Identi.ca [23] and Friendi.ca [16] are based on the popular federated stream server, pump.io [38]. Pump.io [38] makes it easy to disseminate information in the network, and uses OAuth to restrict unauthorized access to private data. Different servers are capable of providing their own functionalities. Friendi.ca [16] provides its own application-level privacy measures through private one-to-one and group messaging, expiring old data and giving users ownership of their data.

3.3 Data storage

Storing data is a fundamental function of many Internet services. Although messaging systems also technically store data, for the purposes of this discussion we consider systems that primarily focus on storage. Compared with centralized storage systems, decentralized systems potentially provide cheaper storage services (since users already have devices), and are potentially resistant to censorship and unauthorized access. However, despite these advantages, scaling decentralized storage systems is hard and such systems often perform poorly across all dimensions.

There is a large amount of literature, mostly from the era in which peer-to-peer systems were popular, on distributed storage systems [1, 8, 10, 12, 20, 26, 51]. To improve performance, these storage systems typically maintain replicas and make decisions about synchronous or asynchronous replication, numbers of maintained replicas, mechanisms of replica production, locations of replica storage, redundancy monitoring, and repair strategies to prevent data loss. These design decisions involve inherent trade-offs among durability, availability, consistency, and performance of decentralized storage. All of these systems focus on physically distributed rather than administratively democratized storage and do not account for malicious nodes.

In contrast, blockchain mechanisms are completely decentralized; many storage systems build on top of blockchains in some way. Table 2 summarizes some recent systems that provide decentralized storage. We observe that, with the exception of IPFS and MaidSafe, many of these decentralized storage systems use blockchains to publicly record contracts and to facilitate payments. Here a contract is an object that defines a service agreement between two parties: storage providers and consumers. The exact contents of this agreement vary from system to system, but, generally, it contains information about storage and retrieval (e.g., how much data should be stored, how often the data should be retrieved), pricing, and proof-of-storage requirements. This use case, like in naming, uses blockchains for its intended purpose (as a slow, but consistent and verifiable public ledger) while minimizing any impacts of its weaknesses.

	Blockchain Usage	Incentive Scheme
IPFS	None	Bitswap Ledgers
MaidSafe	None	Proof-of-resource Distributed transaction
Sia	Blockchain-based contract	Proof-of-storage
Storj	Facilitate payments (storjcoin)	Proof-of-retrievability
Swarm	Ethereum blockchain for domain name resolution, payments, and content availability insurance	Proof-of-storage: SWEAR
Filecoin	Facilitate payments (filecoin)	Proof-of-replication Proof-of-spacetime Proof-of-work
Blockstack	Bind domain name public key and zone file hash	N/A

Table 2: Comparison of Surveyed Storage Systems.

In P2P storage systems nodes must contribute storage and bandwidth and cooperate with each other to store and serve data. However, selfish nodes can interfere with this sharing model if they do not have incentives to behave correctly, since sharing consumes their own resources and degrades their own performance. To address this problem, systems such as Filecoin [27], Sia [50], MaidSafe [28], Storj [52], and Swarm [47] use blockchains to build-in incentives for data storage. Essentially, nodes that wish to store and retrieve data pay other nodes for storing and serving data for them. Nodes are therefore incentivized to contribute storage and bandwidth and to cooperate (and compete) with each other to make the storage system function as a whole. Blockchain mechanisms such as proof-of-work have inspired many variations of storage-focused proof-of-work mechanisms such as: proof-of-storage, proof-of-retrievability, proof-of-replication, and proof-of-spacetime [27, 47, 50, 52]. Proof-of-Replication, for example, allows a node to convince others that they are storing exactly the same number of copies as they have claimed instead of creating multiple identities, and storing data just once (Sybil Attacks), of fetching from others (Outsourcing Attacks), or of generating on-demand (Generation Attacks) [27]. These mechanisms are designed to incentivize nodes to amass as much storage and bandwidth as possible, and to stay online as long as possible. In our table, the only exception is Blockstack, which does not focus on decentralizing storage; its users are to use the data store of their choice, such as from a cloud provider.

3.4 Web Applications

Today, the web service platforms owned by a few large companies, such as Google, Amazon, and Microsoft, provide most of the necessary components for the web (storage, database, computation, content delivery, management, etc.) to web applications and guarantee high service quality, e.g., Amazon Elastic Compute Cloud (Amazon EC2) promises 99.95% availability for each Amazon EC2 Region [13]. Many application developers decide to directly host their web applications on these existing mature web service platforms instead of devoting significant time and money to develop their own. Consequently, many web services are further bound to a few large web service providers and thus censoring of or censorship by the service providers is sufficient to disrupt the web applications running on them.

Recent decentralized systems replace the traditional client-server web architecture with a novel browser-based web architecture in which decentralized applications are no longer hosted by specific servers and are mainly run on the client. This browser-based web architecture typically leverages existing technologies such as HTML5, and combines them with decentralized name registration, communication, and storage mechanisms to allow users to easily create, modify, and share hostless web applications.

Freedom.js [43], for example, uses a browser-based web architecture where a web application, including its back-end logic, runs entirely in a web browser. Three types of APIs, the identity, storage, and transport, are provided to application developers. It leverages existing techniques, e.g., WebRTC is used to establish a direct peer-to-peer connection to transmit data, and a reliable DHT can be selected to store data globally. ZeroNet [54] is a decentralized web platform in which web applications are seeded and served by visitors via the BitTorrent protocol. When a new web application is created by the application developer, the application developer gets a public key pair. The public key is the new site address which can be looked up on trackers or DHTs, and every file of and update about the web application can be securely verified by verifying the corresponding signature. The public key is also a standard Bitcoin address for accepting donations and payments directly to the web application. Beaker [6] is a tailored web browser enabling users to create and host websites directly from browsers. Like ZeroNet [54], resources in Beaker are served and distributed in a peer-to-peer network. What distinguishes Beaker is that, motivated by Git, it explicitly allows forking and merging web applications, advocating openness at the code level.

4 INFRASTRUCTURE FEASIBILITY

Infrastructure is often overlooked by designers of the systems we have considered in this paper; it is assumed that the resources to run democratized services exist. Thus a basic unanswered question is: even if an ideal democratized Internet service architecture were to be developed, would the capacity exist for it to operate at service levels comparable to today?

Here we perform a back of the envelope calculation. We compare the resources of today's cloud infrastructure with the currently-unproductive capacity of distributed infrastructure (e.g., personal devices). We focus on three resources: 1) bandwidth, 2) compute, and 3) storage. For simplicity, we focus on a specific provider's resources, Google, and then scale up to estimate global capacity. No public data exists on

	Cloud Infrastructure	User Devices
Bandwidth	200 Tbps	5000 Tbps
Cores	400 M	500 M
Storage	80 EB	210 EB

Table 3: Estimated capacity of global cloud infrastructure and unused user resources (server-equivalent cores).

Google's network or compute capacity. Various reports from a few years ago [19, 32] estimate that Google has about 1 million servers and 10 EB of storage. We might extrapolate that today Google has about 100 million cores and 20 EB of storage. One recent estimate [48], puts the current rate of Internet traffic at a little over 200 Tbps in 2016. Since Google estimates that it handles one quarter of the Internet's traffic [15], we scale up these figures by a factor of 4, yielding an aggregate estimate across all cloud providers of 200 Tbps of bandwidth, 400 million cores, and 80 EB of storage.

Does there exist enough unproductive capacity among user devices to meet this resource demand? There are roughly 2 billion personal computers, 2 billion smartphones, and 1 billion tablets in use worldwide [11]. We assume that personal computers have an average of 2 unutilized cores and 100 GB of free storage, smartphones have 1 unutilized core and negligible free storage, and tablets have 1 unutilized core and 10 GB of free storage. However, mobile devices such as smartphones and tablets cannot be relied upon to do compute given battery constraints. Thus we estimate that only storage—about 210 EB—is available across all devices. For compute, we take the 4 billion cores available across personal computers and reduce their estimated capacity by a factor of 8 to account for weaker processors (versus server CPUs) and to allow for power management, yielding 500 million server-equivalent cores. Finally, we must estimate the bandwidth available across these devices. Assuming devices are connected to the Internet using a slow broadband connection that has only 1 Mbps upstream bandwidth, in the case of personal computers, and slow 3G connections that also have 1 Mbps upstream bandwidth for mobile devices, this yields 5000 Tbps of bandwidth. We summarize these estimates in Table 3. Roughly speaking, there appears to be sufficient capacity among existing devices.

5 DISCUSSION

Infrastructure feasibility is only a preliminary sanity check that democratizing Internet services is possible. Clearly, there are many difficult challenges in terms of performance and robustness of decentralized approaches. However, we believe that blockchains, like bittorrent and DHTs before them, are key components of recent democratized services, and are a promising avenue of research that we should work on. Improving blockchains and leveraging its properties for applications that do not mind their weaknesses are important avenues for future work. However, we also believe that we need work that goes beyond blockchains to overthrow Internet feudalism.

5.1 Easy Problems

Studying the performance and security of blockchain-based systems: hacker communities dedicated to such efforts have made headway in developing many blockchain-based systems, but have typically neglected performance evaluation and a thorough evaluation of their security models under new requirements (e.g., when used to back a storage system).

Doing what systems researchers do best: design, build, and evaluate new systems and primitives for building performant decentralized systems.

Eliminating single points of failure in federated approaches: federated approaches are an ideal stepping stone from today's feudal model, in that they allow explicit control of the granularity of a domain. However many of these systems have not been architected with canonical systems goals in mind, such as fault tolerance.

5.2 Moderate Problems

Overcoming the mismatch between research/engineer objectives and user needs: systems often solve hard or exciting problems when users' needs and desires are more mundane (e.g., getting systems researchers to attend to usability of complex systems).

Bridging the gap between the research community and the hacker community: many efforts, such as those in federated group communication systems, do not provide significant privacy features, and often do not leverage the latest thinking from the academic network security and privacy community. We can build mechanisms or toolkits that can be plugged in by the hacker community in their projects.

Grappling with infrastructure quality vs. quantity: as we discuss above, there exists more than enough unproductive capacity among user devices worldwide. However, the quality of this infrastructure is much poorer than what a typical datacenter provides. As such, systems must be designed to cope with the intermittency, higher failure rates, and variable performance of user-device-based infrastructure.

5.3 Hard Problems

Incentivizing (financially or otherwise) development of democratized Internet systems: significant engineering hours go into building Google, Facebook, etc., so building alternatives may require similar engineering efforts.

Decoupling authority from infrastructure: how can systems be designed that are not rigid about the infrastructure they run upon while still retaining user control? Technical approaches may include "guerrilla" tactics such as running encrypted services on the cloud.

Preventing the re-emergence of feudalism: while there is a long road to re-democratizing the Internet in the first place, systems must prevent backsliding to the feudal model. Unfortunately, like the other problems in this class, this may not be an entirely technical problem as centralization is frequently driven by economies of scale.

REFERENCES

[1] Atul Adya, William J Bolosky, Miguel Castro, Gerald Cermak, Ronnie Chaiken, John R Douceur, Jon Howell, Jacob R Lorch, Marvin Theimer, and Roger P Wattenhofer. 2002. FARSITE: Federated, available, and reliable storage for an incompletely trusted environment. *ACM SIGOPS Operating Systems Review* 36, SI (2002), 1–14.

[2] Muneeb Ali, Jude C Nelson, Ryan Shea, and Michael J Freedman. 2016. Blockstack: A Global Naming and Storage System Secured by Blockchains.. In *USENIX Annual Technical Conference*. 181–194.

[3] Muneeb Ali, Ryan Shea, Jude Nelson, and Michael J Freedman. 2017. Blockstack: A New Decentralized Internet. (2017).

[4] An encrypted IPv6 network using public-key cryptography for address allocation and a distributed hash table for routing. 2017. https://github.com/cjdelisle/cjdns.

[5] Randy Baden, Adam Bender, Neil Spring, Bobby Bhattacharjee, and Daniel Starin. 2009. Persona: an online social network with user-defined privacy. In *ACM SIGCOMM Computer Communication Review*, Vol. 39. ACM, 135–146.

[6] Beaker: A peer-to-peer Web browser. 2017. https://github.com/beakerbrowser/beaker.

[7] Juan Benet. 2014. Ipfs-content addressed, versioned, p2p file system. *arXiv preprint arXiv:1407.3561* (2014).

[8] Ranjita Bhagwan, Kiran Tati, Yuchung Cheng, Stefan Savage, and Geoffrey M Voelker. 2004. Total Recall: System Support for Automated Availability Management.. In *Proceedings of NSDI*.

[9] Nikita Borisov, Ian Goldberg, and Eric Brewer. 2004. Off-the-record communication, or, why not to use PGP. In *Proceedings of the 2004 ACM workshop on Privacy in the electronic society*. ACM, 77–84.

[10] Byung-Gon Chun, Frank Dabek, Andreas Haeberlen, Emil Sit, Hakim Weatherspoon, M Frans Kaashoek, John Kubiatowicz, and Robert Morris. 2006. Efficient Replica Maintenance for Distributed Storage Systems.. In *NSDI*, Vol. 6. 4–4.

[11] Consumer Electronics | Statista. 2017. https://www.statista.com/markets/418/topic/485/consumer-electronics/.

[12] Peter Druschel and Antony Rowstron. 2001. PAST: A large-scale, persistent peer-to-peer storage utility. In *Hot Topics in Operating Systems, 2001. Proceedings of the Eighth Workshop on*. IEEE, 75–80.

[13] Elastic Compute Cloud (EC2). 2017. https://aws.amazon.com/ec2/?nc2=h_m1.

[14] Emercoin - Distributed blockchain services for business and personal use. 2015. https://emercoin.com/2015-01-15-Emercoin_Peering_Agreement_with_OpenNIC.

[15] Blair Hanley Frank. 2017. Google's Espresso networking tech takes SD-WAN to internet scale. http://www.networkworld.com/article/3187589/cloud-computing/googles-espresso-networking-tech-takes-sd-wan-to-internet-scale.html.

[16] friendica - A Decentralized Social Network. 2017. http://friendi.ca/.

[17] Ali Ghodsi, Teemu Koponen, Jarno Rajahalme, Pasi Sarolahti, and Scott Shenker. 2011. Naming in content-oriented architectures. In *Proceedings of the ACM SIGCOMM workshop on Information-centric networking*. ACM, 1–6.

[18] GNU Social. 2017. https://gnu.io/social/.

[19] Google's Datacenters on Punch Cards. 2017. https://what-if.xkcd.com/63/.

[20] Andreas Haeberlen, Alan Mislove, and Peter Druschel. 2005. Glacier: Highly durable, decentralized storage despite massive correlated failures. In *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2*. USENIX Association, 143–158.

[21] Seungyeop Han, Vincent Liu, Qifan Pu, Simon Peter, Thomas Anderson, Arvind Krishnamurthy, and David Wetherall. 2013. Expressive privacy control with pseudonyms. *ACM SIGCOMM Computer Communication Review* 43, 4 (2013), 291–302.

[22] Hyperboria. 2017. https://hyperboria.net/.

[23] Identi.ca. 2017. https://identi.ca/.

[24] Van Jacobson, Diana K Smetters, James D Thornton, Michael F Plass, Nicholas H Briggs, and Rebecca L Braynard. 2009. Networking named content. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*. ACM, 1–12.

[25] Dan Kaminsky. 2011. Spelunking the Triangle: Exploring Aaron Swartz's Take On Zooko's Triangle. https://dankaminsky.com/2011/01/13/spelunk-tri/.

[26] John Kubiatowicz, David Bindel, Yan Chen, Steven Czerwinski, Patrick Eaton, Dennis Geels, Ramakrishan Gummadi, Sean Rhea, Hakim Weatherspoon, Westley Weimer, et al. 2000. Oceanstore: An architecture for global-scale persistent storage. *ACM Sigplan Notices* 35, 11 (2000), 190–201.

[27] Protocol Labs. 2017. Filecoin: A Decentralized Storage Network. https://filecoin.io/filecoin.pdf.

[28] MaidSafe - The New Decentralized Internet. 2017. https://maidsafe.net/.

[29] Mastodon: A GNU Social-compatible microblogging server. 2017. https://github.com/tootsuite/mastodon.

[30] Matrix.org. 2017. http://matrix.org/.

[31] Andrew Miller, Ari Juels, Elaine Shi, Bryan Parno, and Jonathan Katz. 2014. Permacoin: Repurposing bitcoin work for data preservation. In *Security and Privacy (SP), 2014 IEEE Symposium on*. IEEE, 475–490.

[32] Rich Miller. 2011. Report: Google Uses About 900,000 Servers | Data Center Knowledge. http://www.datacenterknowledge.com/archives/2011/08/01/report-google-uses-about-900000-servers/.

[33] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system.

[34] Namecoin. 2017. https://namecoin.org/.

[35] Nextcloud. 2017. https://nextcloud.com/.

[36] Ostatus Community Group. 2017. https://www.w3.org/community/ostatus/wiki/Main_Page.

[37] Trevor Perrin and Moxie Marlinspike. 2016. The Double Ratchet Algorithm. *GitHub wiki* (2016).

[38] pump.io. 2017. http://pump.io/.

[39] Barath Raghavan. 2015. Abstraction, indirection, and Sevareid's Law: Towards benign computing. In *Proceedings of LIMITS*.

[40] Ring. 2017. https://ring.cx/en/news.

[41] Riot - open team collaboration. 2017. https://about.riot.im/.

[42] Bruce Schneier. 2012. When It Comes to Security, We're Back to Feudalism - Schneier on Security. https://www.schneier.com/essays/archives/2012/11/when_it_comes_to_sec.html.

[43] William Scott, Raymond Cheng, Arvind Krishnamurthy, and Thomas Anderson. 2016. freedom.js: an Architecture for Serverless Web Applications. (2016).

[44] Secure scuttlebutt: A database of unforgeable append-only feeds, optimized for efficient replication for peer to peer protocols. 2017. https://github.com/ssbc/secure-scuttlebutt.

[45] Seok-Won Seong, Jiwon Seo, Matthew Nasielski, Debangsu Sengupta, Sudheendra Hangal, Seng Keat Teh, Ruven Chu, Ben Dodson, and Monica S Lam. 2010. PrPl: a decentralized social networking infrastructure. In *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond*. ACM, 8.

[46] Diana Smetters and Van Jacobson. 2009. *Securing network content*. Technical Report. Technical report, PARC.

[47] Swarm. 2017. http://swarm-guide.readthedocs.io/en/latest/index.html.

[48] The Zettabyte Era: Trends and Analysis. 2017. http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html.

[49] Amin Tootoonchian, Stefan Saroiu, Yashar Ganjali, and Alec Wolman. 2009. Lockr: better privacy for social networks. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*. ACM, 169–180.

[50] David Vorick and Luke Champine. 2014. Sia: Simple Decentralized Storage. https://www.sia.tech/whitepaper.pdf.

[51] Sage A Weil, Scott A Brandt, Ethan L Miller, Darrell DE Long, and Carlos Maltzahn. 2006. Ceph: A scalable, high-performance distributed

file system. In *Proceedings of the 7th symposium on Operating systems design and implementation*. USENIX Association, 307–320.

[52] Shawn Wilkinson, Tome Boshevski, Josh Brandoff, James Prestwich, Gordon Hall, Patrick Gerbes, Philip Hutchins, Chris Pollard, and Vitalik Buterin. 2016. Storj a peer-to-peer cloud storage network. https://storj.io/storj.pdf.

[53] George Xylomenos, Christopher N Ververidis, Vasilios A Siris, Nikos Fotiou, Christos Tsilopoulos, Xenofon Vasilakos, Konstantinos V Katsaros, and George C Polyzos. 2014. A survey of information-centric networking research. *IEEE Communications Surveys & Tutorials* 16, 2 (2014), 1024–1049.

[54] ZeroNet: Decentralized websites using Bitcoin crypto and the BitTorrent network. 2017. https://zeronet.io/.