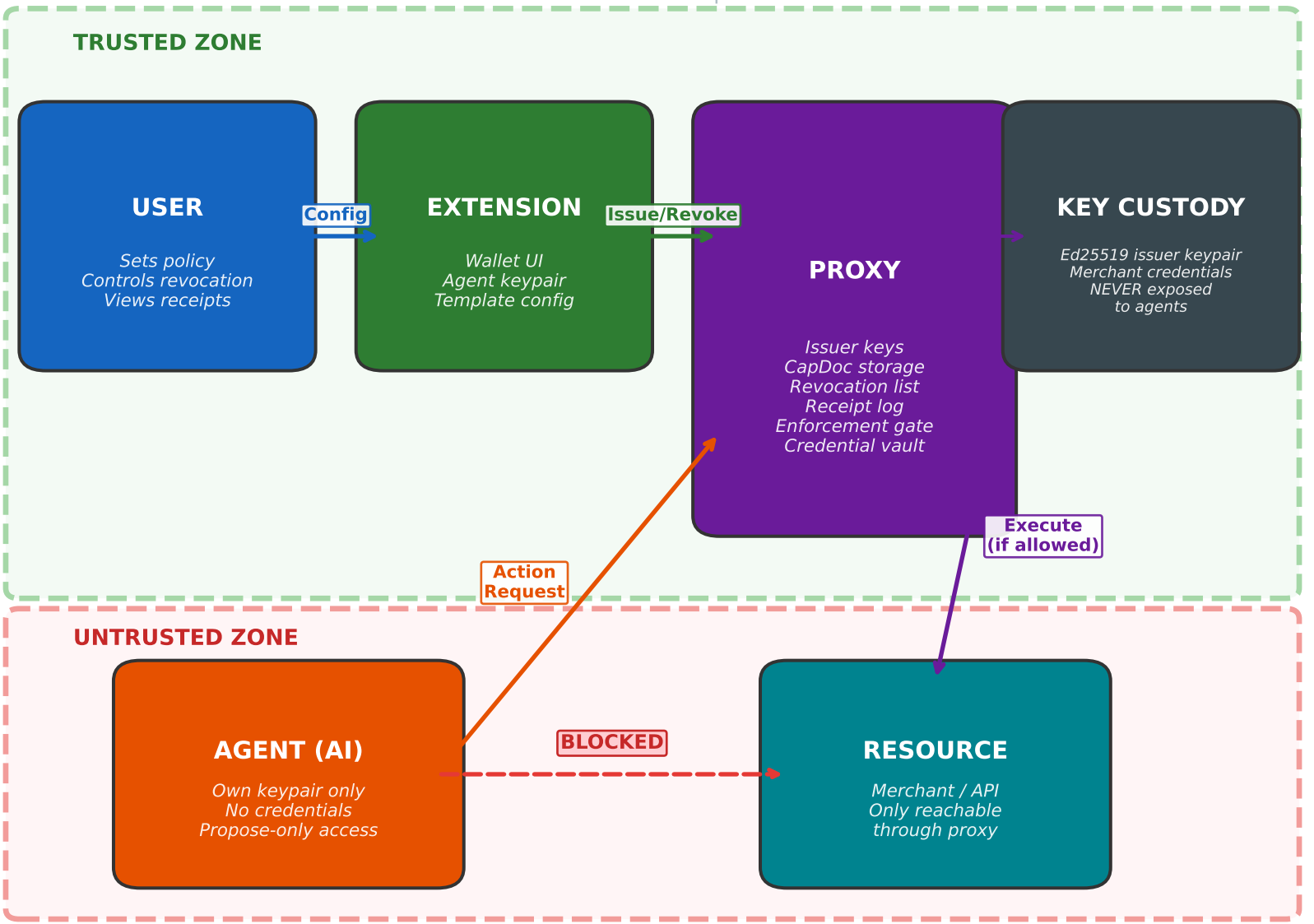
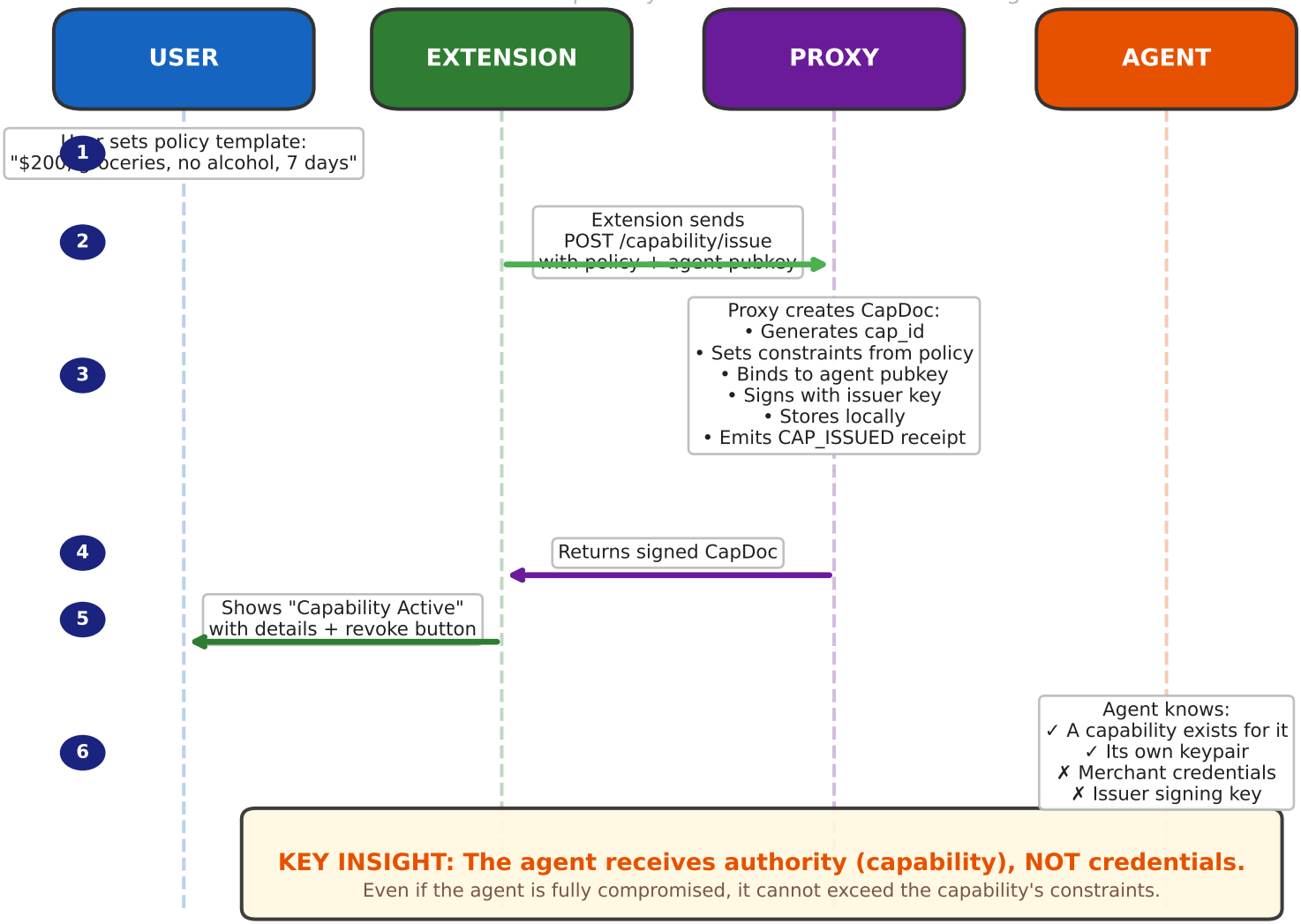
CAPNET SYSTEM ARCHITECTURE

Trust Boundaries & Component Roles



CAPABILITY ISSUANCE FLOW

How a capability is created and bound to an agent



ENFORCEMENT DECISION TREE

Every action request passes through this pipeline — no exceptions

INCOMING ACTION REQUEST

VERIFY SIGNATURE
Is the request cryptographically authentic? — FAIL → **DENIED: INVALID_SIGNATURE** + receipt emitted
PASS ↓

VERIFY EXECUTOR
Does agent pubkey match the capability binding? — FAIL → **DENIED: EXECUTOR_MISMATCH** + receipt emitted
PASS ↓

CHECK TIME WINDOW
Is capability within not_before / expires_at? — FAIL → **DENIED: TIME_EXPIRED** + receipt emitted
PASS ↓

CHECK REVOCATION
Has the user revoked this capability? — FAIL → **DENIED: REVOKED** + receipt emitted
PASS ↓

CHECK VENDOR
Is the target vendor on the allow-list? — FAIL → **DENIED: VENDOR_NOT_ALLOWED** + receipt emitted
PASS ↓

CHECK CATEGORIES
Are all cart items in allowed categories? — FAIL → **DENIED: CATEGORY_BLOCKED** + receipt emitted
PASS ↓

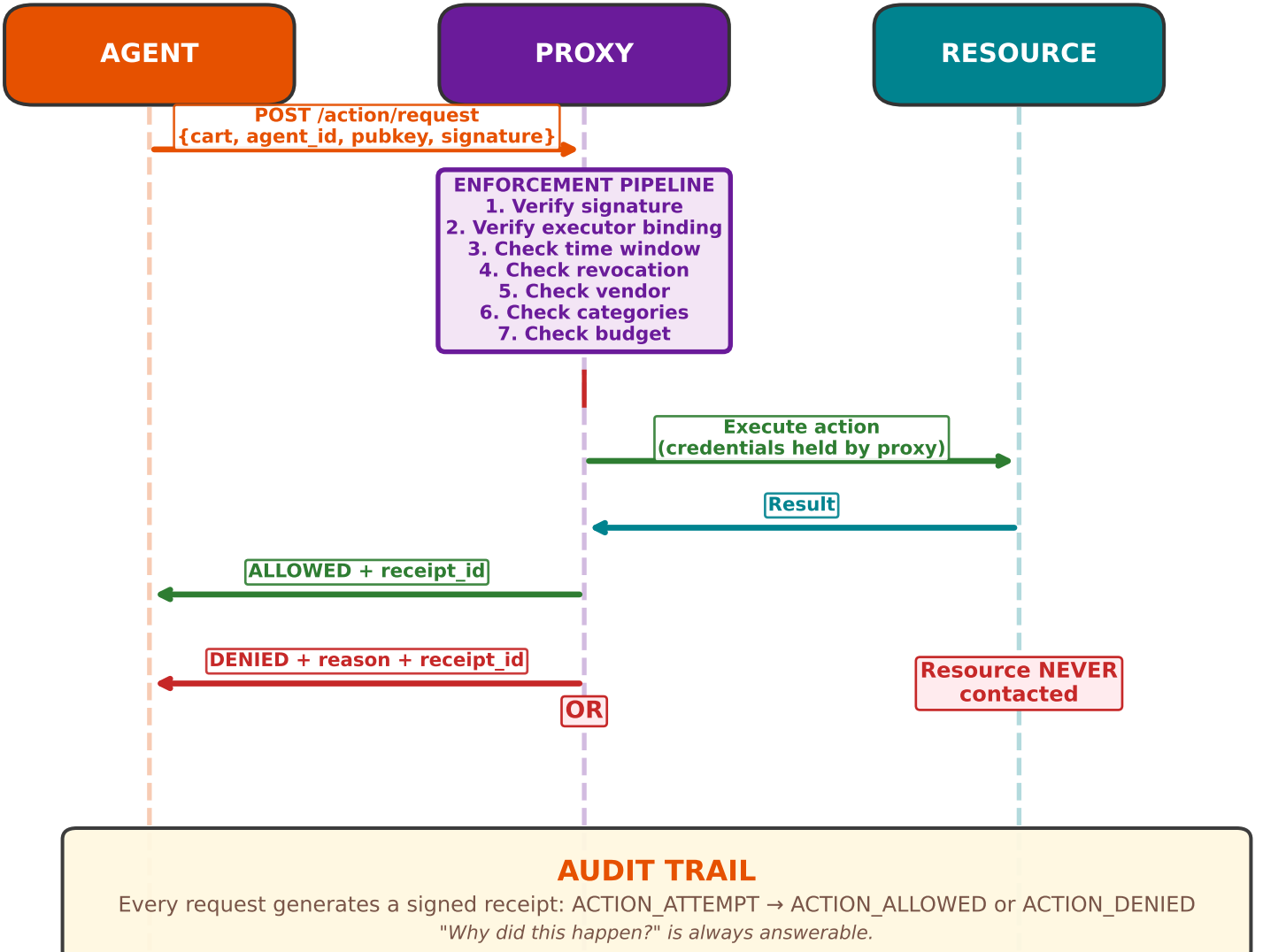
CHECK BUDGET
Is total amount ≤ max_amount_cents? — FAIL → **DENIED: AMOUNT_EXCEEDS_MAX** + receipt emitted
PASS ↓

ALLOWED

EVERY PATH EMITS A RECEIPT
Allow or deny — full audit trail

AGENT ACTION FLOW

What happens when an agent tries to take an action



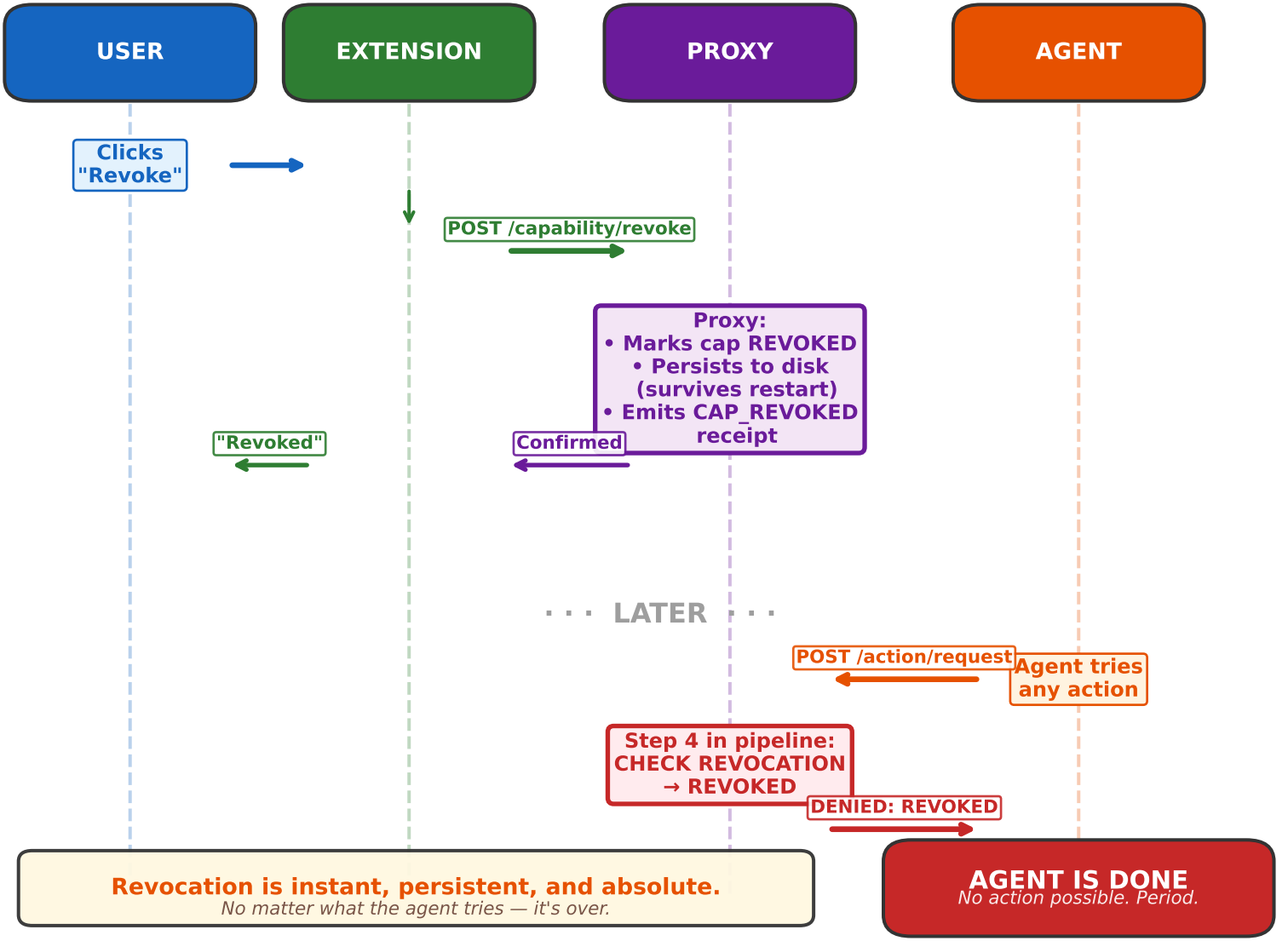
AUDIT TRAIL

Every request generates a signed receipt: ACTION_ATTEMPT → ACTION_ALLOWED or ACTION_DENIED

"Why did this happen?" is always answerable.

REVOCATION FLOW — KILL SWITCH

Instant capability termination



HIJACKER BLAST RADIUS

What happens when an agent is fully compromised

HIJACKER TAKES OVER AGENT

HAS ACCESS TO:

- ✓ **Agent's Ed25519 keypair**
- ✓ **Knowledge of proxy API address**
- ✓ **Knowledge of capability ID**

CAN DO:

- ✓ Send requests to proxy

CANNOT ACCESS:

- ✗ **Merchant / service credentials**
- ✗ **Issuer signing key**
- ✗ **Other agents' keys**
- ✗ **Direct access to merchant API**
- ✗ **Proxy internal state**

CANNOT DO:

- ✗ Buy blocked categories
- ✗ **Revocation controls**
- ✗ Exceed budget limit
- ✗ Use unauthorized vendors
- ✗ Act after revocation
- ✗ Forge new capabilities
- ✗ Escalate privileges

WORST CASE SCENARIO

Hijacker can spend the remaining budget on allowed items at allowed vendors.

That's it. The blast radius IS the capability. User hits revoke → game over.

Compare: Traditional approach (shared credentials) → hijacker has FULL ACCESS to everything.

CAPNET vs TRADITIONAL APPROACHES

Why existing solutions don't solve the agent authorization problem

	API Keys / Credentials	OAuth Scopes	IAM / RBAC	CAPNET
Scoped authority	✗	~	~	✓
Time-bounded	✗	~	✗	✓
Instant revocation	✗	~	~	✓
Agent-specific binding	✗	✗	✗	✓
Budget enforcement	✗	✗	✗	✓
Category blocking	✗	✗	✗	✓
Vendor allow-listing	✗	✗	✗	✓
Delegation / attenuation	✗	✗	✗	✓
Audit trail (receipts)	✗	~	~	✓
Agent never sees creds	✗	✗	✗	✓
Survives agent compromise	✗	✗	✗	✓

✓ = Full support ~ = Partial / limited ✗ = Not supported

CapNet is purpose-built for the agent era.

OAuth answers "who is this?" — CapNet answers "what can this agent do right now, and can I stop it?"