tinySSB

convergent, authenticated data structures
for challenged communication environments

EmComm – Swiss Convention 2024
Oct 19, 2024

Prof Christian Tschudin HB9HUH/K9CFT

University of Basel

Overview

tiny Secure Scuttlebutt (SSB)

a post-Internet communication architecture

A) Why

(rethinking networking and distributed programming)

B) What tinySSB can do

(secure, offline-first collaborative apps)

C) How tinySSB works

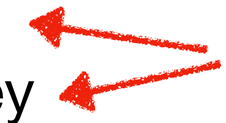
(signed hash-chains replicated over anything + convergent data structures)

A) Why

Servers are everywhere
(web server, file server, DNS server, email server, database server …)

Problems: Servers
-> introduce dependency
-> require intermediate parties to provide the service (ISP, registrar, cloud …)

Intermediares can (and usually do) turn against you:
Example 1: your ISP not allowing incoming connection requests
Example 2: Musk threatening to switch off Starlink for Ukraine
Example 3: SMS switched off after the big earthquake in Turkey



EmComm

Can we create a network, without built-in intermediaries?

A) Why … and how to approach

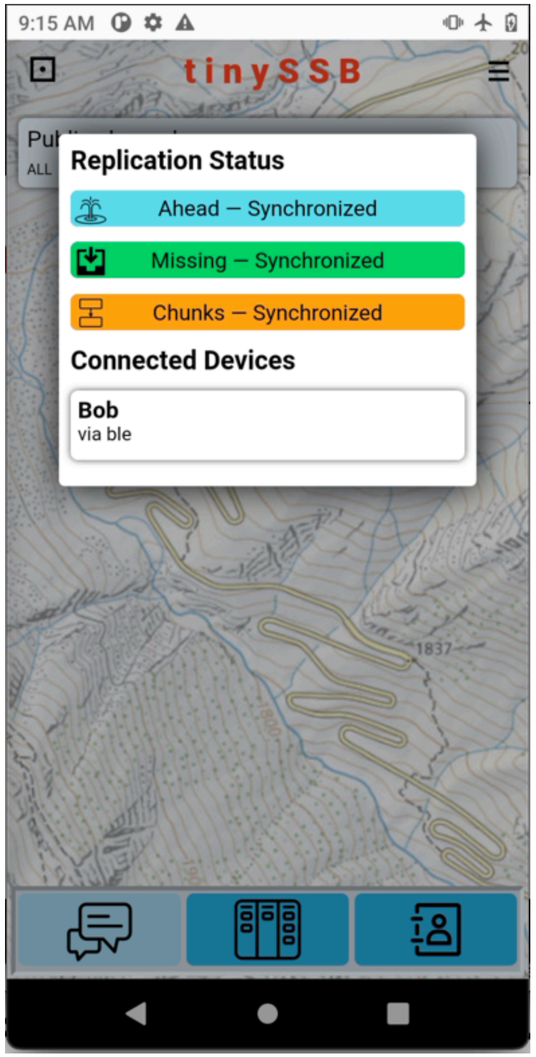
From prev slide: **Can we create a network, without built-in intermediaries?**

… and without HAMs having to offer EmComm services?

Goal: military-grade, bottom-up networking where ALL intermediaries are optional

How? **A post-Internet approach:**
a) enable offline-first applications (no need to be online)
b) enable direct data flow among end devices (typically smartphones w/ BLE)
c) enable opportunistic choice among *“connectivity assists and forwarders”*
-> LoRA, PMR, HF, … Internet if available …, device mobility, even USB sticks!

tinySSB devices



User interface on Smartphones:
- coupled via config-less Bluetooth Low Energy (BLE)
- coupled via config-less **BLE-to-LoRa bridges** (as replication mesh)

User interface on LoRa devices:

Scenario: spread content by walking around with your LoRa watch

B) What tinySSB offers, eventually

New generation of distributed applications … which do NOT need any servers!

- “social media without servers” (chat groups, voice msg, can be e2e-encrypted)
- “Google docs without servers” (collaboration tools like Kanban, blogging)
- “board games without servers” (chess, Settlers of Catan, Jass?)
- “banking without servers” (cryptographically protected “local tokens”)

—> come and see us at the demo booth!

C) How tinySSB works

Three techniques
(that were not available when the Internet was born)

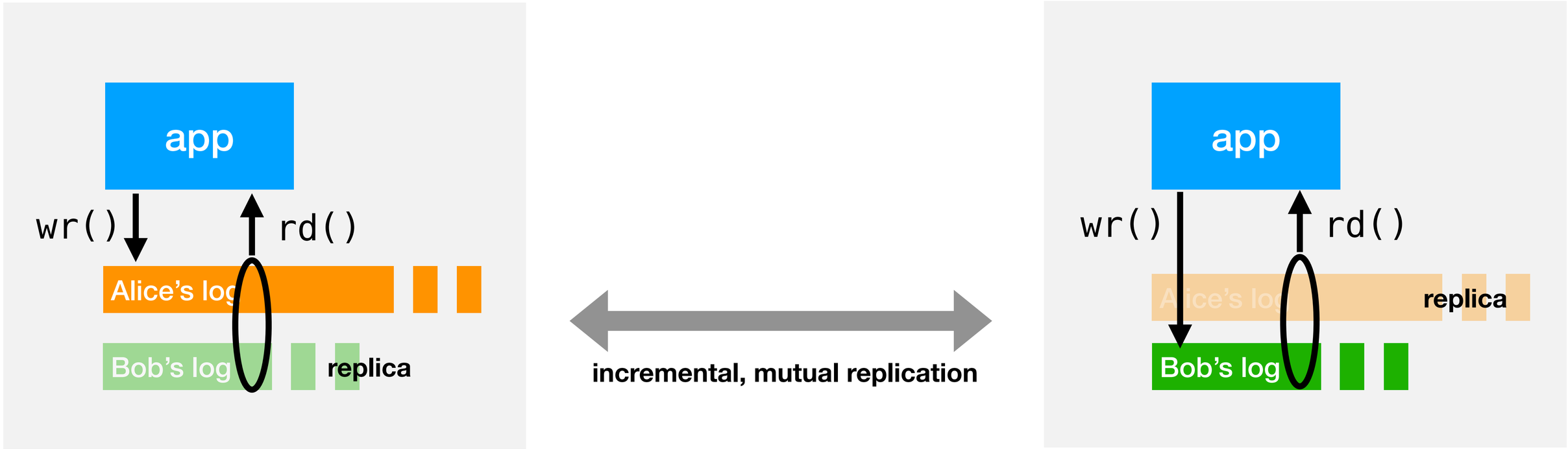
1. Memory in the network
   —> delay-tolerant “log replication”

2. Digital signatures
   self-sovereign identities and digital signatures —> trustless replication

3. Conflict-free Replicated Data Types (CRDT)
   —> synchronization of application data is guaranteed to converge

C.1) Log Replication (no “TCP ports”)

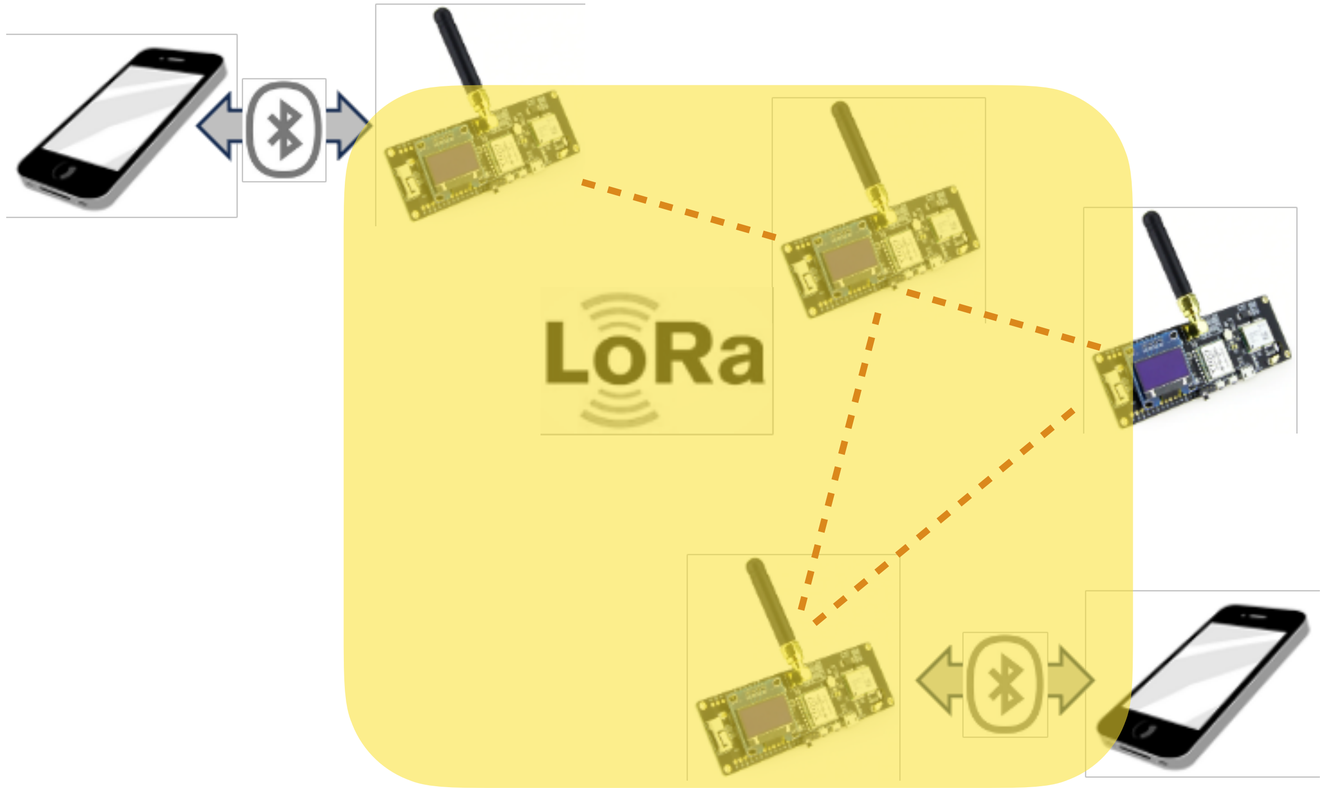
app-to-log *log-to-log* *app-to-log*



- Unlike the Internet, SSB has no (direct) Inter-Process Communication (IPC): SSB apps can only interact with local replicas of logs, no notion of destination.

C.2) Digitally signed append-only logs

- SSB is based on “single-author append-only logs”, each entry is digitally signed



- no need to trust forwarders
- forwarders can verify whether new data is genuine

- Append-only logs are implemented as a secure hash-chain
  —> data source cannot change past events
  —> long-term trust, no need for Certificate Authorities (Web certificates)

C.3) Convergent Data Structures

Research breakthrough in 2011

Shapiro et all from INRIA, France: **Conflict-Free Replicated Data Types (CRDT)**

- Replaces server-centric approach (Google-Docs “operational transform”, 2006)
- Many useful distributed apps can be rewritten to not needing a server
- Provable guarantees: even if (a) working off-line and (b) assuming arbitrary delivery order of updates, at the end, all participants will have the SAME value of a shared global variable (“strong eventual consistency”)

C.3') Convergent Data Structures

CRDT cannot replace *all* distributed data structures

Problems not working well with CRDTs:

- preventing double-spending - this includes Bitcoin
- continuous DB consistency (is replaced by "*eventual* strong consistency")

Another drawback:

- CRDT-design and proofs are not trivial, is actively researched

Impressive progress: CRDTs for collaborative editing of any JSON document, nowadays several libraries available: Automerge, Yjs

tinySSB summary

a post-Internet protocol stack, pure Secure Scuttlebutt (2014):

- SSB, but very tiny (all packets are 120 Bytes long)
- anything goes: Internet, adhoc LoRa, Bluetooth LE, shortwave … even USB sticks
  + minimal config: no device addresses - just 1 self-sovereign identity per author
- memory-in-the-network: intermittent connectivity ok, no end-to-end delivery path required
- decentral:
  - social-media *without* servers (and Kanban and games and …)
  - build-your-own-trust: all data cryptographically signed

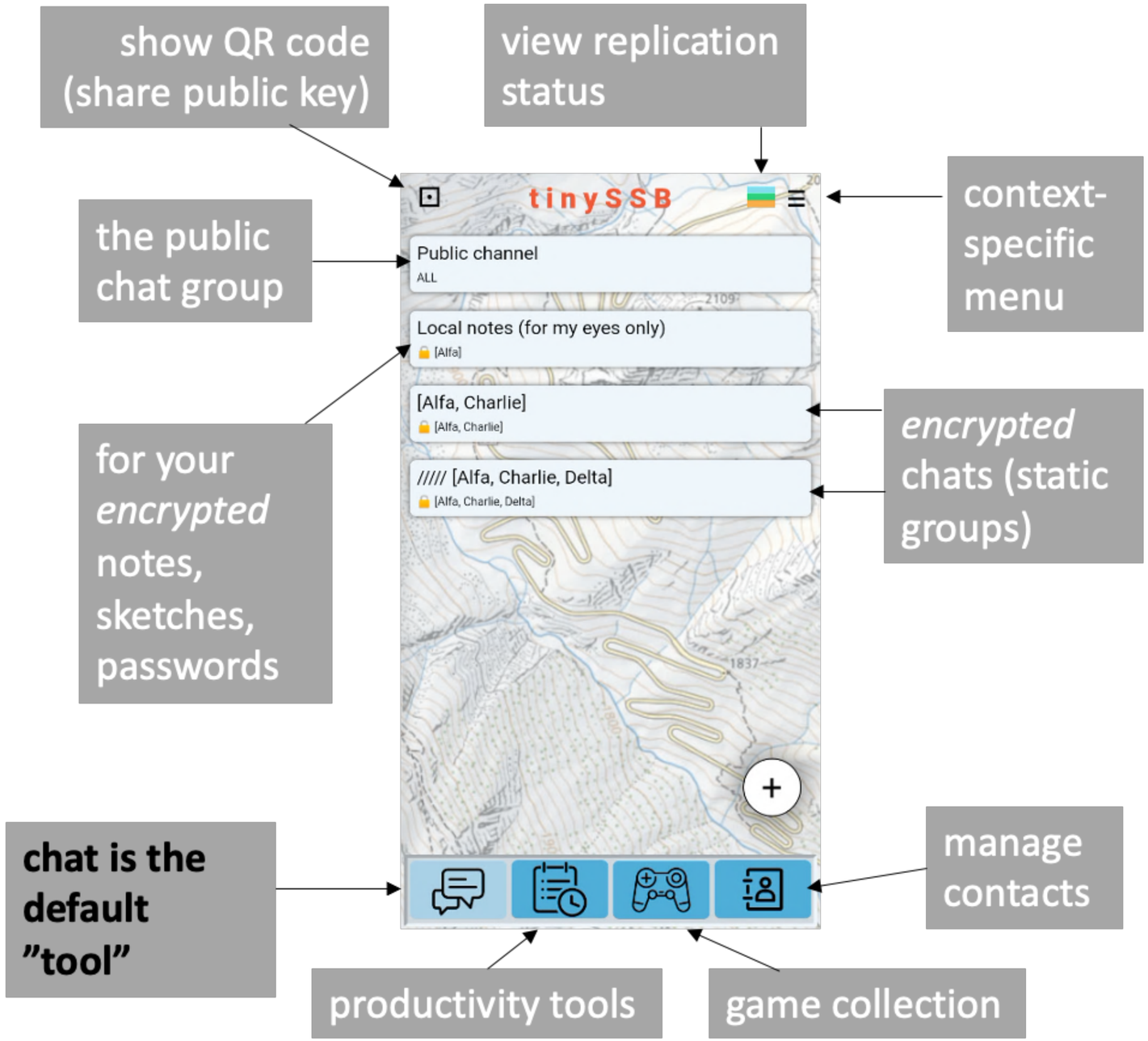
- convergent data structures (CRDT)
- beyond datagrams: “liquid data” mindset - enable novel information to flow anywhere

Try it out

join us at the

—> DEMO

of the tinySSB Android app



URLs of select Decentral Projects

SSB.classic - Web site and client software	https://scuttlebutt.nz/ https://github.com/ssbc/patchwork/releases/tag/v3.18.1
tinySSB (code and docs)	https://github.com/ssbc/tinySSB
P2Panda (append-only-log CRDT)	https://p2panda.org/
Willow (“last-writer-wins” CRDT)	https://willowprotocol.org/
BlueSky (“Merkle Tree CRDT”) = *decentral Twitter*	https://bsky.app/ https://docs.bsky.app/

Thank you for your attention

Questions?

Download the tinySSB Android app at

`https://github.com/ssbc/tinySSB`

(release 0.2.2-20240816)