# Analisi e deployment di Kryfo: messaggistica post-quantum su Tor onion embedded e anti-forensics

# Analisi e deployment di Kryfo: messaggistica post-quantum su Tor onion embedded e anti-forensics

Nel panorama odierno delle comunicazioni cifrate, il compromesso tra usabilità e riservatezza ha condotto la maggior parte delle piattaforme a concessioni architetturali difficilmente compatibili con i principi cypherpunk più rigorosi. Piattaforme eccellenti come **Signal** vincolano l'identità dell'utente al numero di telefono e poggiano su infrastrutture server centralizzate ospitate su cloud commerciali con enclave SGX proprietari. **Session**, pur eliminando numeri di telefono e server centrali attraverso la rete decentralizzata Oxen, introduce complessità legate a incentivi di blockchain (Proof-of-Stake) e metadati di instradamento distribuiti su nodi onion pubblici. **SimpleX Chat** ha rivoluzionato il paradigma introducendo code unidirezionali isolate senza identificatori utente globali, ma richiede la gestione o la fiducia in server di relay intermedi esposti su rete Internet aperta.

Quando la minaccia non è semplicemente l'intercettazione passiva del traffico sul filo, ma include **attacchi avversari di correlazione temporale**, **ispezione forense fisica del dispositivo sotto sequestro o coercizione** e **raccolta indiscriminata preventiva in vista di computer quantistici crittograficamente rilevanti** (*Harvest Now, Decrypt Later*), il modello di sicurezza deve radicalizzarsi.

---

## Che cos'è Kryfo: principi e filosofia

**Kryfo** (progetto sviluppato da Marios / *GnosiWorks*, dal greco *Κρυφό*, ovvero "nascosto" o "segreto") è un'applicazione di messaggistica istantanea autonoma e multipiattaforma progettata per operare senza alcuna infrastruttura centrale proprietaria.

A differenza delle app tradizionali, Kryfo si fonda su assunti radicali:

- **Nessun account, nessun dato personale**: non si utilizzano numeri di telefono, indirizzi email, nomi utente centralizzati o identificatori univoci globali.
- **Topologia puramente peer-to-peer su Tor Onion v3**: ogni client racchiude un demone Tor in-process. Non esistono server centrali che smistano o conservano i messaggi: due interlocutori comunicano direttamente attraverso un circuito di rendezvous a 6 salti crittografici.
- **Zero metadati in chiaro**: gli indirizzi IP non vengono mai rivelati né alla controparte, né al provider internet, né a nodi intermedi.
- **Protezione proattiva contro i computer quantistici**: i messaggi sono protetti da un doppio ratchet post-quantum basato su algoritmi a reticoli (*lattice-based cryptography*).
- **Progettazione anti-forensics estrema**: pensato per proteggere l'utente anche in caso di sequestro fisico del terminale, perquisizioni o coercizione forzata della password.

---

## Come si usa Kryfo: installazione, contatti e multi-dispositivo

Kryfo è sviluppato per funzionare in modo omogeneo su **Android**, **Linux** (desktop nativo), **macOS** e **Windows**. L'esperienza d'uso è volutamente lineare e priva di passaggi di registrazione formali.

### 1. Installazione e generazione dell'identità locale

Al primo avvio su smartphone o PC, Kryfo non richiede login o collegamenti esterni:

1. Genera localmente nel database cifrato le chiavi crittografiche dell'utente (chiavi di identità, coppie di chiavi effimere e l'indirizzo `.onion` v3 privato del servizio).
2. Chiede la configurazione di un **PIN di sicurezza** che servirà a derivare la chiave master del database locale (SQLCipher).
3. L'utente ha facoltà di impostare subito due PIN speciali facoltativi: un **Decoy PIN** (per sbloccare un'interfaccia esca priva

di chat compromettenti) e un **Wipe PIN** (per la distruzione istantanea dei dati in caso di emergenza).

### 2. Aggiunta dei contatti e circuito di rendezvous

Per avviare una conversazione con un altro utente:

1. Si condivide il proprio **QR Code** o la propria stringa di contatto (che include l'indirizzo onion v3 effimero e la chiave pubblica di identità).
2. L'interlocutore inquadra il codice QR o incolla la stringa nella schermata contatti.
3. I due runtime Tor interni avviano la negoziazione di un circuito onion v3 a 6 nodi (3 nodi scelti dal mittente, 3 dal destinatario) incontrandosi in un *Rendezvous Point* anonimo. Nessuno dei due nodi conosce la posizione geografica o l'indirizzo IP reale dell'altro.

### 3. Accoppiamento multi-dispositivo (smartphone e computer)

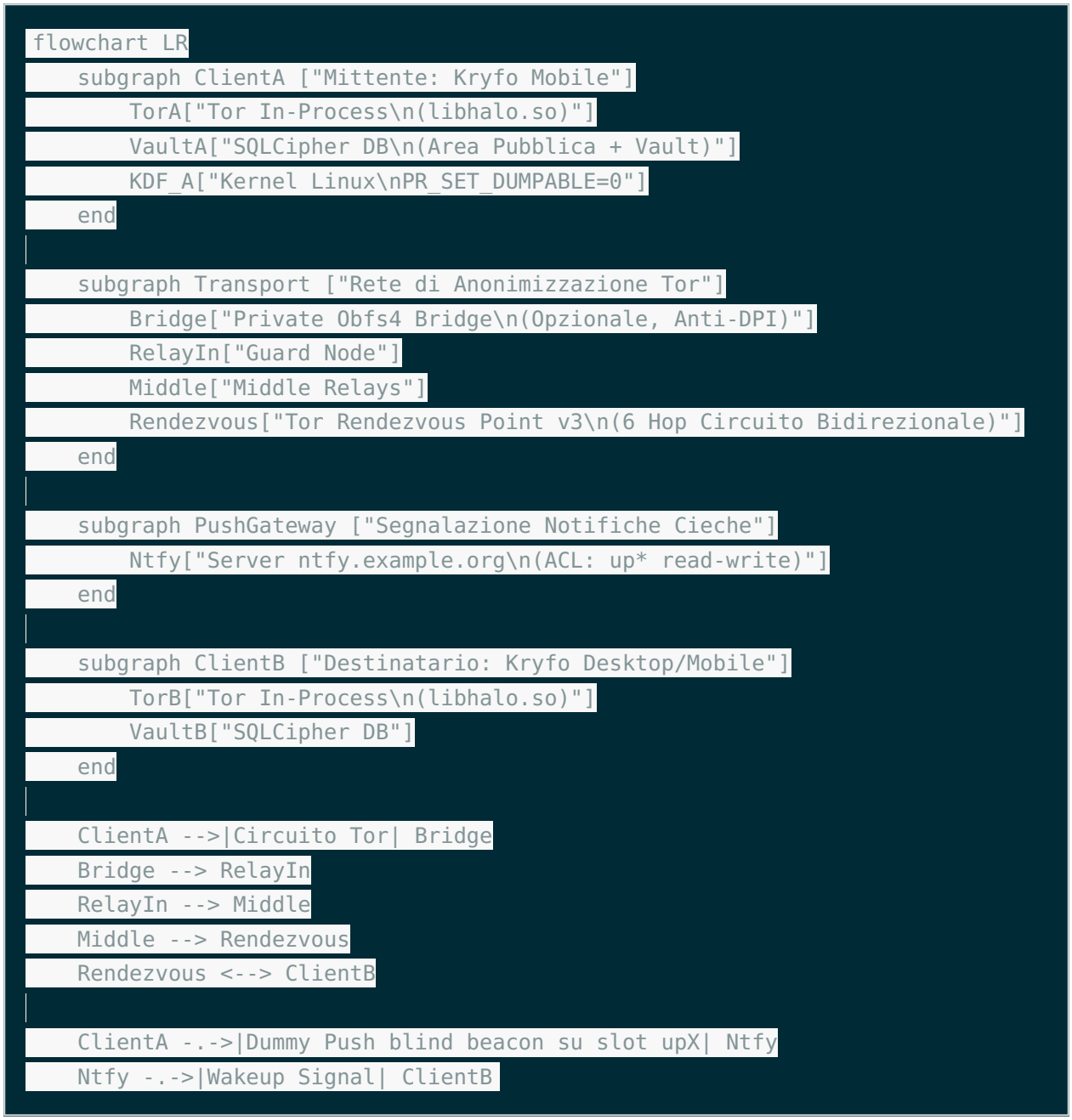
Kryfo consente di affiancare al terminale principale (smartphone) una o più postazioni secondarie (ad esempio un PC Linux o portatile):

- **Pairing via codice QR**: dalla versione desktop si seleziona *Accoppia dispositivo* e si inquadra il QR con l'app Kryfo dello smartphone primario.
- **Isolamento rigoroso del vault**: il PC non riceve né sincronizza in alcun modo i contatti o i messaggi archiviati nell'area nascosta (*Vault*). Il perimetro del vault rimane confinato esclusivamente all'hardware dello smartphone.
- **Handshake iniziale guidato dallo smartphone**: quando un nuovo client desktop accoppiato deve iniziare a chattare con un contatto, il protocollo impone che il primo frame di presentazione provenga dal dispositivo primario. Questo meccanismo previene che un'intrusione sul PC possa condurre a impersonificazioni arbitrarie all'insaputa del terminale mobile.

---

## Peculiarità architetturali e punti di forza

L'architettura di Kryfo si distingue per scelte ingegneristiche ad altissimo livello di sofisticazione:




## 1. Runtime Tor in-process (`libhalo.so`)

Kryfo non delega l'anonimizzazione al demone `/usr/bin/tor` del sistema operativo host, né impone su smartphone l'installazione di app terze come Orbot. Il core di Tor è compilato in una libreria condivisa C/Rust (`libhalo.so`) incorporata direttamente nell'applicazione e pilotata via FFI.

- Ogni istanza gestisce autonomamente il bootstrap e la negoziazione dei circuiti.
- Non vi sono porte proxy SOCKS locali aperte o accessibili da altre app presenti sul medesimo dispositivo.

## 2. Crittografia ibrida post-quantum (PQXDH e SPQR)

I messaggi non si affidano unicamente all'incapsulamento dei pacchetti della rete Tor: a livello applicativo end-to-end, Kryfo implementa una variante dello schema **PQXDH** (*Post-Quantum Extended Diffie-Hellman*):

- **Accoppiamento ibrido**: combina il protocollo Diffie-Hellman classico su curve ellittiche (X25519) con un algoritmo post-quantum a reticoli (**Kyber-1024 / ML-KEM**). Se in futuro un computer quantistico violasse le curve ellittiche con l'algoritmo di Shor, i dati archiviati passivamente oggi rimarrebbero impossibili da decifrare.
- **Symmetric Post-quantum Ratchet (SPQR)**: per ogni singolo messaggio, le chiavi simmetriche effimere avanzano tramite una catena crittografica a senso unico, garantendo sia *Forward Secrecy* che *Break-in Recovery*.

## 3. Notifiche push su UnifiedPush resistenti alla traffic analysis

Mantenere una connessione TCP/Tor permanentemente attiva su smartphone comporta un consumo energetico elevato. Kryfo risolve il problema integrando **UnifiedPush** (standard open source alternativo a Google FCM e Apple APNs):

- **16 slot fissi con payload ciechi**: il client alloca esattamente 16 slot numerati (`up0` .. `up15`) su un server di notifica `ntfy`. I pacchetti di risveglio contengono solo byte casuali di padding: il server ntfy non sa chi scrive, chi riceve o quale chat stia notificando.
- **Ciclo di riposo a 30 giorni**: gli slot attivi vengono ruotati periodicamente per spezzare qualsiasi correlazione statistica nel lungo periodo.
- **Nessuna notifica per le chat nascoste**: per preservare l'inviolabilità del vault, **le chat occultate non generano mai notifiche push**. I messaggi vengono scaricati unicamente quando l'utente sblocca attivamente lo spazio riservato o a intervalli asincroni dilazionati.

### 4. Hardening anti-forensics e negabilità plausibile

Kryfo integra contromisure avanzate per mitigare l'estrazione fisica forense (tramite box hardware tipo Cellebrite) e la perquisizione forzata:

- **Syscall `PR_SET_DUMPABLE, 0`**: blocca a monte la possibilità che altri processi o debugger (tramite `ptrace` o lettura da `/proc/$PID/mem`) leggano la memoria del processo. Inibisce inoltre la generazione di crash dump su disco che potrebbero contenere chiavi di decifratura in chiaro.
- **Partizionamento plausibile del database (SQLCipher)**:
  - **PIN reale**: sblocca la totalità delle conversazioni, incluse quelle del vault segreto.
  - **Decoy PIN**: se costretti a fornire il codice di sblocco, l'immissione del codice esca apre un'interfaccia perfettamente funzionante contenente solo chat ordinarie e innocue. Non c'è alcun indicatore matematico o crittografico che riveli l'esistenza dello spazio nascosto.
  - **Wipe PIN**: se inserito, Kryfo sovrascrive istantaneamente la chiave di decifratura con entropia casuale, azzera le tabelle SQLCipher e simula un crash. I dati sono irreversibilmente irrecuperabili.

---

## Il nostro modello di deployment e debugging su ambiente Linux

Durante i test di compilazione, porting e impiego su ambiente Linux x86_64, abbiamo documentato e risolto diverse problematiche critiche di basso livello.

## 1. Confronto con le piattaforme concorrenti

| Caratteristica architetturale | Kryfo | SimpleX Chat | Session | Signal |
|---|---|---|---|---|
| **Identificatore globale utente** | Nessuno (solo Onion v3) | Nessuno (code unidirezionali) | Session ID (chiave Ed25519) | Numero di telefono |
| **Trasporto di rete predefinito** | Tor v3 onion nativo | TLS / WebSocket su relay | Oxen Service Node Network | Server centralizzato AWS |
| **Crittografia post-quantum (PQ)** | PQXDH + SPQR (Kyber) | Double Ratchet PQ (opzionale) | Non presente nativamente | PQXDH (Kyber-1024) |
| **Dipendenza da server centrali** | Zero | Bassa (relay distribuiti) | Bassa (blockchain Oxen) | Assoluta (infrastruttura Signal) |
| **Anti-forensics hardware/kernel** | `PR_SET_DUMPABLE, 0` nativo | Nessuna protezione speciale | Nessuna protezione speciale | Nessuna protezione speciale |
| **Partizionamento plausibile (Vault)** | Sì (Decoy PIN + Wipe PIN) | Profili multipli | Nessuno | Nessuno |
| **Strategia notifiche push** | 16 slot rotativi (UnifiedPush) | Polling / UnifiedPush | Open Group Push / Polling | Google FCM / Apple APNs |

## 2. Risoluzione della collisione dei simboli SQLCipher

Su distribuzioni Linux moderne, alcune librerie grafiche di sistema possono richiamare la libreria condivisa `/usr/lib/libsqlite3.so.0`. Nel momento in cui Kryfo caricava la propria versione statica di **SQLCipher**, il dynamic linker di sistema (`ld-linux`) effettuava il binding dei simboli verso le funzioni SQLite standard non cifrate. Questo causava il fallimento dell'apertura del database con l'errore `file is not a database`.

Abbiamo affrontato il problema forzando nelle direttive di compilazione della libreria C il flag linker:

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

Questo parametro impone al linker di risolvere i riferimenti ai simboli prioritariamente all'interno della libreria condivisa di SQLCipher prima di cercare nello spazio globale dei simboli esportati dal processo.

### 3. Gestione dei portali desktop XDG e della syscall PR_SET_DUMPABLE

Quando un'applicazione attiva `PR_SET_DUMPABLE, 0`, il servizio `xdg-desktop-portal` su D-Bus non riesce a verificare le credenziali del chiamante analizzando `/proc/$PID/`. Le richieste di apertura di finestre native (come il selettore di file) possono quindi fallire con l'errore:

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

La soluzione richiede la corretta manipolazione dei descrittori di file dedicati o l'esecuzione dei dialoghi di selezione file tramite thread ausiliari senza declassare la protezione sul thread principale che custodisce i segreti crittografici.

---

## Addendum opzionale: configurazione di un private obfs4 Tor bridge

> ***Nota bene**: questa sezione **non è obbligatoria** per l'utilizzo di Kryfo. L'applicazione si connette nativamente e in piena trasparenza alla rete Tor senza richiedere server o nodi personali.*

## 1. Quando è utile un bridge dedicato

Configurare un proprio **Private Obfs4 Bridge** è indicato per scenari specifici:

- **Aggiramento della censura**: in Paesi o reti aziendali/universitarie in cui il traffico Tor standard o gli indirizzi dei guard node pubblici vengono bloccati tramite ispezione DPI.
- **Anonimato rafforzato**: un bridge privato e non pubblicato garantisce che nessun provider internet o osservatore esterno possa associare l'indirizzo IP del vostro telefono o computer alla rete Tor.

## 2. Configurazione Docker Compose e torrc privato

Creiamo la directory `/opt/kryfo-bridge` sul server con il seguente `docker-compose.yml`:

```yaml
services:
  kryfo-bridge:
    image: thetorproject/obfs4-bridge:latest
    container_name: kryfo-bridge
    restart: always
    environment:
      - OR_PORT=9001
      - PT_PORT=9443
      - EMAIL=admin@example.org
    ports:
      # Pubblicazione esclusiva della porta obfs4 offuscata
      - "9443:9443"
    volumes:
      - /opt/kryfo-bridge/torrc:/etc/tor/torrc:ro
      - /opt/kryfo-bridge/data:/var/lib/tor
```

Nel file `/opt/kryfo-bridge/torrc`, la direttiva essenziale è `PublishServerDescriptor 0`, che istruisce Tor a **non registrare il bridge nelle liste pubbliche mondiali**:

```
BridgeRelay 1
Nickname KryfoPrivateGuard
ContactInfo admin@example.org

# CRITICO: Impedisce la pubblicazione nella directory pubblica di Tor
```

```
PublishServerDescriptor 0

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

ExitRelay 0
ExitPolicy reject *:*
DataDirectory /var/lib/tor
```

## 3. Avvio, estrazione della bridge line e generazione del QR code

Avviamo il container e leggiamo la bridge line generata:

```
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

# Attendere circa 30 secondi per l'inizializzazione delle chiavi
sudo cat /opt/kryfo-bridge/data/pt_state/obfs4_bridgeline.txt
```

La riga estratta ha il formato:

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

Per caricarla istantaneamente su Kryfo Mobile, possiamo visualizzarla come codice QR nel terminale:

```
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"
```

Inquadrando il QR dalle impostazioni di rete dell'applicazione (*Impostazioni -> Rete -> Configurazione Bridge Personalizzato*), l'intero traffico del client verrà convogliato attraverso il vostro relay offuscato dedicato, apparendo all'ISP come normale flusso binario opaco privo di impronte Tor o certificati TLS catalogati.

---

## Conclusioni

Sebbene Kryfo sia formalmente etichettato come software in fase "Alpha", la solidità della sua architettura crittografica e la radicalità delle sue misure difensive dimostrano una coerenza progettuale rara.

L'integrazione di un runtime Tor in-process elimina la complessità di gestione per l'utente, mentre la combinazione tra **PQXDH**, **`PR_SET_DUMPABLE`**, partizionamento con **Decoy/Wipe PIN** e segnalazioni cieche su **UnifiedPush** colma lacune critiche lasciate aperte dai software di messaggistica convenzionali.

Per chi richiede comunicazioni insensibili sia all'intercettazione e analisi del traffico di rete che alla coercizione fisica sul dispositivo, Kryfo rappresenta oggi una delle soluzioni più avanzate e intransigenti realizzabili nel panorama open source.

Fluxer Icon

Canale di discussione e supporto

Canale di discussione e supporto. Per confrontarti su configurazioni o altro entra su fluxer.dreadful.work (https://fluxer.dreadful.work/invite/dreadful).