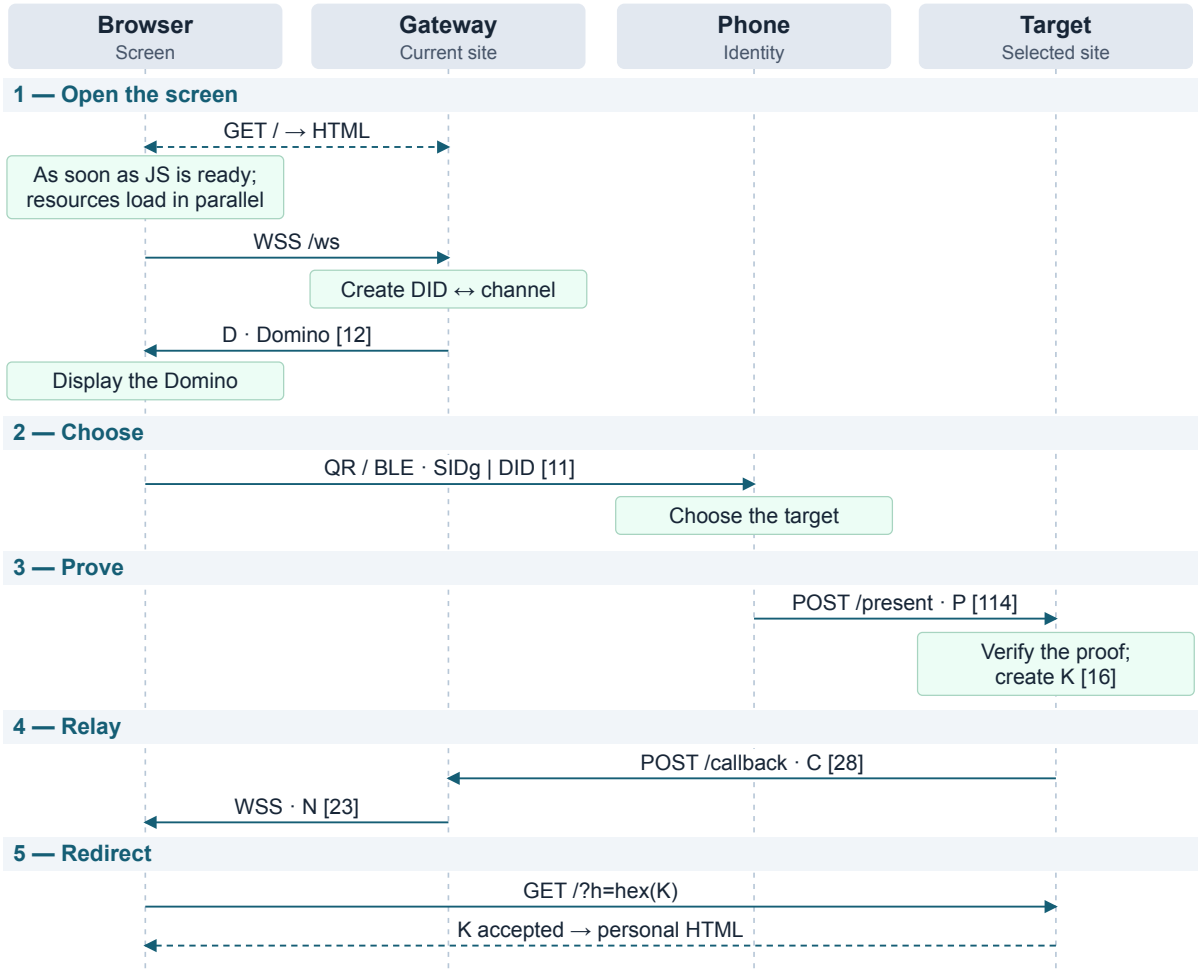
LOGIN SIGL Protocol · sequence diagram and binary formats

Revision 0.3



Message formats

`|` = concatenation; `[n]` = byte count. Opcodes are hexadecimal; offsets are zero-based; sizes include the opcode.

D · `0x44` · DOMINO

WSS gateway → browser · 12 B

```
44 | SIDg[6] | DID[5]
```

Offsets : opcode 0 ; SIDg 1 ; DID 7.
QR: Byte mode, raw `SIDg|DID`, 11 bytes, no opcode.

P · `0x50` · PRESENT

POST /present · 114 B

```
50 | SIDg[6] | DID[5] | UID[4]
   | N[33] | R[33] | z[32]
```

Offsets : SIDg 1 ; DID 7 ; UID 12 ; N 16 ; R 49 ; z 82.
SIDt belongs to the receiving HTTPS server.

C · `0x43` · CALLBACK

POST /callback · 28 B

```
43 | DID[5] | SIDt[6] | K[16]
```

Offsets : DID 1 ; SIDt 6 ; K 12.
The gateway looks up the WSS channel by DID.

N · `0x4E` · LOGIN REDIRECT

WSS gateway → browser · 23 B

```
4E | SIDt[6] | K[16]
```

Offsets : SIDt 1 ; K 7.
The browser resolves SIDt and opens `/?h=hex(K)`.

Transport and responses

Use HTTPS for all GET/POST requests; same-origin WSS at `/ws`. POST: `application/octet-stream`. Each SIGL packet occupies one complete binary WebSocket message (transport opcode `0x02`). Reassemble WebSocket fragments before decoding the SIGL packet.

`/callback`: `201` after sending N over WSS; `/present`: `201` after that response. Empty bodies. `400`: invalid format/opcode; `401`: rejected proof; `404`: unknown SID/DID; `502`: relay failure. Accepted GET: `200 text/html`. GET without `h`: public page; malformed `h`: `400`; unknown or consumed K: `303 Location: /`, removing h and loading the public page.

Identifiers and state

SIDg = gateway SID; SIDt = target SID. Each server has a fixed 6-byte SID, unique in the catalogue configured for all four actors: `SID → HTTPS origin`. DID: 5 random bytes, unique among the gateway's active channels. K: 16 random bytes; the target stores `K → (SIDt, N, UID, active)`. `hex(K)`: exactly 32 lowercase hexadecimal characters, no prefix. Use a cryptographically secure random generator.

Encoding and channel lifecycle

Unsigned integers are big-endian; SID, DID and K are opaque bytes. N and R: compressed SEC1 P-256 points (33 bytes); z: 32-byte integer, `0 ≤ z < q`. Reject incorrect sizes, invalid points and the point at infinity. Opening WSS creates DID and sends D. Closing removes DID without revoking K. K stays active until consumed. The target consumes it atomically before serving the page for identity `(SIDt,N)`. If the callback fails, revoke K and return `502` to the phone.

ZKP proof · Schnorr P-256

x: persistent user secret; r: fresh random value for every proof; both in `1…q−1`. `H = hash_to_curve(SIDt)` ; `N = xH` ; `R = rH` ; `z = (r + c·x) mod q`. Verify `zH = R + cN` and `UID = SHA256("SIGL-MINI-UID-1" | SIDt | N)[0:4]`. `c = OS2IP(XMD(transcript, 48, DSTs)) mod q` ; transcript: fields `"SIGL-ZKP-0-AUTH", SIDt, DID, N, R`, each prefixed by its u32 big-endian byte length. ASCII strings without a trailing zero; N and R encoded as SEC1; `[0:4]` = first four bytes.

Cryptographic parameters

Curve: P-256 (secp256r1), order q: `FFFFFFFF00000000FFFFFFFFFFFFFFFF BCE6FAADA7179E84F3B9CAC2FC632551` (concatenated hex). H : RFC 9380, suite `P256_XMD:SHA-256_SSWU_RO_`, message = raw SIDt, replace the suite DST with DSTh: `SIGL-ZKP-V00-CS01-P256_XMD:SHA-256_SSWU_RO_`
DSTs : `SIGL-ZKP-V00-CS01-P256-SCALAR`. XMD = `expand_message_xmd` SHA-256 (RFC 9380 §5.3.1) ; output length 48 bytes for c; OS2IP = unsigned big-endian integer.

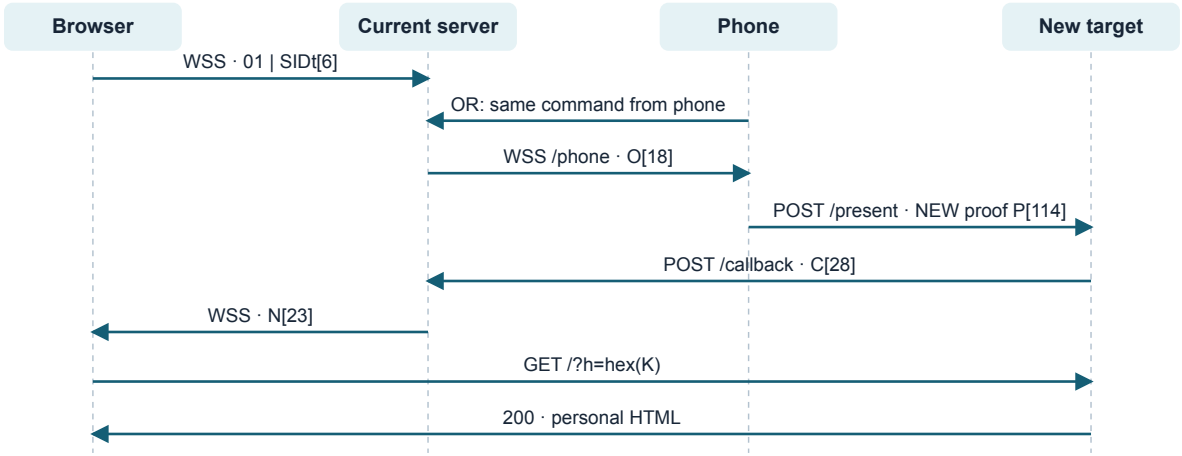
BLE Domino reading

GATT list: `D0 | count[2] | (SIDg[6] | DID[5]) × count` ; count is big-endian; length `3 + 11×count`. Advertise the service UUID only; connect, then read the characteristic. Service : `5349474C-424C-4500-0000-000000000001` ; characteristic: `5349474C-424C-4500-0000-000000000002`.

NAVIGATE / LOGOUT Screen control · phone generates every new target proof

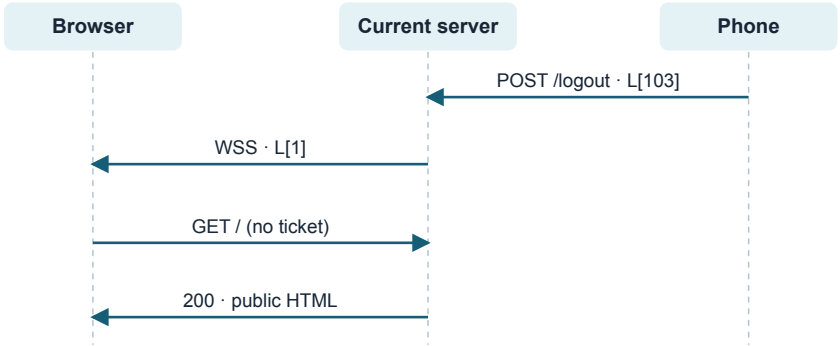
NAVIGATE

Personal browser and authenticated phone channels are already open.



LOGOUT

Server resolves the personal screen by the proven identity N.



Personal browser and phone channels

After GET consumes K, personal HTML opens `/ws?h=hex(K)` on its target. Require a consumed local K with no previous WSS claim; atomically mark it claimed, bind a new DID/WSS to `(SIDc,N)`, and send D[12]. K cannot authorize another GET or WSS (`401`). Public pages still open `/ws`. Allow one personal WSS per `(SIDc,N)`; reject a second with `409` before claiming K. Closing it removes both DID and identity bindings.

After `/present` returns `201`, the phone opens same-origin WSS `/phone` on the new target and sends A[103] first. Verify A, then bind that phone channel to `(SIDc,N)`, independently of browser arrival. Allow one phone channel per identity. Closing it does not log out the browser. Channel setup adds no acknowledgement to LOGIN.

A · `0x41` · PHONE AUTH

First /phone WSS message · 103 B

`41 | UID[4] | N[33] | R[33] | z[32]`
Offsets: UID 1; N 5; R 38; z 71.

`0x01` · NAVIGATE REQUEST

Browser or phone → current server · 7 B

`01 | SIDt[6]` · SIDt offset 1.
Send on the bound browser `/ws` or authenticated `/phone`.

O · `0x4F` · PROOF REQUEST

Current server → phone · 18 B

`4F | SIDt[6] | SIDg[6] | DID[5]`
Offsets: SIDt 1; SIDg 7; DID 13. SIDg = SIDc.
DID selects the current browser WSS; SIDt selects the new target.

L · `0x4C` · LOGOUT

POST /logout · 103 B; browser WSS · 1 B

Request: `4C | UID[4] | N[33] | R[33] | z[32]`
Offsets: UID 1; N 5; R 38; z 71.
Display command: `4C` only, sent on the bound browser WSS.

Proof for A and L

SIDc = receiving/current server SID. Use the page 1 curve, x, fresh r, H = hash_to_curve(SIDc), N, R, UID and z with SIDt = SIDc. Compute c from length-prefixed `domain, SIDc, N, R`, with the same XMD and DSTs. A domain: `"SIGL-ZKP-0-PHONE"`; L: `"SIGL-ZKP-0-LOGOUT"`. Verify UID and `zH = R + cN`. Never bind channels by UID alone.

NAVIGATE processing

Resolve 01 through the sender's bound identity; require its active browser and phone channels and a known SIDt. Send O to that phone. The phone checks SIDg = channel origin SID, computes a fresh P[114] for SIDt using the LOGIN transcript, then POSTs directly to SIDt. C[28] → N[23] → GET follow page 1 unchanged. Keep the old browser personal until N arrives; send no preliminary L. On `201`, the phone opens/authenticates `/phone` on SIDt and closes its old phone channel.

LOGOUT processing

POST L to SIDc. Resolve proven `(SIDc,N)` to its active personal WSS. Send L[1], remove its identity/DID bindings, return empty `201`. No callback or ticket. The browser clears personal content and Domino, closes WSS and calls `location.replace("/")`. The public page opens a new WSS/DID. On `201`, the phone closes that server's phone channel. Stored personal data and screens on other servers are unaffected.

Transport and failures

HTTPS for HTTP; POST L uses `application/octet-stream`. WSS uses complete binary messages. L errors: `400` format; `401` proof/UID; `404` no active screen; `502` send failure. WSS: close `1002` on invalid format, `1008` on failed auth, duplicate phone or missing/unknown route; `1011` on relay failure. Failure sends no browser switch/logout. No scan, polling, delayed switch or rendering acknowledgement. `201` confirms forwarding; it does not confirm HTML rendering.