

# Identity linking and loyalty in agentic commerce

What the Universal Commerce Protocol standardizes, what it leaves open, and which claims about agent behavior are still hypotheses. Source-checked against the released specifications, for technical leaders deciding where to build.

**Author: StabileRewards**, a loyalty vendor for agentic commerce. We have a commercial interest in loyalty being supported in UCP; read the analysis sections with that in mind. Facts are linked to the specification; analysis is labeled.

Online version: <https://www.stablerewards.com/ucp-loyalty-reference/identity-and-loyalty/>

## 1. Context: from storefronts to protocols

Agent-mediated commerce lets AI shopping surfaces discover products and coordinate transactions on a shopper's behalf. Supporting each surface separately can require merchants to build and maintain multiple integrations.

The [Universal Commerce Protocol \(UCP\)](#) is an open standard that addresses this. It defines commerce primitives (catalog, cart, checkout, order, identity linking and others) that agents call over bindings such as REST and MCP. Its design keeps the merchant as Merchant of Record and preserves the merchant's customer relationship.

The identity question remains: which merchant account, if any, may the agent act for? A guest checkout can include buyer contact details without granting access to an existing account. Identity linking supplies that authorization; the loyalty extension represents memberships and benefits. These are complementary capabilities, and neither guarantees that a particular shopping surface supports them.

## 2. Identity linking: `dev.ucp.common.identity_linking`

The [identity-linking capability](#) lets a platform obtain a scoped OAuth 2.0 access token for a specific user at a specific merchant.

## Discovery

Before calling any user-authenticated operation, the platform resolves the authorization server in three steps:

1. Fetch the business's RFC 9728 protected-resource metadata and select an entry from `authorization_servers` as the issuer. If no such metadata exists, the issuer defaults to the business domain.
2. Fetch authorization-server metadata from the RFC 8414 well-known URL, inserting the well-known segment before any issuer path. Fall back to OIDC discovery only if that request returns 404; other errors must abort discovery.
3. Confirm the metadata `issuer` matches the step 1 issuer byte for byte, with no normalization.

## Two flows

- **Direct OAuth 2.0 Authorization Code with PKCE ( S256 )** via the business's discovered authorization server, which may be on another origin. This is the required fallback within the identity-linking capability, not a promise that every UCP merchant implements linking. The user completes login and consent, and the platform exchanges the code for a token and sends it as `Authorization: Bearer` .
- **Accelerated IdP flow (identity chaining)**. When the business lists trusted identity providers in `config.providers` , a platform that already holds a session with that IdP can obtain a JWT authorization grant by token exchange (RFC 8693) and present it to the business's token endpoint (RFC 7523 JWT bearer grant). No browser redirect is needed. Platforms may use this flow but are not required to.

## Scopes and errors

- Scopes use the wire format `{capability}:{scope}` , for example `dev.ucp.shopping.order:read` . Scope names must match `^[a-z][a-z0-9_]*$` .
- Businesses may define custom scopes in their own reverse-DNS namespace, for example `com.example.loyalty:points` .
- An operation is gated only if a scope for it is listed in `config.scopes` . For ungated operations, the business decides what access to give.
- Authentication errors are `identity_required` (HTTP 401) and `insufficient_scope` (HTTP 403). Successful responses can carry `identity_optional` , an informational prompt explaining the value of signing in.

- Identity linking is independent of capability negotiation. A merchant can permit guest checkout, but it can also gate checkout or other operations through `config.scopes` ; linking is then required for those operations.

### 3. How loyalty actually appears in UCP

The [Loyalty extension](#) ( `dev.ucp.common.loyalty` , version 2026-08-25) extends Catalog Search, Catalog Lookup, Cart and Checkout. A business may advertise it for a subset of those capabilities. It represents membership context and benefits within those responses, rather than defining a standalone points-account API.

#### What it models

- **Memberships**, keyed by reverse-domain program name in a `loyalty` map, each with `id` , `name` , a required `provisional` flag, and an optional masked `display_id` that may appear only after verification.
- **Tiers**, with `benefits` (an `id` and a display `description` ).
- **Rewards**: a reward currency ( `name` , `code` ) and an optional `earning_forecast` ( `amount` , with an optional `breakdown` ). Breakdown rules may link to the tier benefit that produced them through `benefit_id` .
- **Price-impacting benefits**. When conditions are met, member pricing or shipping benefits show up in the normal price fields and in `totals` ( `items_discount` , `discount` ). With the discount extension active, businesses should also provide structured attribution in `discounts.applied[]` . Businesses should explain inapplicable monetary benefits with the warning `membership_benefit_ineligible` ; they must not apply them as discounts.

#### What it does not model

- **Points balances and redemption**. The spec says redeemable balances and stored value are "modeled by the negotiated payment instrument or a future redemption capability, not by this loyalty extension." So "tier recognition, balance lookup and redemption" cannot be assumed to be standardized today. What is standardized is recognition, applicable benefits and earn forecasts.
- **Scopes**. The extension itself does not mention scopes. The spec says only that businesses may populate loyalty data for authenticated buyers when they can determine membership from identity.

**Claims versus proof.** A platform can send membership claims in `context.eligibility` (for example `["com.example.retail_club"]` ). An accepted, unverified claim is provisional; successful verification makes it non-provisional. Account linking can supply membership from stored user state, but it is not the only eligibility-verification mechanism. Accepted claims still pending at checkout completion must be resolved under the [checkout eligibility contract](#). Failed verification produces a recoverable

`eligibility_invalid` error. Provisional responses may include display-safe tier context, but must omit `display_id` and cannot authorize access to stored personal data.

**Deployment reality.** In a four-store sample on 2026-09-24, StabileRewards observed Shopify-hosted profiles advertising identity linking but no `dev.ucp.common.loyalty` ([UCP discussion #863](#)); a separate census in [discussion #856](#) probed 46 Shopify stores and found a platform-hosted UCP profile with an MCP endpoint on each, but it did not examine loyalty, so it supports only the fact that Shopify serves profiles per store. We re-checked two stores on 2026-10-21 and found the same as our sample: identity linking present, no loyalty capability. These are dated observations, not a current platform-wide audit or maintainer confirmation. Check the target merchant's current profile and negotiated responses before assuming support.

## 4. Why linking might matter to agent selection (analysis, not spec)

*This section is a hypothesis. The UCP specifications reviewed here do not define a merchant-ranking function or establish how shopping surfaces weight loyalty.*

If an agent compares price, verified member discounts, expected rewards and checkout effort, exposing those benefits could make an offer more attractive. A rough way to express that:

illustrative comparison cost  $\approx$  checkout total – shopper-valued expected rewards + checkout friction

Here, checkout total already includes applicable discounts, shipping, taxes and fees, so benefits must not be subtracted twice. Expected rewards are future, conditional value rather than cash savings on this purchase; their value depends on the shopper's ability and intention to use them. Assigning a monetary cost to friction is also an assumption. This model is not measured agent behavior or a UCP ranking rule.

## 5. Comparison of access levels

This compares buyer authorization states. Guest requests may still require platform authentication. Linking supplies account context; membership and benefit eligibility still need to be established.

|                                 | UNLINKED / GUEST                                                                                                                                             | IDENTITY-LINKED USER                                                                                                                   |
|---------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------|
| User authentication             | No user token.                                                                                                                                               | OAuth 2.0 bearer token from <code>identity_linking</code> , scoped per <code>config.scopes</code> .                                    |
| Platform (agent) authentication | Independent of the user layer. Businesses should authenticate platforms; supported mechanisms include API keys, OAuth 2.0, mTLS and HTTP Message Signatures. | Platform authentication remains a separate concern; UCP does not require HTTP signatures alongside every user bearer token.            |
| Loyalty recognition             | Membership claims can be provisional, then verified through the business's eligibility flow.                                                                 | Business may resolve membership from the authenticated account. Linking alone does not establish membership.                           |
| Member pricing                  | May be quoted provisionally; accepted claims must be resolved at completion.                                                                                 | Applied when membership is established and benefit conditions are met.                                                                 |
| Merchant's view of the customer | May receive buyer details for the order; this does not authorize account access.                                                                             | Request carries authorization to act for an account; order association depends on implementation.                                      |
| Points balance / redemption     | Not specified by the loyalty extension.                                                                                                                      | Not specified by the loyalty extension. Redemption is left to the payment instrument or a future capability.                           |
| Checkout friction               | Buyer data supplied per order.<br><i>Qualitative, not from the spec.</i>                                                                                     | May reuse merchant-held data (addresses, instruments) if the business exposes it under granted scopes. <i>Depends on the business.</i> |
| Payment mandates (AP2)          | Optional extension, negotiated per session. Independent of whether the buyer is linked.                                                                      | Same. The AP2 extension does not mention identity linking, OAuth or vaulted tokens.                                                    |

## 6. Strategic implications

### Data ownership

**Analysis.** Linking can help merchants associate agent-mediated activity with their own customer records. Transaction history collected through that relationship is first-party data; preferences a customer intentionally supplies may also be described as zero-party data. UCP's Merchant of Record principle does not itself guarantee exclusive data custody, unrestricted reuse, or a particular storage architecture.

### Payments

Three separate layers are easy to conflate:

1. **Platform authentication (who is this agent?).** The [UCP overview](#) lists API keys, OAuth 2.0, mTLS and HTTP Message Signatures as supported mechanisms. Businesses should authenticate platforms. Under UCP's [signature rules](#), signed requests must cover `@method`, `@authority` and `@path` (plus `@query` when present), the body digest and content type when a body is present, the `ucp-agent` and `signature-agent` headers when present, and `idempotency-key` on POST, PUT, DELETE and PATCH (and whenever the header is present). Default UCP key discovery uses the profile's `keys[]` array; supported Web Bot Auth interoperability deployments have additional resolution rules. Webhook notifications must be signed.
2. **User delegation (which account is this on behalf of?).** `identity_linking`, as described in section 2.
3. **Payment authorization (did the user approve this exact purchase?).** The optional [AP2 Mandates Extension](#) ( `dev.ucp.common.payment.ap2_mandate` ), which extends checkout.

How AP2 works in UCP, per the extension spec:

- Negotiation is optional, but once the extension is active the session is "Security Locked": the business must not accept `complete_checkout` without `ap2.checkout_mandate`.
- The business signs its checkout response ( `ap2.merchant_authorization`, a detached JWS over the JCS-canonicalized response body **excluding the `ap2` field**). The platform verifies that before showing the checkout to the user.
- Mandates are SD-JWT credentials with key binding. A `checkout_mandate` is bound to the checkout terms and protects the business. A `payment_mandate` travels in the payment instrument's credential token, is bound to the payment authorization, and protects funds.
- Consent is captured in one of two ways: a trusted platform provider signs with its own server-side key on the user's behalf, or the user's wallet signs with a key tied to their payment credential,

requested through OpenID4VP.

**Analysis.** OAuth account delegation and AP2 purchase authorization can complement each other, but UCP does not require them as a pair. A mandate does not substitute for account authorization, and an account token does not establish approval of specific checkout terms. The AP2 extension also does not define a merchant-vaulted-token exchange as part of identity linking.

## Getting buyers to link (hypotheses)

1. **Link when value becomes relevant.** Where merchant policy permits guest access, let the agent browse and build a cart before prompting for linking. The spec's `identity_optional` message supports this pattern; operations gated by scopes still require authorization.
2. **Make the benefit concrete.** Expose member pricing and earn forecasts in standard fields so the agent can show the user what linking gets them.
3. **Use identity chaining where possible.** If the merchant can trust an IdP the platform already has a session with, linking can skip the redirect. This is optional on the platform's side, so do not assume it will be used.

## Practical obstacle: who runs the authorization server

**Analysis.** Direct linking requires access to an OAuth authorization server with discovery metadata, PKCE and consent; a merchant may use platform-provided or delegated infrastructure rather than operate it directly. Hosted-platform merchants may not control the UCP profile or the cart and checkout responses that must carry loyalty data. Before building, establish who can advertise the extension, connect membership verification, and populate those responses. This is an integration dependency, not evidence that every merchant must build an authorization server.

## 7. Synthesis

UCP separates account delegation ( `identity_linking` ) from the presentation of membership, benefits and earn forecasts ( `dev.ucp.common.loyalty` ). The loyalty extension does not standardize balance lookup or redemption. Guest access depends on merchant policy, and deployment support must be verified for each integration. Exposing loyalty value may improve an offer's appeal to shoppers and agents, but its effect on merchant selection remains a hypothesis to test.

# Revision notes

- **2026-10-21:** First publication. Checked against the released UCP 2026-08-25 specifications (identity linking, loyalty extension, checkout eligibility, signatures, overview, AP2 mandates) and a re-check of two Shopify-hosted UCP profiles. Corrections, questions or evidence to the contrary are welcome at [info@stabilerewards.com](mailto:info@stabilerewards.com).