

# Jabber/XMPP

**25 years of Digital Independence**



**Daniel Gultsch - August 24, 2026**

## Infrastructure

“We should own our infrastructure.” A lot of people would instinctively nod in agreement with that statement. Yet who “we” refers to shifts depending on the type of infrastructure. Highways, railways, bridges, and ports require nation-scale efforts. The water supply is usually put into the hands of municipalities. And the desire to own infrastructure goes down to a much smaller level: Owning your home is a dream for many—though such ownership doesn’t necessarily have to be organized on an individual level. Instead, cooperatives or city-owned housing<sup>1</sup> can provide similar benefits.

China’s neo-colonialism, which manifests, among other things, as building and buying infrastructure in sovereign nations, is rightly criticized by many. Not selling your water supply to Nestlé is a universally accepted principle, and landlords are one of the most hated classes.

For a long time, Europe has not held digital services to the same standard. In part, this can be explained by Europe implicitly including American corporations in a collective “we”— an assumption that officially fell apart under the current Trump regime, but should have been regarded with skepticism well before then. Corporations are not our friends. However, the larger factor at play is that Europe simply did not consider digital services infrastructure. While anti-Americanism is en vogue again and drives much of the digital sovereignty movement, Europe must be careful not to simply replace American corporations with European ones, but to strive towards collective ownership instead.

Under capitalism, profit-oriented companies will always play a part in building and even operating our infrastructure. However, they need to be forced into a position where they are easily replaceable. It’s acceptable to hire a company to build a road, but when it comes to maintaining and repairing it half a century later, we need to be able to hire a different company for the job. It’s acceptable to hire a company to build and operate the backbone transmission lines, but we don’t want that company to own the entire power grid. We want smaller players to be able to connect to and interoperate within the grid. That’s where open standards come in.

---

<sup>1</sup>[https://en.wikipedia.org/wiki/Housing\\_in\\_Vienna](https://en.wikipedia.org/wiki/Housing_in_Vienna)

The Internet used to be — and to some degree still is — built around standards. A data center operator can buy servers from one company, switches from another, routers from a third, and connect them to a backbone internet provider that runs hardware from yet another company. If a company goes out of business or shifts to anti-consumer practices, the next generation of hardware can easily be ordered from a different vendor. The need for and the benefits of this supply chain independence are easily understood even by people who don't operate data centers for a living. However, when it comes to communication tools, even the tech-literate fail to apply the same critical scrutiny.

After breathing, eating, and procreating, communicating is probably the fourth most important thing humans do. Yet we often fail to recognize our communication tools as part of our infrastructure.

Digital rights advocates often point to Signal, Wire and Threema as examples of communication tools developed and operated by entities with slightly more ethical business practices than their Big Tech counterparts. What most privacy enthusiasts fail to understand is that these companies are still in the business of operating walled gardens with no escape. They do not interoperate. It's not that Signal has done something inherently malicious — although paying its CEO close to a million dollars a year and running its servers on AWS are certainly questionable — it's that we don't have a hedge in place if it ever does.

Open-source software is orthogonal to this problem. It helps to ensure that the software isn't spyware — unlike WhatsApp and other Meta products<sup>2</sup> — and that the end-to-end encryption is sound, but it does not protect us if Signal shuts down its servers tomorrow or ceases EU operations<sup>3</sup>. Open-source alone is not sufficient to meet the requirements we should have for our infrastructure.

To live up to the standards we set for ourselves, we need to design systems in which self-hosting is structurally possible but not strictly necessary. Like owning a home, running your own server should be possible, and so should

---

<sup>2</sup><https://localmess.github.io/>

<sup>3</sup>[https://mastodon.world/@Mer\\_edith/112535616774247450](https://mastodon.world/@Mer_edith/112535616774247450)

collective ownership. Digital systems can and should replicate the advantages of cooperative housing alongside those of individual ownership.

Treating digital communication as true infrastructure can only be achieved by adopting and mandating open standards.

The Extensible Messaging and Presence Protocol (XMPP)<sup>45</sup> is a standard for communicating online. It wasn't created to fit a particular zeitgeist or address the current political climate. In fact, its roots go back more than 25 years.

## Standards

Interoperability and vendor independence are achieved by setting and adhering to standards. To avoid individual vendors pushing standards that explicitly or implicitly exclude potential competitors or otherwise give unfair advantages, standards-developing organizations (SDOs) are set up for mutual cooperation, and usually have safeguards in place that prevent a single company from becoming too powerful. Well-known examples of such organizations include the ISO, the IETF, the W3C, and the Unicode Consortium.

There is a distinction to be made between a vendor publishing its API and allowing others to use it, and stakeholders coming together to collectively develop a standard within the framework of an SDO. Organizations like the IETF succeed because they force different people with different needs to agree. Protocols aren't dictated by the priorities of a single company; instead, they are reviewed and tested by competitors, security researchers, and independent developers.

Element, formerly known as Riot and NewVector, develops an instant messaging product with a feature set — such as self-hosting and federation — similar to that of XMPP-based solutions. Notably, however, it chose not to adopt XMPP, but instead published its own API under the name Matrix for others to use. Unlike with traditional standards, Element maintains tight control over any modifications or additions to its public API. Key leadership positions in the Matrix Foundation are predominantly held by current and

---

<sup>4</sup><https://www.rfc-editor.org/rfc/rfc6120.html>

<sup>5</sup><https://www.rfc-editor.org/rfc/rfc6121.html>

former Element employees. Getting outside contributions accepted into the specification is notoriously difficult.<sup>6</sup> Yet European public administrations, in their push for digital sovereignty, routinely fall into the trap of procuring such single-vendor platforms, confusing an open-source codebase with an open standard.

It's natural for standard proposals to originate within a single organization. JMAP, a modern replacement for IMAP and SMTP Submission, which is not too dissimilar from Matrix — a JSON API over HTTP — started within Fastmail before being brought to the IETF. Jabber started out as an open-source community project before it was brought to the IETF and renamed to XMPP. Ideas start small, but to create a standard, outside feedback, collaboration, and the structure of an SDO are needed.

For consumers, the difference in the approaches of Fastmail and Element is striking. Not only was JMAP noticeably improved on a protocol level while going through the IETF working group process, but it now has at least three independent servers and numerous independent client applications. Matrix, on the other hand — despite dating back to the same era around 2014 — is still stuck with one predominant reference implementation and a second alternative still in its infancy and struggling to gain traction. Operating that reference implementation is notoriously resource-intensive, which makes self-hosting difficult for smaller organizations and individuals. Element sells closed-source plugins to speed up performance.

## **The X in XMPP**

The origins of XMPP — which started out as Jabber — go back over a quarter of a century. The original RFC<sup>7</sup> dates to October 2004 and only received minor revisions in March 2011<sup>4</sup>. Requirements for instant messaging will naturally change over a time span that long. Luckily, the X in XMPP stands for Extensible, and extensions provide a way for the protocol to adapt and change over time. Extensions to XMPP are called XMPP Extension Protocols (XEPs) and are managed by the XMPP Standards Foundation (XSF). The XSF

---

<sup>6</sup><https://github.com/matrix-org/matrix-spec-proposals/pull/4174>

<sup>7</sup><https://www.rfc-editor.org/rfc/rfc3920.html>

doesn't write extensions itself; rather, it provides the framework of an SDO for developers to propose and standardize their own.

Adapting to changing requirements hasn't always been smooth sailing. XEP-0198 (Stream Management), an extension crucial for preventing message loss in mobile deployments, was stabilized in 2009, but only gained widespread implementation around 2014–2015. The iPhone was released in 2007; the HTC Dream, the first commercial Android phone, followed in 2008. OMEMO (XEP-0384), XMPP's specification for industry-standard end-to-end encryption, gained traction from 2016 onward, three years after Edward Snowden<sup>8</sup> exposed the NSA's global surveillance and put the need for E2EE on the map. The articles "The (Sad) State of Mobile XMPP in 2014" by Georg Lukas<sup>9</sup> and "The State of Mobile XMPP in 2016" by this author<sup>10</sup> illustrate this rocky transition into the mobile era.

This demonstrates that merely having specifications is not enough. Standards need to be backed by multiple, preferably independent, implementations. Today, the XSF keeps track of the implementation status of its XEPs<sup>11</sup>. This data helps authors and the XSF guide proposals through their lifecycle, such as determining the right moment to advance an XEP from Experimental to Stable. It also allows developers to easily identify other clients and servers that support a given specification for interoperability testing. Finally, by providing a reverse lookup of which software supports which features, it helps end users find the right client for their needs.

Modern clients like Dino on Linux or Conversations on Android are on par with alternatives built on proprietary protocols. Recent additions to the feature set include emoji reactions, cross-device read-state synchronization, and time zone indicators to avoid messaging contacts during their local night hours. A unique feature among self-hostable instant messaging solutions, which sadly became relevant after a state-sponsored attack on a public XMPP provider<sup>12</sup>, is channel binding, a mechanism to prevent certain machine-in-the-middle attacks.

---

<sup>8</sup>[https://en.wikipedia.org/wiki/Snowden\\_disclosures](https://en.wikipedia.org/wiki/Snowden_disclosures)

<sup>9</sup>[https://op-co.de/blog/posts/mobile\\_xmpp\\_in\\_2014/](https://op-co.de/blog/posts/mobile_xmpp_in_2014/)

<sup>10</sup><https://gultsch.de/posts/the-state-of-mobile-xmpp-in-2016/>

<sup>11</sup><https://xmpp.org/extensions/>

<sup>12</sup><https://notes.valdikss.org.ru/jabber.ru-mitm/>

Looking to the not-too-distant future, the XMPP community is currently working on message replies, gallery-style multi-image sharing, and OAuth support. All of these features already have experimental XEPs backing them, but the community is currently awaiting implementation experience before advancing them. Meanwhile, the community is also exploring options for updating the RFC and bringing the protocol back to the IETF as “XMPP 2.0.”

Instant messaging is not a homogeneous user experience. A messenger for teams might require a different feature set than something optimized for use with friends and family. Not every XMPP client aims to provide the same user experience, but the standards exist for developers to build whatever specialized client their users need without inventing a protocol from scratch.

## **A future in the Past**

There is something fascinating about the fact that XMPP has developers in its community who are younger than the protocol itself. It has quietly outlived venture-funded startups, proprietary platforms, and entire tech cycles. That endurance provides the resilience we need in challenging times. It is the anchor, the backbone, the infrastructure.

Matrix reinvented the wheel as a rubber-tyred metro. On paper, it provides real benefits, such as climbing steeper inclines, which are then used to aggressively advertise and lobby local governments to buy in. But in the end, the municipality gets locked into a single vendor.

A changing geopolitical situation and the realization that Big Tech holds too much power lead us to seek out and develop alternatives. But what if the alternative has been right under our noses for over 25 years? The standard for instant messaging — RFC 6120: Extensible Messaging and Presence Protocol (XMPP).

## About standard federated instant messaging

In this text, Daniel Gultsch, maintainer of the Conversations XMPP client for Android, celebrates the 25 years of the XMPP RFC standard, reminding us why we cannot trust a single actor with our communications.

Despite all its limitations, the Jabber/XMPP ecosystem retains today vibrant non-profit communities and projects for smaller hosts independent of the tech giants.

# NO SLOP EDITIONS

