ID Card Picker: Consent UX That Prevents Undersharing, Oversharing, and Replay

People & In-Person / Last updated 2026-06-11 / https://www.scrambleid.com/learn/unified-id-card-picker-consent

> **Status (June 2026):** Early access. The ID Card Picker ships as part of the People verification family, which is live with early-access customers and isn't generally available yet. This article describes the shipped design as it stands today; talk to your ScrambleID account team about access and timelines.

In one sentence: The ID Card Picker is the consent UX that lets a verifier declare a **minimum required profile** (Work / Minimal / Anonymous / None) while ensuring the presenter can preview and consent, without forcing overshare and without allowing under-share.

TL;DR (canonical)

- The picker enforces *minimum* requirements while preserving presenter autonomy over optional fields.
- Strict requirements reduce UI complexity (Accept/Reject or Filtered Picker) instead of increasing it.
- The presenter always sees a preview of exactly what will be shared before tapping **Accept & Share**.
- The picker is a security control: it prevents under-sharing (missing mandatory fields) and reduces oversharing.

The core problem

People verification fails when:

- verifiers can't request what they need (undersharing), or
- presenters over-share sensitive data (oversharing), or
- the process is slow enough that teams revert to screenshots.

The picker is designed to make the safe path the fastest path.

Requirement pill (what the verifier sets)

The verifier chooses a simple requirement:

- **Need: None** → presenter can share any profile
- **Need: Work** → presenter must share a Work-compliant view
- **Need: Minimal** → presenter must share a minimal compliant view
- **Need: Anonymous** → presenter shares only handle + human-verified cues

This requirement is expressed as a stable data contract.

Requirement data contract (concept)

```
{
  "requiredProfile": {
    "level": "WORK",
    "mandatoryFields": ["legalName", "companyName", "title"]
  }
}
```

> **Cryptographic detail:** how a picker share is signed by the presenter's device key and bound to the Trust Check session (payload structure, atomic validation, replay handling) is specified in Session binding cryptography in the architecture reference.

Receiver UI states (S0 / S1 / S2)

The receiver experience is context-adaptive:

State	Trigger	Receiver UI	Why it matters
S0 Full Picker	no requirement	full set of profile tiles	maximizes choice when safe
S1 Accept/Reject	requirement set and last-used profile already complies	single preview + Accept/Reject	fastest path, reduces taps
S2 Filtered Picker	requirement set and last-used profile non-compliant	only compliant tiles; confirm disabled until compliant	prevents under-share without nagging

Modify / Quick Edit (bounded autonomy)

Even when the receiver sees S1 Accept/Reject, they should be able to tap **Modify** to:

- toggle optional fields *without breaking compliance*
- keep mandatory fields locked

Example:

- Work requirement mandates Name + Company + Title.
- Receiver may toggle LinkedIn or Department OFF.
- Receiver may not hide Name/Company/Title.

Admin controls (enterprise policies)

Recommended enterprise toggles:

- Default requirement pill to **Work** for enterprise tenants
- Disable Anonymous tile (if required)
- Lock mandatory fields (directory-backed)

Analytics (what to log)

The picker is also an instrumentation surface. Log:

- receiver_state: S0 / S1 / S2
- accept / reject / modify_used
- time_in_picker
- overrides (optional fields toggled)

This yields defensible metrics for rollout and iteration.

The picker flow (sequence)

The picker is easiest to review when you treat it like a strict, replay-resistant transaction:

```
sequenceDiagram
    participant V as Verifier
    participant P as Presenter
    participant S as ScrambleID

    V-->>S: Create request (requiredProfile + DID + TTL)
    S-->>V: Render requirement pill + DID/QID
    V-->>P: Present request (in person / QR / code)
    P-->>S: Open picker (shows requirement + preview)
    P-->>P: Review preview (mandatory locked)
    alt Compliant default (S1)
        P-->>S: Accept & Share
    else Non-compliant default (S2)
        P-->>P: Choose compliant tile / modify optional fields
        P-->>S: Accept & Share
    end
    S-->>V: Return signed view + provenance (single-use)
```

Consent copy templates (copy/paste)

Use simple, non-legalistic language. The goal is for a non-technical user to understand **what** will be shared and **why**.

Requirement pill labels

- **Need: Work** → "Share your work identity (Name, Company, Title)"
- **Need: Minimal** → "Share a minimal identity (Name + one verifier cue)"
- **Need: Anonymous** → "Share an anonymous confirmation (handle + verified cues)"

Preview header

- "You're about to share this info with **{VerifierName}**."
- "Required fields are locked. You can toggle optional fields."

Accept button

- **Accept & Share**

Reject button

- **Reject** ("Nothing is shared")

Modify link

- **Modify** ("Optional fields")

Optional field disclosure hint

- "Optional fields help the verifier finish the workflow faster."

Threat model + mitigations (why this UX is a security control)

Threat	What happens in the real world	Picker mitigation	Recommended policy add-ons
Undershare	presenter omits required field; workflow fails; teams revert to screenshots	requirement pill + S2 filtered picker	lock required fields in enterprise tenants
Overshare	presenter shares too much; sensitive fields leak	optional fields default OFF; preview before share	enforce minimal defaults for high-risk workflows
Replay / forwarding	screenshot/forwarded card reused later	bind to DID + short TTL; single-use responses	require step-up (XFactor, in development) for risky verifications
Shoulder-surfing	bystander views on-screen details	minimal default views; short preview; optional redactions	allow "privacy mode" for sensitive attributes
Coercion	presenter is pressured to share more than necessary	requirement sets a ceiling (minimum) not a demand for extras	add "deny and escalate" path (supervisor today; Lockstep, in development, once it ships)

Accessibility + localization considerations

- **Time limits:** if the request TTL is short, warn early and allow a simple "Try again" flow rather than a hard failure.
- **Screen readers:** ensure the preview reads "required vs optional" fields distinctly.
- **Localization:** do not translate requirement levels (WORK/MINIMAL) for policy logic; translate only display labels.
- **Color safety:** never rely on color alone for "required" vs "optional" (use icons/text).

Key Takeaway

The ID Card Picker is a consent UX that prevents three failure modes: undersharing (user omits required fields), oversharing (user reveals more than necessary), and replay (user reuses a stale

share). It enforces policy-required fields, filters available attributes to what's relevant, and generates single-use, time-bounded verification tokens.

FAQ

Does the picker force oversharing?

No. It enforces a minimum (mandatory fields) but preserves control over optional fields.

Why not let the receiver always choose any profile?

Because many workflows require a minimum to be safe and operationally useful (e.g., Work identity for a field technician).

How does this help security?

It prevents under-share (missing required fields) and reduces overshare (sensitive fields default OFF). Combined with provenance, it reduces spoofing.

How does this help adoption?

The fast path (S1) is one tap in the common case, which is essential for non-technical users.

Can we measure if requirements are too strict?

Yes. If S2 state frequency and reject rates spike, your requirement defaults are too strict or your profiles aren't well configured.

References (public)

- WCAG 2.2 (timeouts / enough time): https://www.w3.org/TR/WCAG22/
- Understanding WCAG 2.2.1 Timing Adjustable: https://www.w3.org/WAI/WCAG21/Understanding/timing-adjustable.html
- NIST Privacy Framework (Version 1.0): https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.01162020.pdf

Related reading

- People Trust Checks: Overview
- People Verification Implementation Guide
- Unified ID Card: Attribute Provenance