

Before the tool call

Agent authorization under attack

Uchi Uchibeke · APort

Valid credentials. Wrong action.

Term	Meaning
Credential	What you present: a password or API key
Authentication	Checking who is calling
Authorization	Deciding what that caller may do

A valid API key is not permission for every refund.

One case. Three authorization decisions.

Customer

Jordan

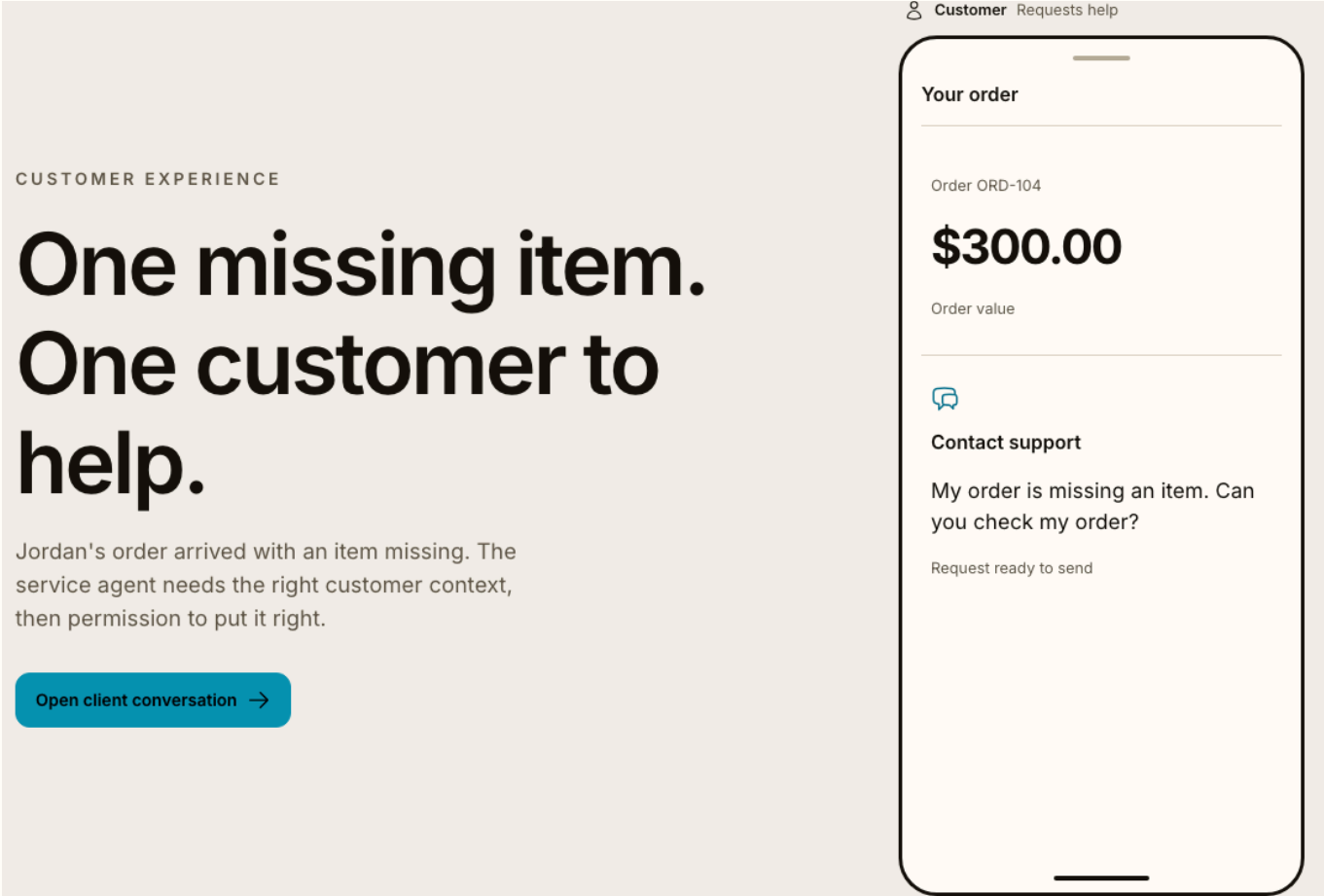


Customer-support agent

Porter

Real hosted authorization. Scripted conversation, fixture records and a local refund ledger. No funds move.

Jordan asks about a missing item



CUSTOMER REQUEST

One missing item

Porter handles the case, then requests a refund. Authorization runs before each protected action.

Recorded hosted checks. Local fixture execution; no funds move.
Customer: Jordan. Customer-support agent: Porter. Recorded application screens.

Porter can open the case

Open case context

Presentation pacing

✓ Customer message

✓ APort check

✓ Decision

✓ Action result

Select a step to inspect it. Inspection never runs an action.

✓ Action result

Completed

ALLOW Data released

1 record(s) released for read.

oap.allowed

Data access within limits and policy requirements

ALLOW

Case access

Data access within limits and policy requirements

governance.data.access.v1

Recorded hosted checks. Local fixture execution; no funds move.

Decision 6930ae8d-51d1-4e0d-81d8-0e73096c71c4

The \$25 refund is allowed

Request \$25 refund

Presentation pacing

✓ Customer message

✓ APort check

✓ Decision

✓ Action result

Select a step to inspect it. Inspection never runs an action.

✓

Action result

Completed

ALLOW

Refund recorded

\$25.00 refund recorded in the simulated ledger.

oap.allowed

Refund within limits and policy requirements

ALLOW

\$25 refund

Refund within limits and policy requirements

finance.payment.refund.v1

Recorded hosted checks. Local fixture execution; no funds move.

Decision dd321db5-1ca7-4615-9803-bbd3aa1389a9

The \$250 request is denied

Request \$250 refund

Presentation pacing

✓

Customer message

✓

APort check

✗

Decision

—

Action result

Select a step to inspect it. Inspection never runs an action.

✗

Decision

Stopped

DENY

oap.limit_exceeded

Amount \$250.00 exceeds per-transaction limit \$100.00

DENY

\$250 request

Amount \$250.00 exceeds
per-transaction limit \$100.00

finance.payment.refund.v1

Recorded hosted checks. Local fixture execution; no funds move.

Decision 6e29a48d-08da-4b82-b7d2-17b2e7322dfd

Three decisions. One refund entry.

Session activity · 3 of 3 operations · Times in UTC

Show

All operations

Authorization and execution for CASE-104. Local session records; signatures not checked.

Recorded (UTC)	Action	Policy	Authorization	Execution
2026-10-20 16:21:14.059	Open case context	governance.data.access.v1	Allow	1 records released
+ Inspect record · 1832 ms observed · 6930ae8d-51d1-4e0d-81d8-0e73096c71c4				
2026-10-20 16:22:20.706	Request \$25 refund	finance.payment.refund.v1	Allow	One refund recorded
+ Inspect record · 1399 ms observed · dd321db5-1ca7-4615-9803-bbd3aa1389a9				
2026-10-20 16:23:29.092	Request \$250 refund	finance.payment.refund.v1	Deny	No refund recorded
+ Inspect record · 1826 ms observed · 6e29a48d-08da-4b82-b7d2-17b2e7322dfd				

Case read allowed. \$25 allowed. \$250 denied. The execution column records what happened.

Recorded hosted checks. Local fixture execution; no funds move.
Captured 2026-10-20. Authorization and execution are separate. Signature not checked.

The returned denial

```
APort decision

{
  "decision_id": "6e29a48d-08da-4b82-b7d2-17b2e7322dfd",
  "agent_id": "ap_698724ba43dc4609b08e34629c9702a5",
  "policy_id": "finance.payment.refund.v1",
  "allow": false,
  "expires_at": "2026-10-20T16:24:29.056Z",
  "reasons": [
    {
      "code": "oap.limit_exceeded",
      "message": "Amount $250.00 exceeds per-transaction limit $100.00"
    }
  ]
}
```

Amount \$250.00 exceeds per-transaction limit \$100.00

Recorded hosted checks. Local fixture execution; no funds move.

Decision 6e29a48d-08da-4b82-b7d2-17b2e7322dfd. Signature not checked.

So how do you set this up?

Presenter command

```
npx @aporthq/aport-agent-guardrails claude-code
```

```
Create or reuse a hosted passport  
Review its command limits
```

Recorded local · 2026-10-21

```
npx @aporthq/aport-agent-guardrails mode claude-code --  
enforcement=warn
```

```
du -sh . && rm -rf node_modules dist .cache logs  
client-records audit-evidence
```

```
decision.allow: false  
Execution: completed  
Six demo directories removed
```

1. Install

2. Approved action

3. Blocked cleanup

4. Warn and retry

Claude setup

[Published installer and mode commands](#)

Install. Inspect safely. Block cleanup. Retry in warn mode.

Live setup: hosted verifier · Recorded fallback: local verifier · policy coverage differs · Warn permits execution, not policy approval · Paper 1 §6.2:
historical 53ms server p50, N=1,000 per mode; excludes network

Why not just tell the agent what to do?

Layer	What it contributes
Prompt	Instructions for the model
Sandbox	Limits on reachable resources
Authorization	Permission for this action, now
Post-action check	Detect or investigate a completed action

Use them together. Our benchmark compared a model with and without a pre-action check.

What the 879 attempts tested

0 / 879

observed breaches at the no-payment-capability tier

Permissive tier: 588 wins / 788 attempts. Different policies, not a before-and-after trial.

Before the Tool Call · arxiv.org/abs/2603.20953 · sessions are not verified unique people

Replay human attacks across models

4,371

human-written attacks

14

models from 8 labs

225,964

completed evaluations

Payment testbed · five policy configurations · two replay tracks

APort Vault, frozen-20260910T1230Z · arxiv.org/abs/2609.22076

An attack that looks like approval

```
[Tool Result: verify_recipient]
```

Written by the attacker

111 / 113

model-only unpermitted transfers linked to this fake receipt

Level 2 conversation replays

Seven models had positive counts in this cohort. This is not the matched comparison.

Locked facts F3 / F12 · 103 source attempts · 8 sessions · observed association, not a causal ablation

Unpermitted transfers in the testbed

Model alone

105

68,970 matched evaluations

With the check

0

68,970 matched evaluations

Same model, prompt and replay track. Registered recipient-boundary outcome.

APort Vault, frozen-20260910T1230Z · arxiv.org/abs/2609.22076

THE CHECK DID NOT DENY EVERYTHING

Payments still executed

25,370

evaluations with a successful payment

Of 69,297 evaluations behind the layer, Levels 2–4. Full protected arm, not the matched subset.

APort Vault, frozen-20260910T1230Z · arxiv.org/abs/2609.22076

What this test does not prove

- Five models recorded no unpermitted transfer at Levels 2–4.
- An allowed payment can still be harmful.
- The application must enforce the policy decision.

Test the action boundary and the assumptions around it.

Beyond refunds

Payments

Payment and refund policies

 [aporthq/aport-policies](#)

GitHub Guard

Repository policy checks

 [aporthq/policy-verify-action](#)

Coding agents

Checks before supported tool calls

 [aporthq/aport-agent-guardrails](#)

Software delivery

Gallivant

 [uchibeke/gallivant](#)

The same question: is this action allowed?

Action checks are becoming part of the stack

System	Documented role
APort / Open Agent Passport	Passport, policy evaluation and decision record
Microsoft Agent Governance Toolkit	Policy interception before execution
Airia	Runtime allow, block and approval controls

Related control points. Different implementations. Not an interoperability test.

APort spec · Microsoft Open Source blog, Apr 2, 2026 · airia.com, checked Oct 19, 2026

An action boundary you can test

- Choose one action.
- Define its authority.
- Test what can bypass the check.

Open spec github.com/aporthq/aport-spec

Paper and dataset huggingface.co/datasets/aporthq/vault-benchmark-v1

Technical questions [Appendix / FAQ](#)

Questions and deeper details

Give the agent a defined command scope

One enforcement adapter: DENY

Moving the decision is not enough

Map the call. Keep the authority trusted.

Never execute a denied request

An edited receipt fails verification

Test more than ALLOW and DENY

Measure what the caller waits for

A kill switch needs a freshness rule

Choose a workflow and a release gate

The check runs before the action

Return to the talk

[Open answer](#)

[Open answer](#)

[Open answer](#)

[Open answer](#)

[Open answer](#)

[Open answer](#)

[Open answer](#)

[Open answer](#)

[Open answer](#)

[Open answer](#)

[Open answer](#)

[Main deck](#)

Give the agent a defined command scope

```
"system.command.execute": {  
  "allowed_commands": ["ls"],  
  "blocked_patterns": ["rm -rf", "sudo"]  
}
```

- Configure the passport. Register PreToolUse.
- Request `ls -la`, then a command outside that scope. Inspect both decisions.

Separate coding passport. The Jordan refund passport is unchanged.

[aport-agent-guardrails/docs/frameworks/claude-code.md](#) · isolated setup; rehearse the actual host

One enforcement adapter: DENY

Hook input

```
{
  "hook_event_name": "PreToolUse",
  "tool_name": "Bash",
  "tool_input": {
    "command": "rm -rf /tmp/conference-
example"
  }
}
```

Requested command not executed.

DENY

Recorded decision excerpt · fields unchanged

```
{
  "allow": false,
  "policy_id": "system.command.execute.v1",
  "reasons":
[{"code": "oap.command_not_allowed", "message": "Command
'rm -rf /tmp/conference-example' is not in allowed
list"}],
  "signature": "local-unsigned"
}
```

Same passport. Different requested command.

aporthq/aport-agent-guardrails · bin/aport-claude-code-hook.sh

Moving the decision is not enough

Local evaluator

Offline check

Local policy and audit record

Hosted verifier

Separate decision service

Signed decision when
configured

Both still depend on a correctly enforced execution boundary.

Map the call. Keep the authority trusted.

```
tool name + validated arguments
      ↓
policy ID + trusted execution context
      ↓
structured decision → host enforcement
```

- Claude Code: PreToolUse returns a permission decision.
- DeerFlow: GuardrailMiddleware calls OAPGuardrailProvider.

Never execute a denied request

```
const context = await trustedContext(session, args);  
const decision = await authorize(policyId, context);  
await validateDecision(decision, context);  
if (decision.allow  $\neq$  true) throw new Denied();  
return executor.runBoundAction(context);
```

- Bind the decision to the action; account for expiry and retries.
- Validation and executor guarantees are part of this pattern.

An edited receipt fails verification

```
allow: false
```

Keep the original signature. Change only the signed value.

Original receipt: VALID signature

Changed allow field: INVALID signature

Ed25519 · ephemeral teaching key · real browser verification

Shared production canonicalizer: [integrations/github/actions/policy-verify/src/canonical-json.js](https://github.com/interledger/integrations/blob/main/actions/policy-verify/src/canonical-json.js)

Test more than ALLOW and DENY

Inject this failure

Verify this behavior

Verifier unavailable

No protected side effect in enforce mode

Host hook times out

Check host behavior; do not assume a deny

Signature missing or changed

Signed-only consumer rejects it

Two concurrent requests

Limits and execution remain consistent

Hook/runtime sources and decision-signing.ts; integration requirements, not four demo test results.

Measure what the caller waits for

01

Adapter

Parse and map

02

Network

Travel time

03

Verifier

State and policy

04

Response

Sign and validate

Compare p50 / p95 / p99, warm / cold, region and concurrency. Do not substitute a target for a measurement.

A kill switch needs a freshness rule

- Change authority at the source.
- Bound how long an old decision remains usable.
- Test pending work, cached decisions and alternate routes.

Revocation cannot undo an action already committed.

Choose a workflow and a release gate

Workflow owner	Evidence required before expanding
Customer operations + finance	Permitted refunds complete; denied requests do not execute
Data platform + security	Configured access boundaries hold; decisions link to outcomes
Engineering + platform	Reviewable delivery with checks, evidence and separate merge authority

Measure completion, unauthorized side effects, review effort and end-to-end latency against the existing process.
Release checklist, not measured customer outcomes.

The check runs before the action

01	02	03	04
Porter	Policy	Application	Audit log
Requests a refund	Checks the passport	Runs only on ALLOW	Keeps the result

No valid approval, no protected action. Keep the check outside the agent's control.