Making Federated Networks More Distributed

Alex Auvolat
Univ. Rennes, Inria, CNRS, IRISA
alex.auvolat@inria.fr

***Abstract*—Federated networks such as Mastodon or Matrix have seen rising usage thanks to their ability to provide users with good privacy and independence from large service providers, while retaining the familiar model of server-backed websites or mobile apps with its advantages of speed, availability and ease of use. However such systems are fragile since each individual server of the federation is a single point of failure for its users.**

We argue that new secure distributed algorithms could be conceived and applied without changing the server-backed nature of the system, and that such a configuration would provide systemic resilience and better independence of end users from their service providers, without sacrificing privacy, availability, efficiency or ease of use.

***Index Terms*—distributed system, online social networks, federation of servers², privacy, zero-knowledge architecture, CRDT**

I. Introduction

Mainstream online applications rely on massive centralized platforms that have demonstrated several important modes of failure such as non-respect of user privacy, vendor lock-in, abuse of authority and conflicts of interest. Acting upon the observation that centralized platforms owned by large corporations cannot be trusted with users' personal data, the academic community as well as the Free and Open Source Software movement have produced many alternative systems. Federated networks [1], [2] are such systems that have caught up in usage thanks to the familiar model they provide to users of access through simple and efficient websites or mobile apps.

However we argue that federated networks are inherently fragile, since they rely strongly on each individual server of the federation to realize critical tasks such as identity management, access control and data management. The diversity of providers involved augments the probability of crash failure, data leak and Byzantine behaviour by a provider. Moreover since many such servers are run by volunteers, the probability of failure of the social support structure (lack of funding, lack of members' involvement) is also high.

To correct these issues, authority and trust must be removed from the individual servers, allowing for redundancy without coordination. To realize this vision, we suggest an architecture where:

- Centralized user directories at each server of the federation, which are used to identify users and authenticate their actions, are replaced by a distributed scheme based on public key cryptography where users detain their identity themselves in the form of a private signing key.
- Authoritative copies of objects (such as a user's set of messages) held by a single server are replaced by a distributed data model where any node may produce updates, as long as they are signed by an authorized user, and may do so without coordination with other nodes thanks to eventually consistent data types such as CRDTs [3].

In this manner, users may store their data on several servers to ensure redundancy. When any issue arises with a specific server, they may easily switch to another server node. The process of selecting server nodes and switching from one to another could be completely automated and invisible to the user.

In this paper, we propose a preliminary exploration of this architecture and of its impacts on privacy and usability.

II. A Decentralized Data Model

To make the architecture we propose possible, we must define a new distributed data model that allows for eventual consistency and gives authority to the users. We propose a data model where data is stored in objects that have a defined access control policy as well as CRDT semantics for conflict-free merges, enabling eventual consistency without coordination. Applications may combine many objects with different parameters in order to provide familiar functionalities such as chat rooms, news feeds, discussion forums or private messages. In order to protect user privacy, only encrypted data is stored in these objects and applications have a key distribution mechanism so that only authorized clients and no server may obtain the encryption keys [4], [5].

The CRDTs that have been studied up to date do not incorporate a permission model: all nodes that participate in the system are allowed to update the data. In order to implement a permission model, verification of user identity must be achieved. This can be achieved in a decentralized fashion using public key cryptography where end users' devices hold the private keys. Thus all updates produced to a shared object are signed by the key of a user, and the validity of updates is proved by such a signature, and not by authority of a server.

The simplest example of such an object is a set of messages where only messages signed by an authorized user may be stored and no message may be deleted. Conflict resolution consists simply in exchanging missing messages from one side

This work has been partially supported by the French ANR projects DESCARTES n. ANR-16-CE40-0023, and PAMELA n. ANR-16-CE23-0016. I wish to thank my advisors, François Taïani and David Bromberg, for their feedback and support.

2575-8462/19/$31.00 ©2019 IEEE
DOI 10.1109/SRDS47363.2019.00058

to the other. All applications can be implemented over such a simple abstraction [6], [7], however the ever-growing nature of the data makes its use impractical.

To allow for the data to shrink, updates must be allowed to supersede previous events. However allowing any user to produce an update that erases previous events allows for abusive deletion of information, which must be prevented. In order to implement safely any specification of update and delete rights without a trusted authoritative server, a set of users could be designated as verifiers whose validation is required before any data is terminally erased. A collective verification scheme has already been proposed by Frientegrity [8] in the context of an untrusted central server. We propose to revisit this scheme in a distributed setting where events do not form a linear stream ordered by a central server but form a DAG of events produced in a decentralized fashion such as in the Matrix network [2]. In such a scheme, each event of the DAG would contain a reference to the whole state of the object at that point, for instance in the form of a Merkle tree [9]. The validity of such a state would be proved by a small number of predecessors in the event DAG, under the hypothesis that no more than a certain number of trusted validators are Byzantine [1]. Thus nodes of the network could safely delete older events and states and reclaim unused space.

III. Impacts on Privacy

With standard end-to-end encryption [4], [5], neither the server nor an unauthorized eavesdropper may read the content of messages. However metadata such as the timing of messages and identity of participants in a conversation can still be leaked. Strong metadata privacy requires costly approaches such as Vuvuzela [10].

Unfortunately, our proposed scheme for CRDTs with verified states is at odds with metadata privacy, since updates must be signed by a valid key from a set of authorized users, thus revealing participant identity and activity to the servers. However we propose several means to improve data protection:

- A honest server node will communicate object data to another node only when authorized by a user that has read rights to that object. Thus metadata may leak only if one user accesses the object through a malicious or compromised server.
- The client generates different key pairs for every object with which the user interacts, so that analysis of the data at rest reveals limited information on the social graph.
- If required, clients can connect to servers through an anonymous network such as Tor, making the link between a pseudonymous identity and the physical person of the user untraceable.
- Our data model is generic and not limited to the federation of servers model. It could be used in more restrictive network topologies such as friend-to-friend networks.

[1] In the case of a permissioned network with a small number of known participants, tolerance to one or a few Byzantine node is already much stronger than zero Byzantine tolerance.

A detailed analysis of the properties of our system under a defined thread model remains to be done. While our primary aim is not to provide stronger privacy than existing solutions such as to end-to-end encrypted Matrix, an interesting avenue of research consists in finding appropriate cryptographic techniques for ensuring CRDT safety in a sufficiently general model while revealing as little as possible on the identity of participants.

IV. Usability Considerations

Being based on a client/server model, our system could be implemented as a traditional website where all encryption is done by the JavaScript client code and keys are kept in the browser's local storage. With the help of servers for storing, serving and disseminating information, the performance loss compared to a regular centralized or federated service could be minimal. Lightweight devices such as smartphones could also connect to the service without needing to store or exchange large amounts of data. Thanks to compact proofs of object state validity, clients need not trust the servers in any fashion and can verify the provided data at a limited cost.

Identity and key management remains an issue for non-technical users, however we believe that good user interface design can help with this task. Mechanisms and user interfaces to ease key management is an actively researched area in the context of cryptocurrencies, with solutions such as social key recovery or hardware wallets.

V. Perspectives and Future Work

Future works will consist in formalizing our system and providing a detailed technical analysis of the privacy and safety properties with respect to the goals we have set. In this regard, precise threat models and attack scenarios will have to be defined and analyzed. A first prototype could then be implemented and evaluated from an efficiency point of view. In addition to micro-benchmarks, direct comparisons with systems such as Secure Scuttlebutt (SSB) [6] could be made for instance by crawling the SSB network and replaying the whole set of events in the different systems that are evaluated.

References

[1] `https://joinmastodon.org/`.
[2] `https://matrix.org/`.
[3] M. Shapiro, N. Preguiça, C. Baquero, and M. Zawirski, "Conflict-Free Replicated Data Types," in *SSS*, 2011.
[4] S. Taheri-Boshrooyeh, A. Kupcu, and O. Ozkasap, "Security and Privacy of Distributed Online Social Networks," in *ICDCS Workshops*, 2015.
[5] A. De Salve, P. Mori, and L. Ricci, "A survey on privacy in decentralized online social networks," *Computer Science Review*, 2018.
[6] `https://www.scuttlebutt.nz/`.
[7] `https://github.com/orbitdb/orbit-db`.
[8] A. J. Feldman, A. Blankstein, M. J. Freedman, and E. W. Felten, "Social Networking with Frientegrity: Privacy and Integrity with an Untrusted Provider," in *USENIX Security*, 2012.
[9] A. Auvolat and F. Taïani, "Merkle search trees: Efficient state-Based CRDTs in open networks," in *SRDS*, 2019.
[10] J. Van Den Hooff, D. Lazar, M. Zaharia, and N. Zeldovich, "Vuvuzela: Scalable private messaging resistant to traffic analysis," in *SOSP*, 2015.