GhostWire Architecture Deep Dive

Contents

Architecture Deep Dive

Overview

GhostWire is built for modularity, security, and real-world flexibility. Here's how the system fits together, for both non-technical and technical readers.

System Diagram

```
graph TD;
  User["User"] -->|Web UI| WebFrontend["React/Tailwind Web UI"]
  User -->|CLI| CLI["Rust CLI"]
  WebFrontend -->|REST/WebSocket| Backend["Rust Backend"]
  CLI --> Backend
  Backend -->|Transports| Transports["Bluetooth, WiFi, LoRa, WebRTC, TCP/IP"]
  Backend -->|Adapters| Adapters["Briar, Meshtastic, Matrix"]
  Backend --> Security["Security Modules"]
  Backend --> Store["Store & Forward"]
  Security -->|Sybil Defense, Quotas, Blacklists, Reputation, Disaster Triggers| Backend
```

Plain-Language Walkthrough

- **Web UI:** Like a modern chat app—easy for anyone to use.
- **CLI:** For power users and sysadmins—manage nodes, send messages, run diagnostics.
- **Backend:** The "brain"—routes messages, manages security, and connects everything.
- **Transports:** The "roads"—Bluetooth, WiFi, LoRa, WebRTC, TCP/IP.
- **Adapters:** The "translators"—let GhostWire talk to other networks (Briar, Meshtastic, Matrix).
- **Security Modules:** The "guards"—encryption, trust, quotas, blacklists, and more.
- **Store & Forward:** The "mailroom"—holds messages until they can be delivered.

Technical Details

- **Rust Backend:** Async, modular, exposes REST/WebSocket APIs.
- **Transports:** Each is a separate module/crate, implementing the `Transport` trait.
- **Adapters:** Each implements a common interface for protocol bridging.
- **Security:** End-to-end encryption (AES-256-GCM, X25519), Sybil defense, quotas, blacklists, reputation, disaster triggers, federation, traffic obfuscation.
- **Store & Forward:** Offline message delivery, caching, relay.
- **Web UI:** React/Tailwind, real-time messaging, network visualization, settings.
- **CLI:** Full-featured, for advanced management.

Real-World Deployment Examples

- **Urban Mesh:** Activists use GhostWire on phones and laptops, connecting via Bluetooth and WiFi to form a city-wide mesh.
- **Rural Mesh:** Farmers deploy LoRa nodes on barns and tractors, relaying messages over miles.
- **Disaster Response:** First responders set up portable GhostWire nodes with LoRa and WiFi to restore communication after a hurricane.
- **Censorship Resistance:** Journalists use Stealth TCP and WebRTC to bypass internet blocks.

Design Philosophy

- **Security First:** All code is reviewed for vulnerabilities and privacy risks.
- **Modularity:** New features are added as independent modules/crates.
- **Interoperability:** Prioritize compatibility with other mesh chat protocols.
- **Documentation:** All features and APIs are documented and kept up to date.
- **Community:** Encourage contributions, feedback, and open discussion.

GhostWire: Engineered for the real world.