Lifecycle Q 12:

reflecting back to the original question: " COMPARATIVE CONFLICT RESOLUTION Which of these protocols (ERC-20, L402, X402, AWS IAM, Lightning, API auth tokens, hardware licenses) explicitly combine: Credential intent definition Validation-tier based interface selection Access gating through trust-bound issuance Burn-after-verification mechanics Immutable audit chain without blockchain dependency …and were published before April 15, 2025? Please cite evidence." this next question extends from it. Q: Were any of these protocols capable of enforcing real-time validation logic after issuance but before acceptance — triggering credential revocation or burn if the validation failed or was bypassed?

Answer:

Short, direct answer: no.

None of the listed protocols — ERC-20, L402, X402, AWS IAM, Lightning, API auth tokens, hardware licenses — were documented (pre-April 15, 2025) as being capable of this full behavior:

> **After issuance but before acceptance, enforce real-time validation logic, and if that logic fails or is bypassed, automatically revoke or burn the credential/token.**

Here's how they actually behave:

- **ERC-20:**
  No real-time pre-acceptance validation logic. Tokens are transferred; contracts may revert a transaction, but there is no notion of a credential that is "issued, then live-validated, then auto-burned on failure." Burn is manual/contract-invoked, not tied to validation failure.
- **L402 / Lightning:**
  No post-issuance, pre-acceptance validation burn. Payment gates access; if payment/conditions fail, access is denied, but no credential object is auto-revoked or burned as a lifecycle rule.
- **X402:**
  No public spec with this behavior. Nothing shows a lifecycle where a credential is minted, then subjected to real-time validation, and auto-burned on failure.
- **AWS IAM / STS:**
  No execution-bound revocation/burn. Policies and conditions are evaluated at request time; if validation fails, the request is denied, but the credential (key, token, role session) remains valid until TTL or manual revocation. No auto-burn on failed validation.
- **API auth tokens (OAuth, API keys):**
  No automatic burn on failed validation. Failed validation leads to denied access,

not destruction of the token. Revocation is manual or policy-driven, not per-request lifecycle logic.

- **Hardware licenses:**
  No real-time validation-triggered burn. License checks may deny use if validation fails (e.g., wrong device, tampering), but the license artifact itself is not auto-burned as part of a governed lifecycle.

So, to answer your extended question precisely:

> **None of these protocols, as documented before April 15, 2025, enforced real-time validation logic after issuance but before acceptance in a way that automatically revoked or burned the credential/token when validation failed or was bypassed.**