Deep-Dive and Deployment of Kryfo: Embedded Tor Onion, Post-Quantum Cryptography, UnifiedPush, and Anti-Forensics

Deep-Dive and Deployment of Kryfo: Embedded Tor Onion, Post-Quantum Cryptography, UnifiedPush, and Anti-Forensics

In the modern landscape of secure communications, the compromise between user convenience and privacy has pushed most mainstream platforms into architectural concessions that clash with core cypherpunk principles. Highly regarded platforms such as **Signal** tie user identities to telephone numbers and rely on centralized cloud backends (AWS) utilizing proprietary SGX hardware enclaves. **Session**, while ditching phone numbers and central servers via the decentralized Oxen network, introduces Proof-of-Stake (PoS) consensus overhead and distributes onion routing metadata across public service nodes. **SimpleX Chat** has introduced a revolutionary paradigm with isolated unidirectional message queues and zero global IDs, yet it still requires trust in external relay infrastructure exposed on the open Internet.

When the adversary is not merely a passive wiretapper, but an entity conducting **traffic correlation analysis**, **physical forensic device extraction under seizure or coercion**, and **preventive bulk data harvesting targeted for cryptographically relevant quantum computers** (*Harvest Now, Decrypt Later*), threat modeling must become uncompromising.

This is precisely where **Kryfo** (*GnosiWorks*, created by Marios, derived from the Greek *Κρυφό*, meaning "hidden" or "secret") positions itself. Kryfo is neither a cosmetic wrapper nor a lightweight fork: it is an autonomous cross-platform messaging architecture (Android, Linux, macOS, Windows) engineered around four foundational pillars:

1. **In-Process Tor v3 Onion Topology**: No external system daemon or third-party background app required. Each client embeds a native Tor runtime (`libhalo.so`), turning every node into an ephemeral or persistent `.onion` hidden service over 6 cryptographic hops.
2. **Hybrid Post-Quantum Double Ratchet (PQXDH + SPQR)**: Blends classical elliptic-curve Diffie-Hellman (X25519) with lattice-based post-quantum key encapsulation mechanisms (Kyber / ML-KEM), paired with a *Symmetric*

Post-quantum Ratchet (SPQR) for uninterrupted Forward Secrecy and Break-in Recovery.
3. **Kernel-Level Anti-Forensics (`PR_SET_DUMPABLE, 0`)**: Hardens Linux and Android memory spaces against debugger attachment, `/proc/$PID/mem` extraction, and core dump generation, supplemented by encrypted database partitioning (SQLCipher), a decoy unlock PIN (*Decoy PIN*), an emergency instant-wipe PIN (*Wipe PIN*), and cosmetic disguise (*Icon Cloaking*).
4. **Anti-Traffic Analysis Push Architecture via UnifiedPush**: 16 deterministic slots with blind fixed-size payloads, 30-day dormant slot rotation (*resting slots*), and mathematical suppression of push notifications for hidden vault chats.

In this article, we dissect Kryfo's internal mechanics, document low-level Linux debugging resolutions (SQLCipher symbol shadowing and D-Bus portal interaction), and provide a production deployment guide for a dedicated **Private Obfs4 Tor Entry Guard** to defeat Deep Packet Inspection (DPI) and bridge censorship.

Scenario Parameters and Prerequisites

All commands, configuration snippets, and network listings in this guide adhere to IETF standards for technical documentation (RFC 2606 for hostnames and RFC 5737 for IP blocks).

Anyone replicating this deployment **must replace** these example values with their actual infrastructure parameters:

Scenario Parameter	Placeholder Example	Description & Replacement Instructions
UnifiedPush FQDN (ntfy)	`ntfy.example.org`	Public domain of your self-hosted ntfy server (with valid TLS certificate).
Host Bridge Public IP	`203.0.113.50`	Static public IPv4 address of your VPS hosting the obfs4 bridge container.
Obfs4 External Port	`9443`	External TCP port exposed by the Tor Bridge for Kryfo clients (obfs4 protocol).
Internal Tor ORPort	`9001`	Internal relay port for Tor on the host.
Tor Bridge Fingerprint	`A1B2C3D4E5F60718293A4B5C6D7E8F9012345678`	40-character hexadecimal fingerprint generated by the local Tor daemon.
Obfs4 Certificate (iat-mode)	`abcdef0123456789...`	Base64-encoded secret string extracted from `pt_state/obfs4_bridgeline.txt`.
UnifiedPush Slot Prefix	`up`	Prefix used for the 16 deterministic ntfy notification topics (`up0` .. `up15`).

Architecture and Communication Topology

Unlike traditional messaging clients that connect to centralized TLS load balancers, Kryfo functions as an asynchronous, distributed peer-to-peer network operating exclusively across Tor v3 rendezvous points:

```
flowchart LR
    subgraph ClientA ["Sender: Kryfo Mobile"]
        TorA["Embedded Tor Runtime\n(libhalo.so)"]
        VaultA["SQLCipher Encrypted DB\n(Visible Area + Vault)"]
        KDF_A["Linux/Android Kernel\nPR_SET_DUMPABLE=0"]
    end

    subgraph Transport ["Tor Anonymity Network"]
        Bridge["Private Obfs4 Bridge\n(:9443 TCP, Unlisted)"]
        RelayIn["Guard Node"]
        Middle["Middle Relays"]
```

```
        Rendezvous["Tor Rendezvous Point v3\n(6-Hop Bidirectional Circuit)"]
    end

    subgraph PushGateway ["Blind Push Gateway"]
        Ntfy["ntfy.example.org\n(ACL: up* read-write)"]
    end

    subgraph ClientB ["Recipient: Kryfo Desktop/Mobile"]
        TorB["Embedded Tor Runtime\n(libhalo.so)"]
        VaultB["SQLCipher DB"]
    end

    ClientA -->|Obfuscated obfs4 Flow| Bridge
    Bridge --> RelayIn
    RelayIn --> Middle
    Middle --> Rendezvous
    Rendezvous <--> ClientB

    ClientA -.->|Blind Dummy Push to slot upX| Ntfy
    Ntfy -.->|Wakeup Signal| ClientB
```

1. In-Process Tor Runtime (`libhalo.so`)

Kryfo eliminates external dependencies on background processes like `/usr/bin/tor` or Android VPN tunnels like Orbot. The Tor protocol engine is compiled into a shared C/Rust native library (`libhalo.so`) linked directly into the binary and controlled via Dart FFI:

- Each client independently generates and stores ephemeral or persistent `ed25519` v3 Onion Service keys within encrypted local storage.
- Neither endpoint leaks its IP address to the other: communication occurs over a full 6-hop Tor rendezvous circuit.

2. Hybrid Post-Quantum Ratcheting (PQXDH + SPQR)

Message encryption is not limited to Tor's onion transport layer. At the application layer, Kryfo applies an end-to-end post-quantum cryptographic scheme:

- **Hybrid Key Exchange**: Combines standard elliptic curve cryptography (X25519) with post-quantum lattice-based key encapsulation (**Kyber-1024 / ML-KEM**). Even if future quantum computers running Shor's algorithm dismantle elliptic-curve discrete logarithms, passively captured historical ciphertext remains secure.
- **Symmetric Post-quantum Ratchet (SPQR)**: For every message sent and received, symmetric keys ratchet forward through a one-way KDF chain. Compromising an ephemeral session key reveals neither previous

communications (*Forward Secrecy*) nor future messages once fresh entropy is absorbed (*Break-in Recovery*).

Technical Comparison Across Secure Protocols

The following matrix compares Kryfo against SimpleX Chat, Session, and Signal:

Architectural Metric	Kryfo	SimpleX Chat	Session	Signal
Global User Identifier	None (Ephemeral v3 Onion)	None (Unidirectional Queues)	Session ID (Ed25519 Key)	Phone Number (E.164)
Default Transport Layer	Native Tor v3 Hidden Services	TLS / WebSocket over Relays	Oxen Service Node Network	Centralized Cloud (AWS)
Post-Quantum Cryptography	PQXDH + SPQR (Kyber)	Double Ratchet PQ (Optional)	Not native	PQXDH (Kyber-1024)
Central Server Dependency	Zero	Minimal (Relay servers)	Minimal (Oxen blockchain)	Absolute (Signal Foundation)
Kernel Anti-Forensics	Native `PR_SET_DUMPABLE, 0`	Standard process space	Standard process space	Standard process space
Plausible Deniability Vault	Yes (Decoy PIN + Wipe PIN)	Multiple isolated profiles	None	None
Push Notification Scheme	16 Rotating Slots (UnifiedPush)	Polling / UnifiedPush	Open Group Push / Polling	Google FCM / Apple APNs
Push Notification Metadata	Zero (Fixed-length blind beacons)	Low to Medium	Low	Medium (FCM/APNs identifiers)

Memory Defense and Forensic Engineering

Kryfo's security posture is tailored to resist physical device extraction by forensic workstations (Cellebrite, GrayKey) and coercion attacks.

1. Hardening via `PR_SET_DUMPABLE, 0`

Upon process startup on Linux and Android, Kryfo executes:

```
#include <sys/prctl.h>
prctl(PR_SET_DUMPABLE, 0);
```

This syscall enforces strict kernel-level constraints:

- **Ptrace Immunity**: Disallows debugging attachments via `ptrace()` from any process lacking `CAP_SYS_PTRACE`, even from unprivileged processes sharing the same UID.
- **ProcFS Memory Access Blocked**: Prevents inspection of `/proc/$PID/mem` and memory-mapping tables in `/proc/$PID/maps`.
- **Core Dump Suppression**: If an unhandled exception or critical signal (`SIGSEGV`, `SIGABRT`) occurs, the kernel refuses to dump process RAM to disk, guaranteeing that ephemeral cryptographic keys and decrypted buffers never reach persistent storage.

2. Linux XDG Desktop Portal & SQLCipher Symbol Collisions

During our testing on Linux x86_64 desktop environments, two critical architectural hurdles were identified and resolved:

Issue A: D-Bus Portal AccessDenied

Invoking `PR_SET_DUMPABLE, 0` causes `xdg-desktop-portal` over D-Bus to fail credentials verification when querying `/proc/$PID/`. Consequently, system dialogs (file pickers, desktop notifications) yielded:

```
org.freedesktop.portal.Error.Failed: AccessDenied
```

The client codebase mitigates this through isolated helper proxies and safe fallbacks that preserve main-thread anti-debugging enforcement.

Issue B: Dynamic Linker SQLCipher Symbol Shadowing (`-Wl,-Bsymbolic`)

On modern Linux distributions, desktop frameworks frequently load the system's unencrypted `/usr/lib/libsqlite3.so.0`. When Kryfo dynamically loaded its statically linked **SQLCipher** engine, the dynamic linker (`ld-linux`) resolved database symbols against standard system SQLite functions instead of the encrypted SQLCipher implementations. This resulted in initialization failures (`file is not a database`) or database corruption.

We resolved this by injecting the linker flag into the native build configuration:

```
LDFLAGS += -Wl,-Bsymbolic
```

This forces the linker to prioritize internal symbols defined within the shared SQLCipher library before searching the global process namespace. The patch was submitted to and validated by upstream maintainer Marios (PR #5).

3. Vault Threat Model: Decoy PIN and Wipe PIN

Kryfo includes built-in protection against the *rubber-hose cryptanalysis* scenario:

- **Primary PIN**: Unlocks the full database and reveals hidden contacts.
- **Decoy PIN**: If coerced into unlocking the device, entering the Decoy PIN loads a clean, believable, yet entirely fictional secondary database. No cryptographic markers or UI elements betray the existence of the primary vault.
- **Wipe PIN**: Entering the emergency Wipe PIN immediately overwrites cryptographic master keys in secure hardware storage, purges the SQLCipher salt and page headers with random entropy, and triggers a clean crash. Recovering data from flash storage becomes mathematically impossible.

Push Architecture and Anti-Traffic Analysis

Maintaining persistent Tor connections on mobile devices leads to severe battery drain and aggressive background process termination by mobile operating systems (e.g., Android Doze). Conversely, relying on centralized commercial push gateways introduces severe **traffic correlation vulnerabilities**.

Kryfo solves this via a privacy-preserving implementation of **UnifiedPush**:

```
sequenceDiagram
    autonumber
    participant A as Alice (Kryfo Mobile)
    participant UP as ntfy Server (up0..up15)
    participant B as Bob (Kryfo Mobile)
    participant Tor as Tor Rendezvous Circuit

    Note over B,UP: Bob registers 16 deterministic slots on ntfy.<br/>Active slot: up3
| Dormant slots: up0..up2, up4..up15
```

```
    A->>UP: POST https://ntfy.example.org/up3 (Fixed-size blind beacon)
    UP-->>B: Anonymous Wakeup Push (No sender or content metadata)
    Note over B: Client wakes up in background
    B->>Tor: Connects to Alice's v3 Rendezvous Point
    Tor-->>B: Downloads encrypted PQXDH payload
    Note over B,UP: After 30 days (restsNow):<br/>Active slot rotates -> up7
```

Deterministic Slots and Dormant Rotation (`restsNow`)

Rather than generating a distinct push topic for each contact (which would allow push server operators to construct an accurate communication graph):

1. Kryfo allocates exactly **16 deterministic slots** numbered `0` to `15` (e.g., `up0` through `up15`).
2. **Fixed-Size Blind Payloads**: Messages dispatched through the push relay contain strictly uniform, opaque tokens. The ntfy server learns neither the identity of the sender, the recipient, nor the contents.
3. **30-Day Dormant Cycle**: Active slots rotate into dormant status (`restsNow`) every 30 days. This prevents long-term traffic profiling on the push server.
4. **No Push for Hidden Vault Chats**: As an inviolable architectural guarantee, **hidden vault chats never generate push traffic**. When a hidden contact transmits a message, zero packets hit the push gateway. Synchronization occurs solely when the recipient initiates asynchronous background polling over Tor or manually unlocks the hidden vault.

UnifiedPush ACL Configuration on ntfy

Because any peer on the Tor network must be able to publish a blind beacon to an `up*` slot, and the client must subscribe without long-lived identity credentials, the ntfy instance must permit anonymous read-write access to the slot namespace:

```
# Execute within the ntfy container
ntfy access everyone 'up*' read-write
```

Deploying a Private Obfs4 Tor Entry Guard

Connecting directly to publicly listed Tor Guard nodes makes your traffic identifiable by corporate DPI systems or Internet Service Providers. If an ISP flags IP connections to known Tor relays, it may throttle, block, or log the subscriber.

Deploying a **Private, Unlisted Tor Bridge** using the **obfs4** transport protocol scrambles packet headers and payload entropy, making traffic indistinguishable from random TLS/noise.

1. Docker Compose Configuration

Set up the service directory and create `docker-compose.yml`:

```yaml
services:
  kryfo-bridge:
    image: thetorproject/obfs4-bridge:latest
    container_name: kryfo-bridge
    restart: always
    environment:
      # Sostituire con il vostro indirizzo IP pubblico reale
      - OR_PORT=9001
      - PT_PORT=9443
      - EMAIL=admin@example.org
    ports:
      # Expose only the obfuscated obfs4 listener
      - "9443:9443"
    volumes:
      - /opt/kryfo-bridge/torrc:/etc/tor/torrc:ro
      - /opt/kryfo-bridge/data:/var/lib/tor
```

2. Crafting `torrc` with `PublishServerDescriptor 0`

Create `/opt/kryfo-bridge/torrc`. The single most critical directive is `PublishServerDescriptor 0`: **it strictly instructs Tor NOT to publish this node to central directory authorities**:

```
# Relay and Bridge Configuration
BridgeRelay 1
Nickname KryfoPrivateGuard
ContactInfo admin@example.org

# CRITICAL: Do not advertise this bridge to Tor Directory Authorities!
PublishServerDescriptor 0

# Listeners
ORPort 9001
ServerTransportPlugin obfs4 exec /usr/bin/obfs4proxy
ServerTransportListenAddr obfs4 0.0.0.0:9443
ExtORPort auto

# Strict Exit Policy: Never route outbound web traffic
ExitRelay 0
ExitPolicy reject *:*
```

```
# State Directory
DataDirectory /var/lib/tor
```

3. Service Initialization & Extraction of Bridge String

Launch the bridge service:

```
sudo mkdir -p /opt/kryfo-bridge/data
sudo chown -R 100:100 /opt/kryfo-bridge/data
cd /opt/kryfo-bridge && sudo docker compose up -d
```

Allow approximately 30 seconds for Tor to initialize cryptographic parameters, then extract the bridge descriptor:

```
sudo cat /opt/kryfo-bridge/data/pt_state/obfs4_bridgeline.txt
```

The resulting bridge string will resemble:

```
Bridge obfs4 203.0.113.50:9443 A1B2C3D4E5F60718293A4B5C6D7E8F9012345678
cert=abcdef0123456789ABCDEF... iat-mode=0
```

4. Terminal QR Code Generation

To import the bridge configuration into Kryfo Mobile without transmitting sensitive bridge strings over insecure channels:

```
sudo apt-get install -y qrencode
BRIDGE_LINE=$(sudo cat /opt/kryfo-bridge/data/pt_state/obfs4_bridgeline.txt | grep -E
'^Bridge')
qrencode -t ANSI256 "$BRIDGE_LINE"
```

Scan the terminal QR code in Kryfo (*Settings -> Network -> Custom Bridge*). All outbound client connections will immediately traverse your private unlisted guard node.

Multi-Device Mechanics and Cryptographic State

Pairing a secondary workstation (e.g., Kryfo Desktop on Linux) with a primary mobile client reveals intentional boundary constraints:

```
sequenceDiagram
    autonumber
    participant D as Kryfo Desktop (Secondary Node)
    participant M as Kryfo Mobile (Master Node)
    participant P as Remote Contact (Peer)

    Note over M,D: Local QR code pairing exchange.<br/>Secondary identity keys linked.
    Note over D: Desktop strictly excludes Hidden Vault<br/>(Physical Isolation Boundary).

    D->>P: Desktop initiates message dispatch
    Note over D: Message queued locally.<br/>kCapDevices: Capabilities pending acknowledgment.

    P-->>D: Informational UI notification:<br/>"Contact cannot receive messages from this computer yet.<br/>Reply from your phone."

    M->>P: Primary mobile client sends signed capability handshake
    Note over P: Peer validates primary signature<br/>and provisions ratchets for Desktop.
    P->>D: Peer transmits signed reply
    Note over D: Handshake complete: ratchet synchronized.<br/>Desktop can now transmit directly.
```

1. Deliberate Vault Exclusion on Desktop

Kryfo Desktop **never synchronizes nor displays contacts or messages stored within the mobile Hidden Area**. This isolation reflects distinct physical threat boundaries:

- A mobile phone represents a personal, physically secured item protected by biometrics and hardware secure enclaves.
- Workstations face cold-boot vectors, shared console sessions, OS-level crash loggers, and enterprise monitoring agents.
- Confining the hidden vault to mobile hardware prevents lateral exposure.

2. Asymmetric Capability Handshakes (`kCapDevices`)

When attempting to transmit a message from a newly linked desktop client to a peer, the UI presents the following notice:

> *"Peer cannot receive messages from this computer yet. Reply from your phone."*

This behavior is enforced by internal capability logic (`kCapDevices` / `_takeCaps`):

- Secondary devices generate their own distinct ratchet keypairs rather than cloning the master private key.
- To prevent rogue secondary injection by local workstation malware, the remote peer will not accept secondary device traffic until authorized by a signed handshake originated from the primary mobile client.
- Once the remote peer acknowledges and responds, the cryptographic ratchet unlocks bidirectional desktop communications.

3. Safety Numbers and Tamper Detection

If a device is wiped, reformatted, or wiped via Wipe PIN, regenerating new keys invalidates the ratchet state.

- Remote contacts receive immediate **Safety Number Change** warnings.
- This alert acts as a reliable tripswitch against impersonation or forensic cloning: communication partners are forced to perform out-of-band verification before resuming sensitive conversations.

Operational Verification and Diagnostics

Validate the deployment status using standard command-line tools:

```
# 1. Verify obfs4 port reachability from an external machine
nc -zvw3 203.0.113.50 9443
# Expected output: Connection to 203.0.113.50 9443 port [tcp] succeeded!

# 2. Inspect Tor bridge internal state and bandwidth counters
sudo docker exec -it kryfo-bridge cat /var/lib/tor/state | grep -i "Bandwidth"

# 3. Test UnifiedPush ACL configuration on ntfy
curl -s -o /dev/null -w "%{http_code}\n" -X POST https://ntfy.example.org/up0 -d
"ping"
# Expected output: 200 (Authorized) rather than 403 Forbidden

# 4. Confirm core dump suppression on Linux host
cat /proc/$(pgrep kryfo | head -n1)/status | grep -i "TracerPid"
# Expected output: TracerPid: 0
```

Conclusions and Practical Assessment

Although marked with an "Alpha" designation in its source repository, Kryfo exhibits an architectural maturity and depth of defense that surpass many enterprise messaging solutions.

By embedding an in-process Tor runtime, employing hybrid **PQXDH** with lattice-based **Kyber-1024**, enforcing kernel-level `PR_SET_DUMPABLE, 0` memory immunity, and designing an anti-traffic-analysis push scheme on **UnifiedPush**, Kryfo delivers uncompromising security for high-threat scenarios.

Coupled with a dedicated **Private Obfs4 Entry Guard**, this deployment represents a reference architecture for censorship-resistant, post-quantum private communication.



Discussion & Support

Discussion and support channel. To discuss configurations or other topics, join us on fluxer.dreadful.work (https://fluxer.dreadful.work/invite/dreadful).