Unable to decrypt this message

Kegan Dougal

Unable to decrypt this messa

Credit:
https://mastodon.delroth.net/@delroth

How does it happen?

Data failed

Why does this happen?

Servers

Software reliability issues

Hardware reliability issues

Network reliability issues

Server diversity

Server	Synapse	Dendrite	Conduit	Construct	Conduwuit
Reverse Proxy	HAProxy	nginx	traefik	Caddy	Apache
Extensions	Sliding Sync				

Why does this happen?

Clients

Software reliability issues

Robustness issues

Client diversity

Client	FluffyChat	Nheko	Cinny	Element Web/ Desktop	Element X	Element Android/ iOS
SDK	Dart	Nheko	JS		Rust	Kotlin/Swift
Crypto Library	libolm			vodozemac		

Why does this happen?

Protocol

Unbounded group sizes

Federation

Backwards compatibility

Ecosystem inertia: Even if your client/server is perfect, UTDs can be caused by the ***sender*** *or receiver*, so the ecosystem matters.

How is this being fixed?

Identifying bugs: the canonical list of UTD bugs is at https://github.com/element-hq/element-meta/issues/245

Reducing complexity: reducing the diversity of the ecosystem for crypto - converging on Vodozemac and matrix-rust-sdk

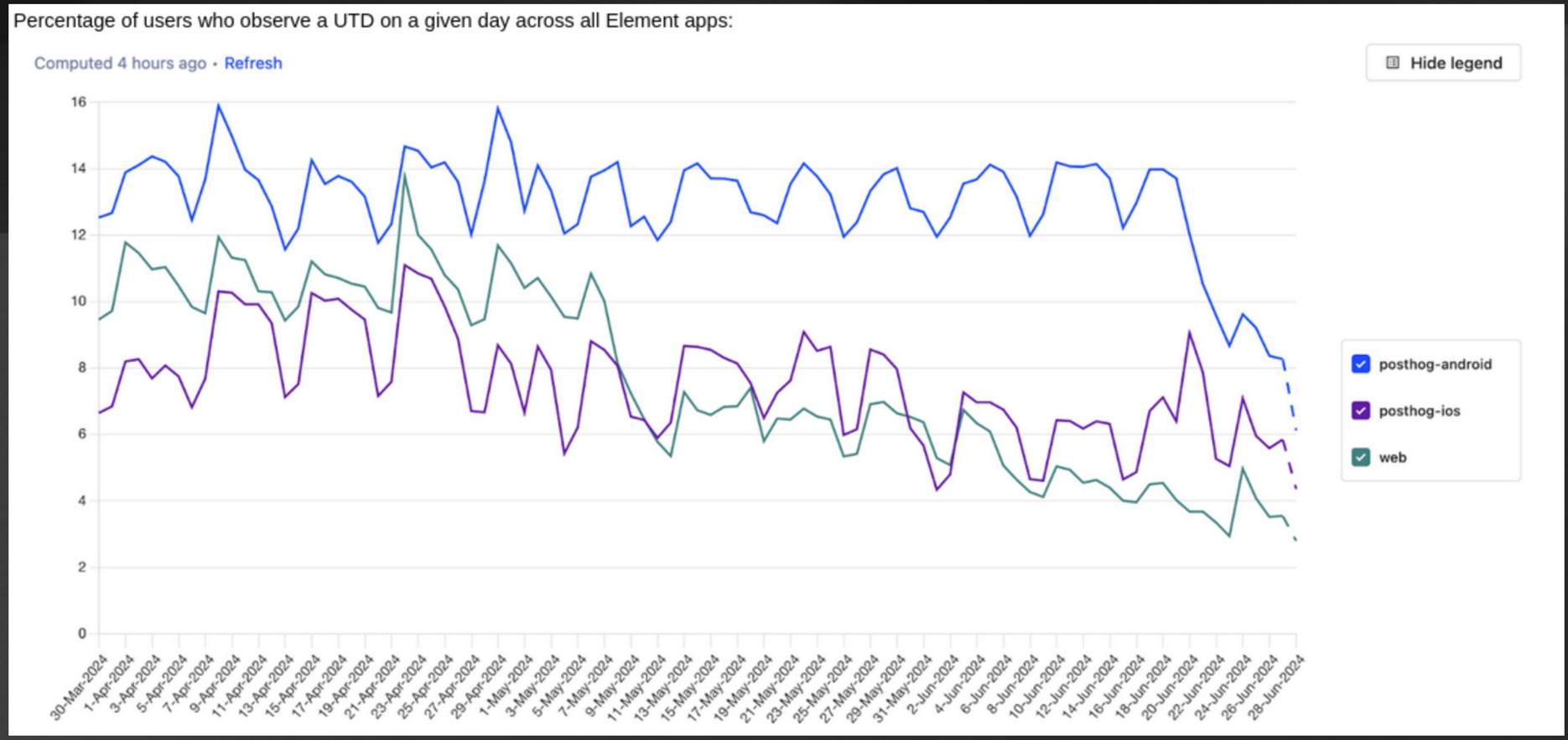
Protocol changes: mostly around increasing network robustness

Adding end-to-end regression tests: complement-crypto

Tracking progress: opt-in analytics track UTDs in the wild.

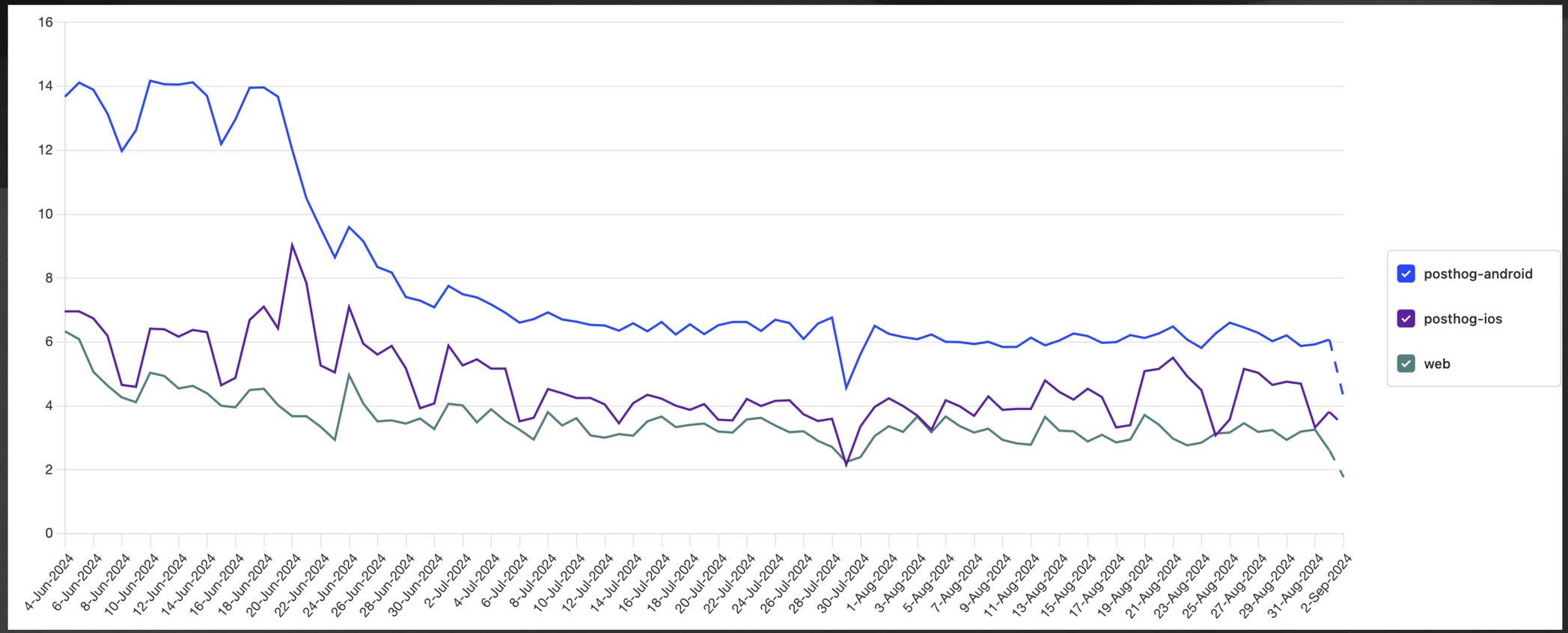
Is it working?

YES! The below chart is the % of users seeing at least 1 UTD



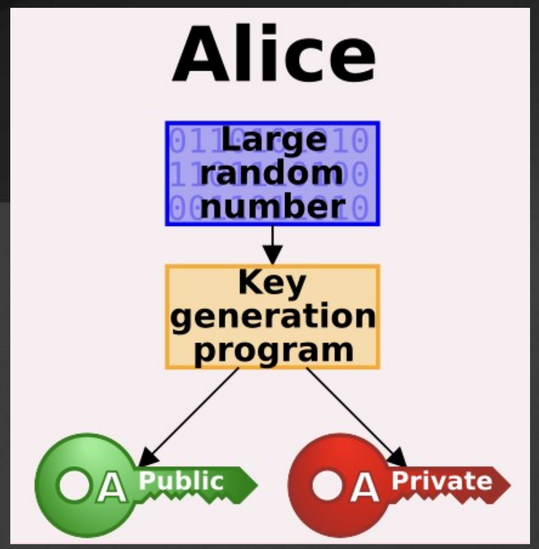
Is it working?

YES! The below chart is the % of users seeing at least 1 UTD



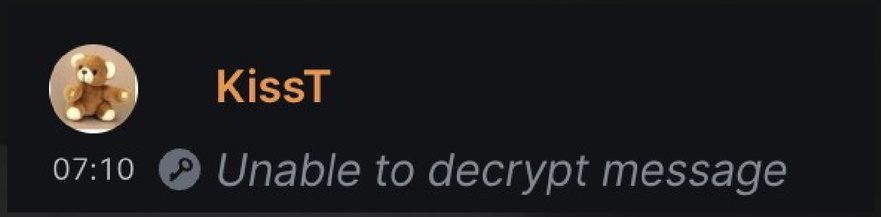
The anatomy of a UTD

Public Key Cryptography



Credit: https://en.wikipedia.org/wiki/Public-key_cryptography

What even is a UTD?



If the message was sent **before you logged in**, the key may be in your backup if you had another device logged in when this message was sent.

If the message was sent **after you logged in**:

The message may decrypt if you wait a while,

Or it may remain permanently undecryptable. This is a UTD.

Crypto Stack

..for events sent <u>after</u> you login

We’re going to use a simplified model to enable us to explore failure
rated devices, cross-signing, secret s
upon Nico’s excellent blog post https://blog.nek

Crypto Stack

..for events sent <u>after</u> you login

Failures to send/receive to-device messages

Failures to upload

This talk expand u

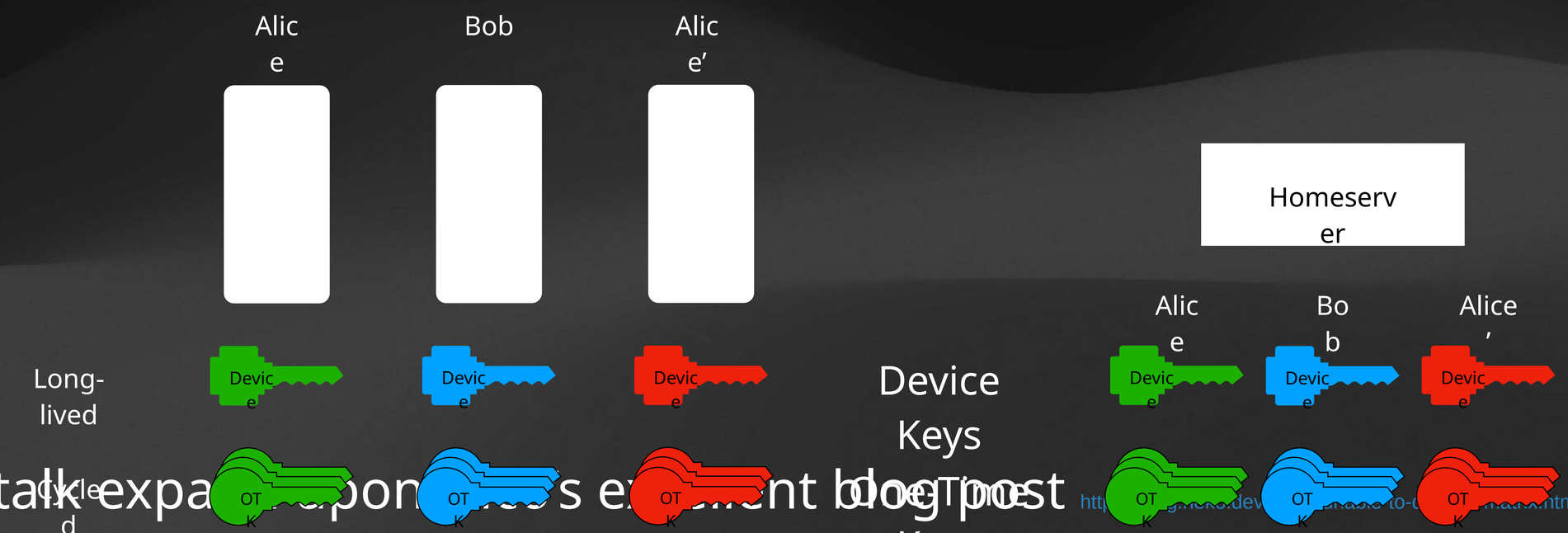
Olm
Establishment

Megol
m

2024-09-02 14:36:46.1...	POST	http://ssproxy1:6789/_matrix/client/unstable/org.matrix.msc3575/sync?pos=1&timeout=...	200
2024-09-02 14:36:46.105	POST	http://hs1:8008/_matrix/client/v3/keys/upload	200
2024-09-02 14:36:46.127	POST	http://ssproxy1:6789/_matrix/client/unstable/org.matrix.msc3575/sync?pos=1&timeout=...	200
2024-09-02 14:36:46.132	POST	http://hs1:8008/_matrix/client/v3/keys/query	200
2024-09-02 14:36:46.1...	POST	http://ssproxy1:6789/_matrix/client/unstable/org.matrix.msc3575/sync?pos=2&timeo...	!
2024-09-02 14:36:46.1...	GET	http://hs1:8008/_matrix/client/v3/rooms/!FydVlSFxTLAzNzakkQ:hs1/members	200
2024-09-02 14:36:46.1...	POST	http://hs1:8008/_matrix/client/v3/keys/claim	200
2024-09-02 14:36:46.2...	PUT	http://hs1:8008/_matrix/client/v3/sendToDevice/m.room.encrypted/9de4ec0a8781459c...	200
2024-09-02 14:36:46.210	PUT	http://hs1:8008/_matrix/client/v3/rooms/!FydVlSFxTLAzNzakkQ:hs1/send/m.room.encryp...	200

Olm Data Model

Oversimplified!



This talk expands upon Matrix.org's excellent blog post

For more information on how Megolm works, see Sumner’s blog post

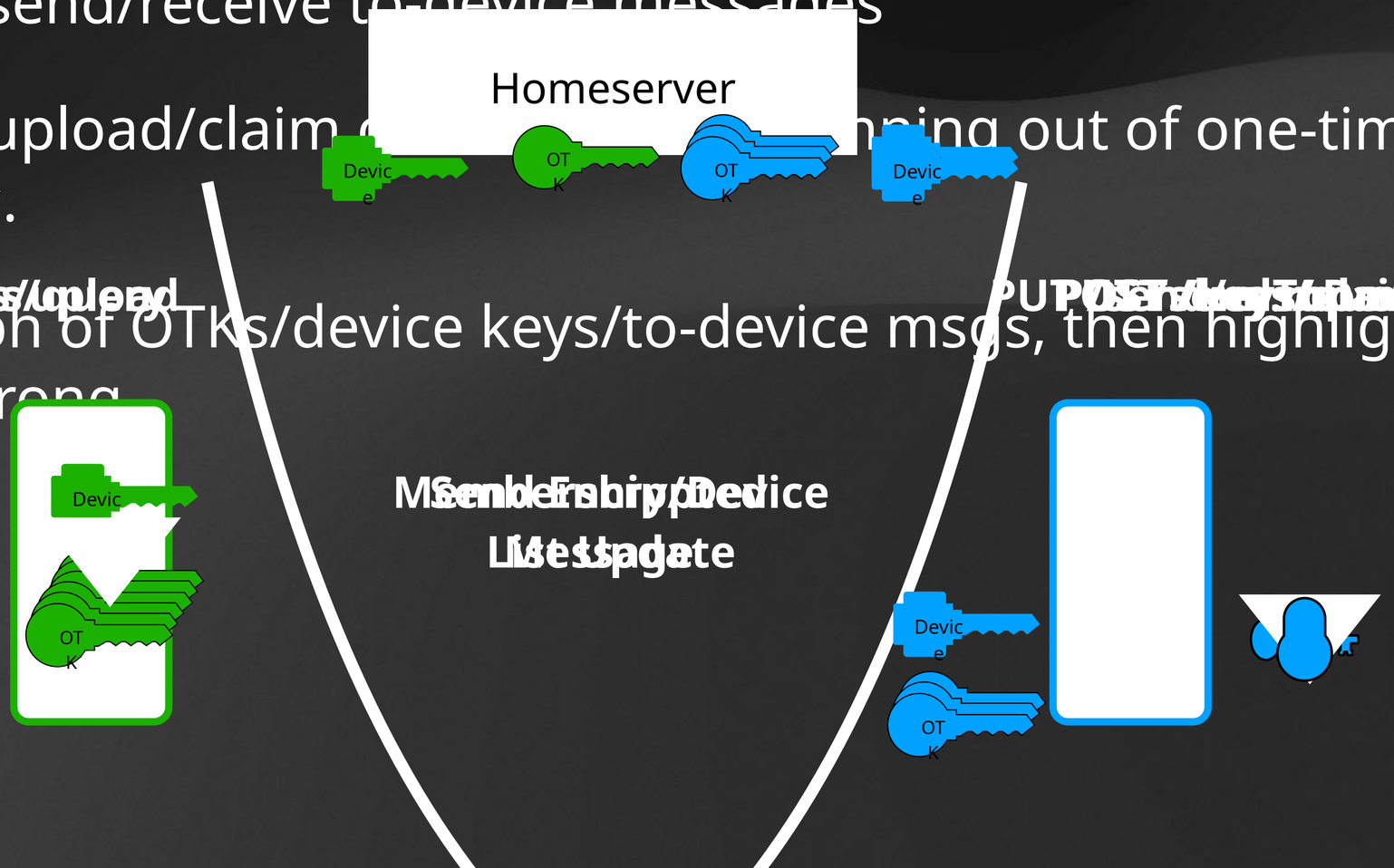
https://sumnerevans.com/posts/matrix/megolm/

Happy Case

Failures to send/receive to-device messages

Failures to upload/claim [illegible]ning out of one-time keys. OTK desync.

TODO: graph of OTKs/device keys/to-device msgs, then highlight sections which go wrong



What goes wrong

Failure to upload/claim One-Time Keys

Failures to send/receive to-device m

Failures to upload/claim one-time keys. Running out of one-time keys
OTK desync.

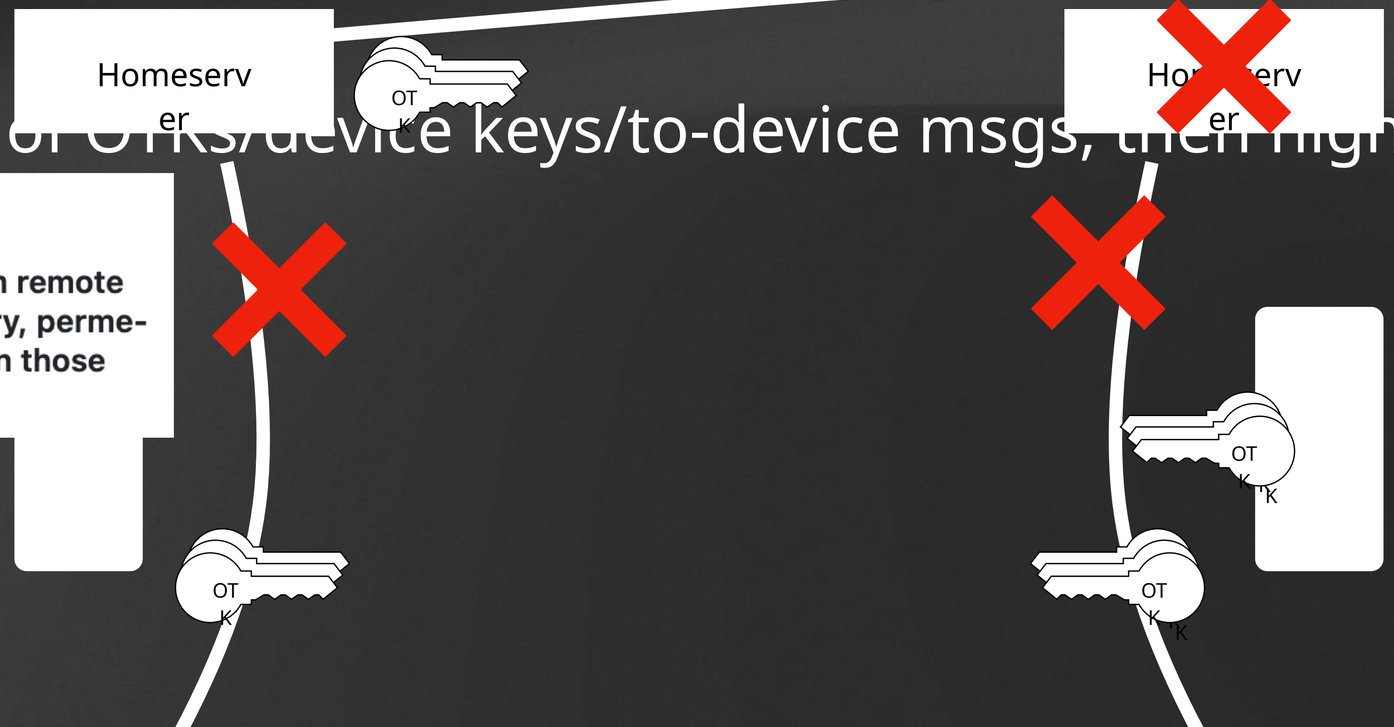
TODO: graph of OTKs/device keys/to-device msgs, then highlight sections

element-hq/element-meta on Oct 19, 2023

Users whose servers were unreachable will receive undecryptable messages due to failed OTK claim #2154

element-hq/element-web on Jan 2, 2023

We abandon claiming OTKs from remote servers after 10s, and never retry, permenantly breaking E2EE to users on those servers. #24138



What goes wrong

One-Time Keys get out-of-sync

matrix-org/matrix-spec on Jun 14, 2022

One-time-key upload/claim is racy #1124

device messages

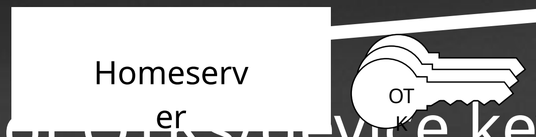
Failures to upload/claim one-time keys. Running ou

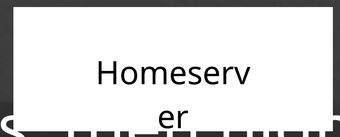
OTK desync.

TODO: graph of OTKs/device keys/to-device msgs, then highlight sections

element-hq/element-meta on Oct 19, 2023

Rolling a homeserver's database back via backup could cause duplicate OTKs and hence UISIs #2155





matrix-org/matrix-rust-sdk on Jan 27, 2023

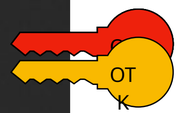
OneTime key already exists error #1415

element-hq/element-ios on Apr 6, 2023

Decryption failure caused one-time-key to be discarded #7480

element-hq/element-meta on Feb 23, 2017

We can throw away one-time keys which are still published on the server, or have mes-sages in flight #2356



What goes wrong

Membership or Device list gets out-of-sync

element-hq/synapse on Dec 19, 2023

Synapse doesn't send out device list updates to previously unseen homeservers when joining a room #11374

element-hq/synapse on Dec 16, 2023

Invited users don't trigger device_list updates when their device lists change. #3504

element-hq/synapse on Feb 19

`/sync` and `/members` do not return "current state" #16940

matrix-org/matrix-rust-sdk on Jun 28

Delayed membership responses in /sync cause UTDs #3622

element-hq/element-meta on Jan 12

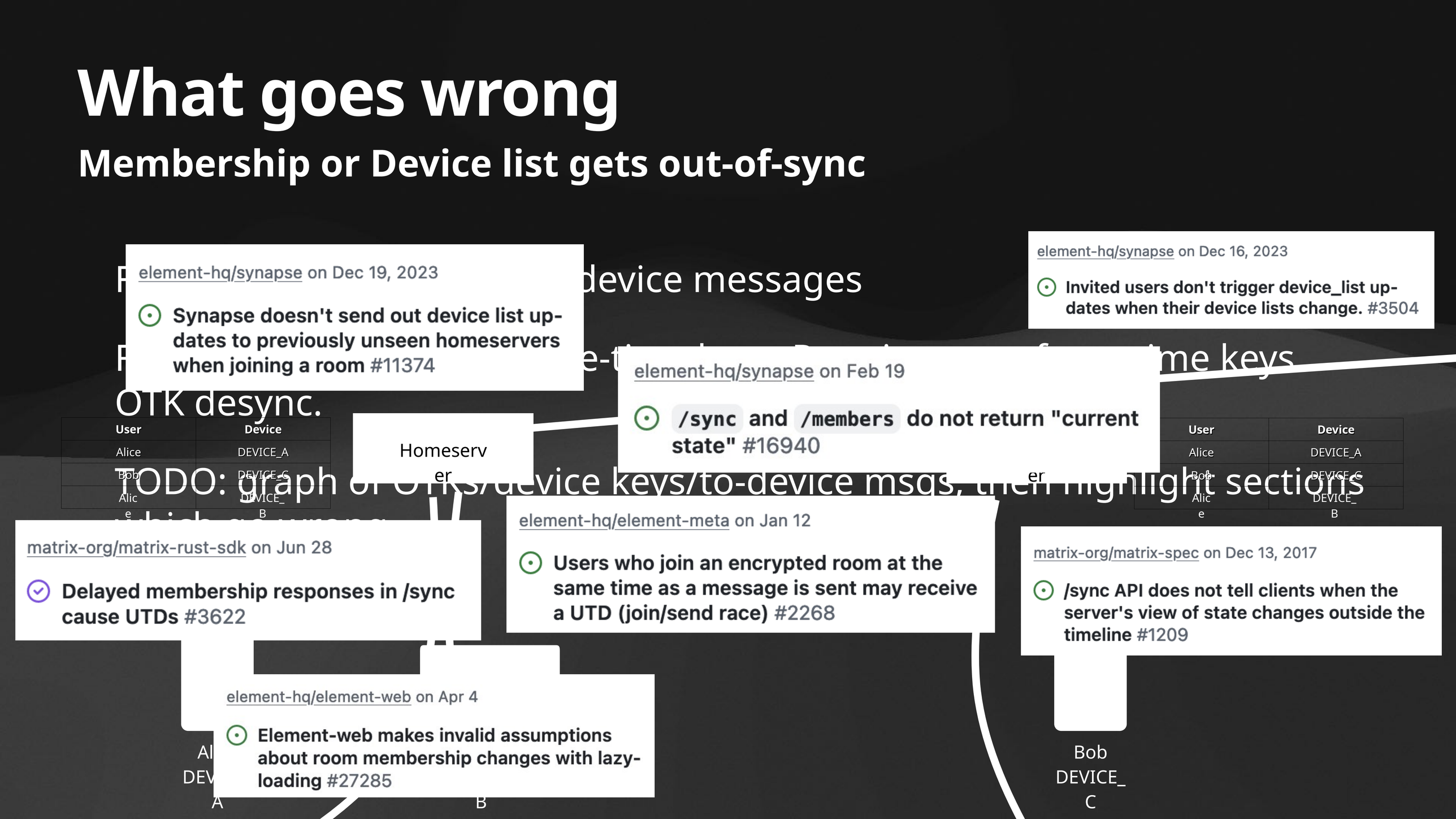
Users who join an encrypted room at the same time as a message is sent may receive a UTD (join/send race) #2268

matrix-org/matrix-spec on Dec 13, 2017

/sync API does not tell clients when the server's view of state changes outside the timeline #1209

element-hq/element-web on Apr 4

Element-web makes invalid assumptions about room membership changes with lazy-loading #27285



What goes wrong

Failure to send/receive room key (Megolm)

matrix-org/matrix-spec on Jun 14, 2022

- Room events arrive faster than to-device messages during federation connectivity problems, causing decryption failures #1123

element-hq/synapse on Dec 21, 2023

- Synapse does not attempt to send events in the device outbox at startup #16680

matrix-org/matrix-spec on Oct 5, 2023

- Important/urgent to-device messages (eg room keys) can get blocked behind big queues of less important ones #1659

Failures to se e messages

element-hq/synapse on Apr 24

- Synapse drops received federated to-device messages if it cannot talk to worker processes #17117

aim one-time keys. Running out of c

matrix-org/synapse on Nov 23, 2023

- Running under sqlite, Synapse incorrectly populates the to-device messages current stream ID #16681

OTK desync:

Homeserver

Homeserver

TODO: graph of OTKs/device keys/to-device msgs, then highlight sections which go wrong.

element-hq/element-web on Feb 27, 2023

- batching of `m.room_key` to-device messages is suboptimal #24680

element-hq/element-web on Jul 5, 2023

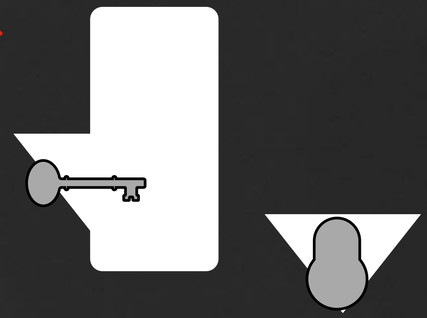
- to-device messages can be processed out-of-order, causing dropped keys and decryption errors #25723

element-hq/element-meta on Oct 3, 2022

- Queue and retry to-device msgs, so E2EE works in bad networks or with unreliable HS #673

element-hq/element-meta on Oct 14, 2022

- incoming encrypted to-device messages can be lost when the application is restarted #762



Client State Machine

All clients need to receive data from the network and reliably persist this to disk.

All clients need to atomically perform read-modify-write operations in response to user interaction or data from the network.

Client State Machine

Causality violations: invites

Invite a user to a room… then send a message.

If the client only updates its member list in response to /sync responses, it will fail to encrypt for the invited user.

Client State Machine

Non-atomic operations: sending keys and event, incomplete OTK claim

Send room keys to all participants in a room… but some time out.

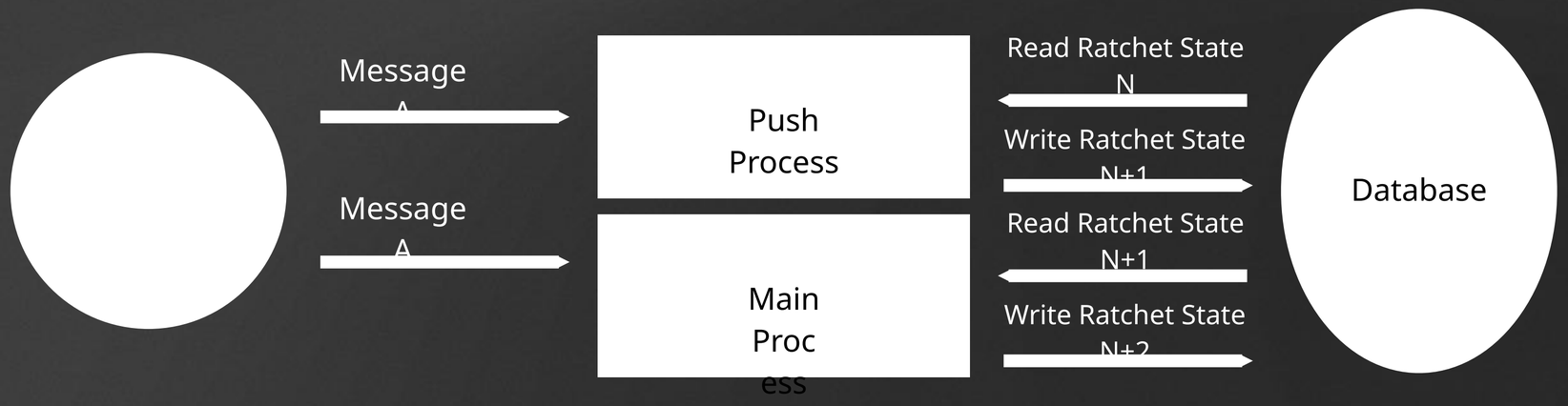
If the client still sends the event despite this, this will cause a UTD.

Client State Machine

Read-Modify-Write violations: split-brain processes, inadequate txn locking

Two processes receive the same data to persist… but they don’t co-ordinate correctly.

If this data is ratchet state, this will corrupt the ratchet causing UTDs on the next message send.



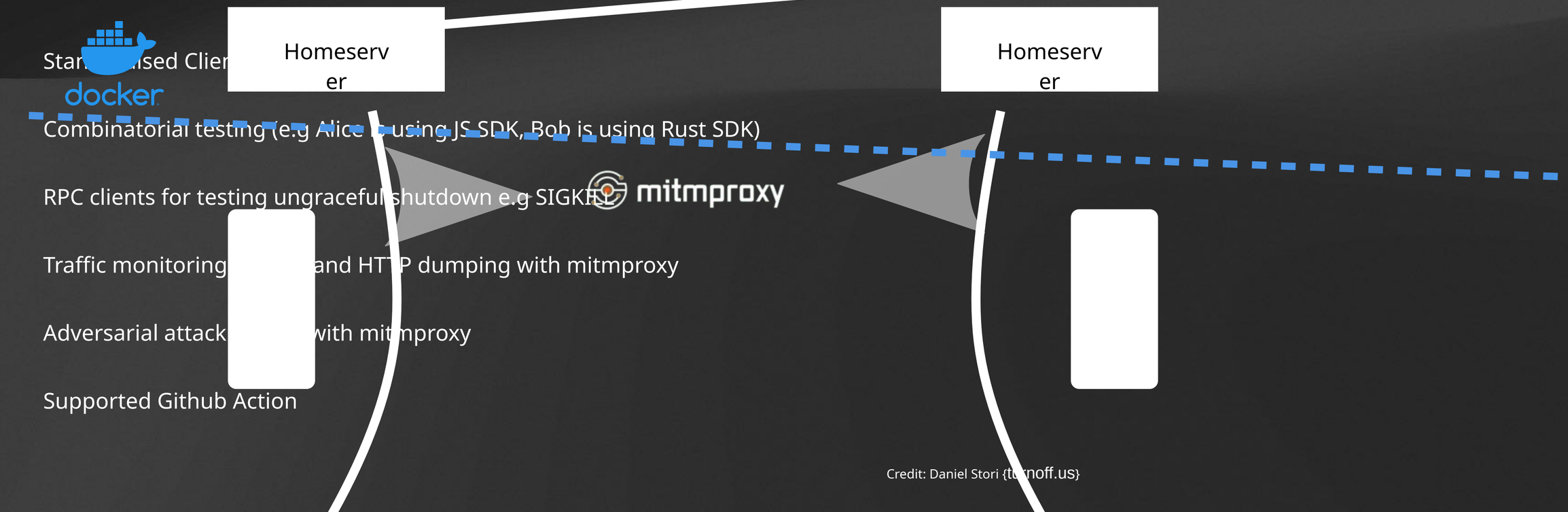
Complement Crypto

Complement Crypto

Architecture

Complement-Crypto is a black box client testing framework, built on top of Complement which is used for black box server testing. Whilst it was originally written for crypto reliability, it isn't restricted to just that. It has been successfully used to detect numerous UTD issues and some CVEs.

The key novel features it has are:



Start[illegible]ised Clien[illegible]

Combinatorial testing (e.g Alice is using JS SDK, Bob is using Rust SDK)

RPC clients for testing ungraceful shutdown e.g SIGKILL

Traffic monitoring [illegible]and HTTP dumping with mitmproxy

Adversarial attack[illegible]with mitmproxy

Supported Github Action

Credit: Daniel Stori {turnoff.us}

Complement Crypto

Feature Set

Standardised Client API for SDK testing

Combinatorial testing (e.g Alice is using JS SDK, Bob is using Rust SDK)

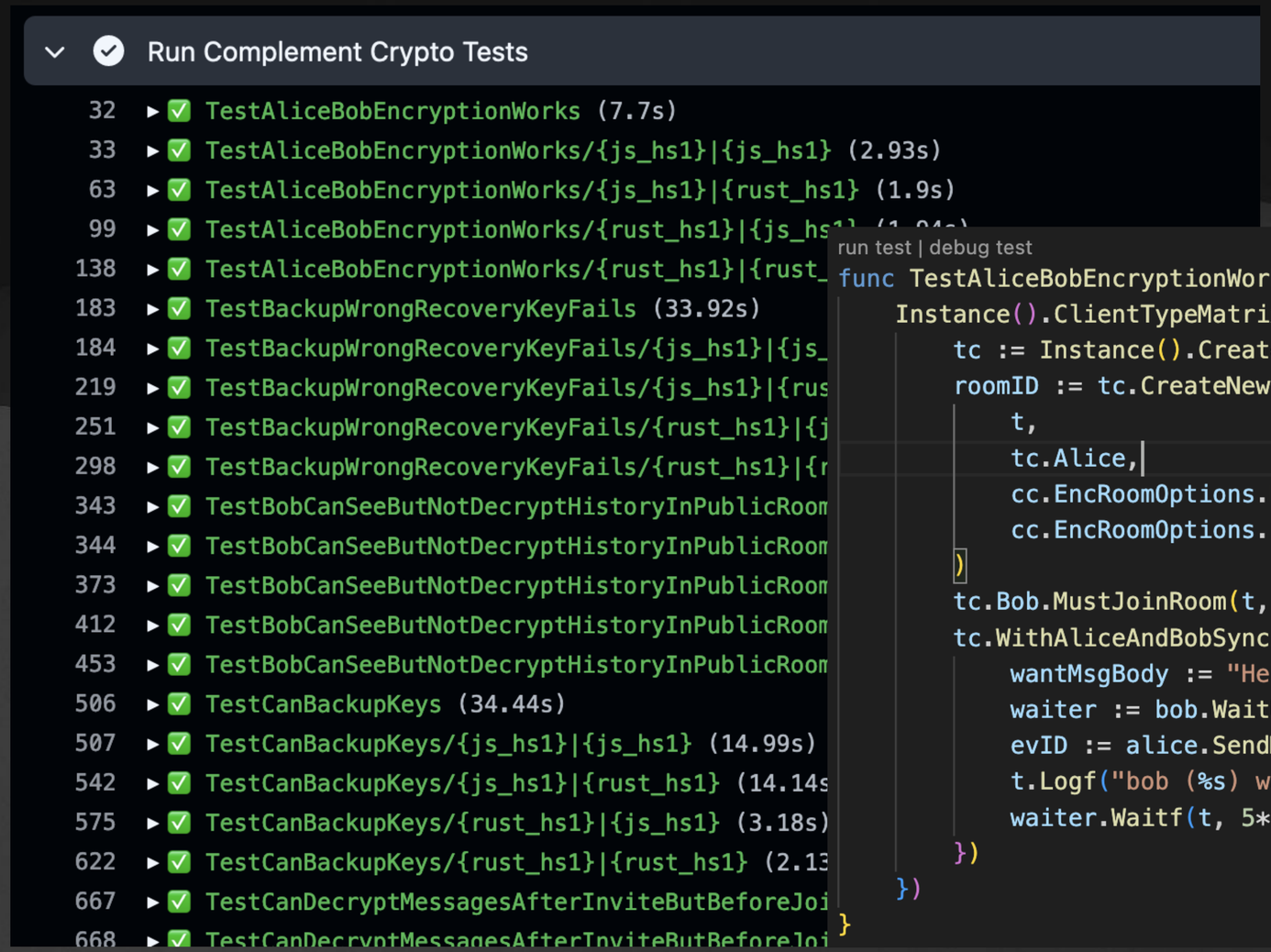
RPC clients for testing ungraceful shutdown e.g SIGKILL

Traffic monitoring in tests and HTTP dumping with mitmproxy

Adversarial attack testing with mitmproxy

Supported Github Action

Testing



run test | debug test

```go
func TestAliceBobEncryptionWorks(t *testing.T) {
    Instance().ClientTypeMatrix(t, func(t *testing.T, clientTypeA, clientTypeB api.ClientType) {
        tc := Instance().CreateTestContext(t, clientTypeA, clientTypeB)
        roomID := tc.CreateNewEncryptedRoom(
            t,
            tc.Alice,
            cc.EncRoomOptions.PresetTrustedPrivateChat(),
            cc.EncRoomOptions.Invite([]string{tc.Bob.UserID}),
        )
        tc.Bob.MustJoinRoom(t, roomID, []string{clientTypeA.HS})
        tc.WithAliceAndBobSyncing(t, func(alice, bob api.Client) {
            wantMsgBody := "Hello world"
            waiter := bob.WaitUntilEventInRoom(t, roomID, api.CheckEventHasBody(wantMsgBody))
            evID := alice.SendMessage(t, roomID, wantMsgBody)
            t.Logf("bob (%s) waiting for event %s", bob.Type(), evID)
            waiter.Waitf(t, 5*time.Second, "bob did not see alice's message")
        })
    })
}
```

Further Work

There are known gaps in the completeness of the existing Complement-Crypto tests. These tests need to be added.

Additional SDKs *must* be added: only when the entire ecosystem is tested will we reduce the absolute lower bound. Notable missing SDKs include: Dart, mautrix, matrix-bot-sdk. For languages where Go FFI bindings are impractical, defining and maintaining an RPC API will allow Client implementations in any language.

Continue to expand the dev UX: e.g allow the developer to load up the client UI for a specific user in a specific test.

Thanks to BWI for sponsoring this work