Authority Continuity for Verifiable Lineage in Distributed Systems and AI Agents

Nicola Gallo, Stefano Silvestri, Antonio Radesca

Nitro Agility S.r.l. · ICAR, National Research Council of Italy (CNR)

AUTHORITY CONTINUITY · PIC (Provenance Identity Continuity)

Authority can continue only if the executor can prove a causal continuation, in the temporal dimension of the authorization model.

Without solving this problem, autonomous AI agents are unlikely to be truly ready for production.

A warning: this talk is provocative.

On purpose: it presents a concept that **puts everything back into question.**

Without provocation,
it would not get through.

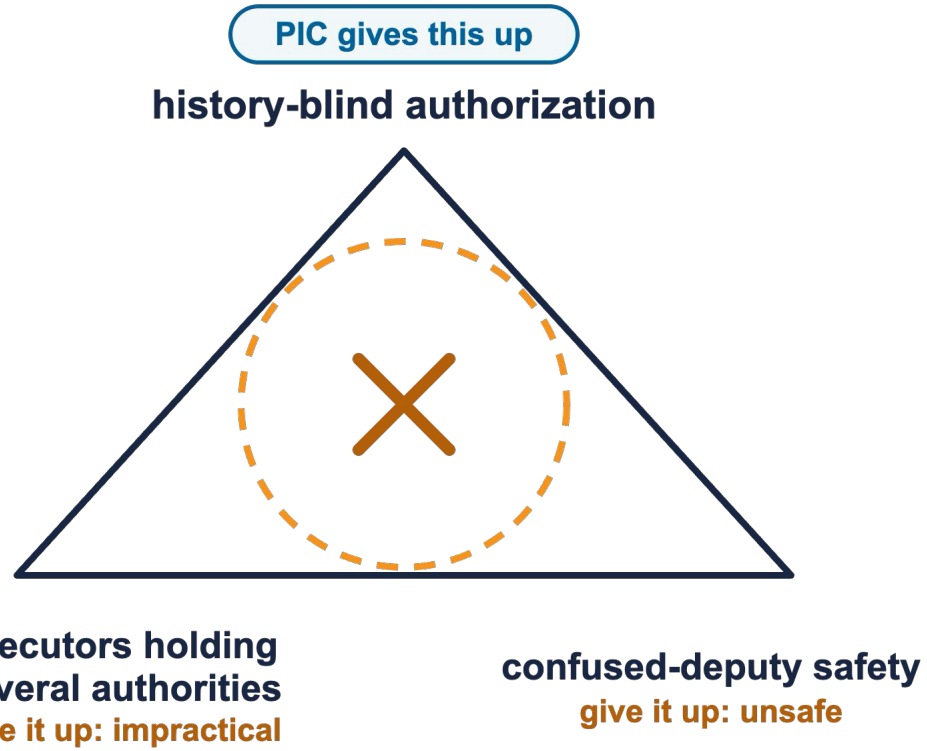
It comes from industrial experience: building distributed systems in production, at scale, and seeing where they break.

And a challenge to academia: most theory treats authority as possession.
It is time to study it through the PIC model as a question of authority continuity: causal time embedded in the protocol itself, not merely treated as metadata.



Our starting point: the trade-off theorem

Theorem 5 *(Possession, delegation, safety trade-off)* · the three cannot hold together



PIC was built to eliminate the Confused Deputy problem of 1988 over causal, temporal executions: mathematically, within the model.

Capabilities
The single hop, by discipline: a confused deputy is still expressible as a logic error in the deputy.

PIC
A formal exclusion theorem over time, even when the executor errs. To our knowledge, the first.

N. Hardy, "The Confused Deputy", ACM SIGOPS OSR 22(4), 1988 · Theorems 5 and 6, under the paper's stated assumptions.

How does most of today's security work?

Authority is something you hold:

a key, a token, a credential.



Possession-based authorization

Many authorization mechanisms rely on possession-based evidence: authority is bound to an artifact. **If you hold the key, the door opens.**



Key: the door opens

No key: it stays closed

$$Possess(h, k) \wedge Grants(k, o, r) \Rightarrow Authorized(h, o, r)$$

h: the holder · *k*: the key, a token, credential or capability · *o*: the operation · *r*: the resource

The receiver checks the key, the operation, the resource. **Not the execution that brought the request.**

FOR EXAMPLE

Bearer tokens, as in OAuth 2.0



Who holds it, uses it.

Capabilities



Designation and authority travel together.

⚠ **Possession alone cannot establish whether a valid artifact is being used within the intended causal execution.**

Today we live hoping nobody steals our keys.

And if a key is lost or stolen? We hope nobody uses it, **or that we revoke it in time.**

The paradox:
formal proofs for the cryptography,
hope for whoever holds the key.



But is hope security?

Or just hoping that, maybe, it works?

And when hope runs out, we revoke the key

Lost, stolen, or suspected in the wrong hands: the common response is to **revoke the key (or the credentials), or let it expire.**



What revocation does

It closes the door for the future: the stolen key stops working.

What it does not

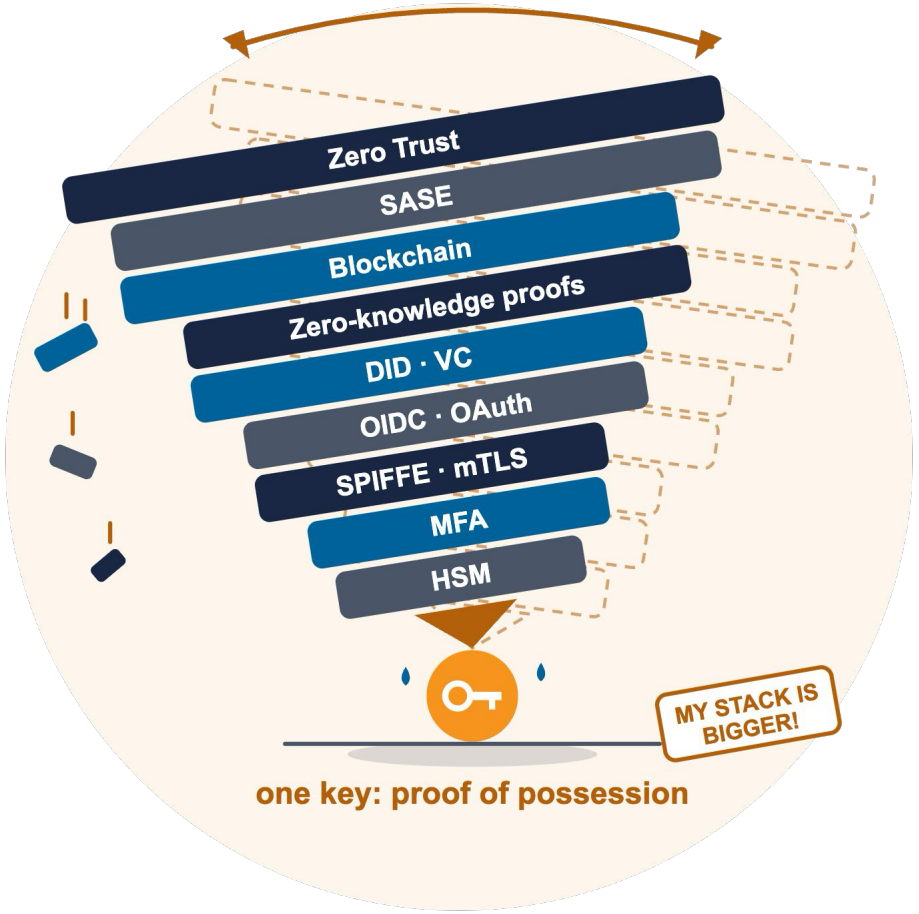
It cannot undo the window: the damage is mitigated, not prevented.

Revocation addresses compromised credentials, not authority mixing by a legitimate holder.

When that is not enough, we mitigate.

Layer upon layer, a contest for the biggest stack: **Zero Trust, blockchain, zero-knowledge, DID, VC…**

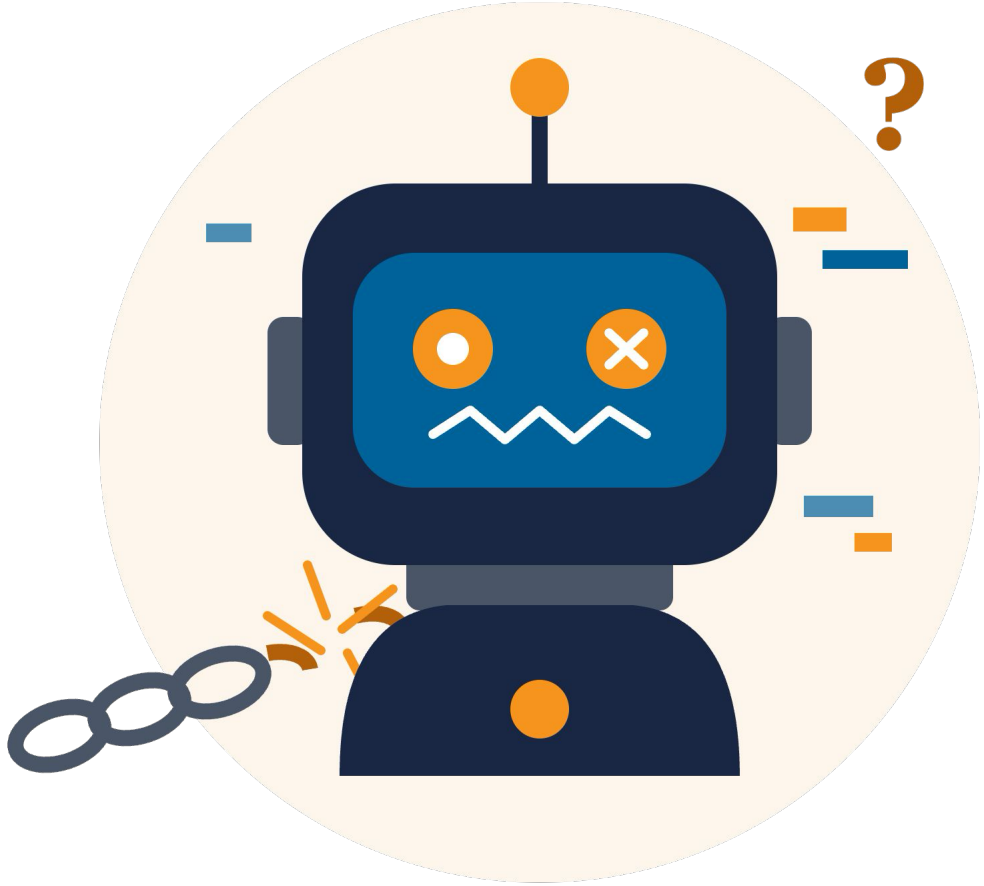
The whole pyramid balances on one point:
the hope that nobody steals the key that carries our authority.



Like every good story, the spell ends.

We cannot live on hope with an AI agent: **that is where everything breaks.**

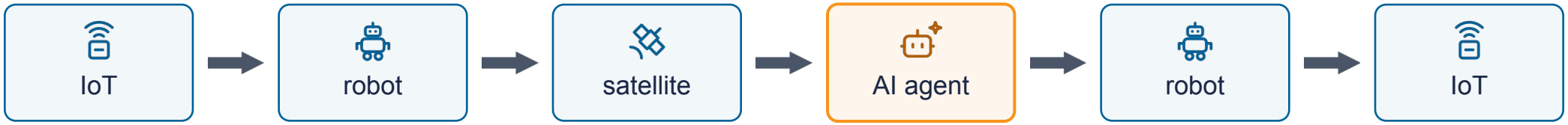
How do I trust an AI agent I cannot tame, one that can hallucinate, or attack me?



The same problem, in every multi-hop execution

Enterprise cloud	Government	Satellites	IoT	Robots	AI agents
services, queues, functions	a private office linked to a public system	in a hostile environment	devices in the field	physical machines moving among people	intelligent agents acting on our behalf

The future: machines calling machines, with no human in between

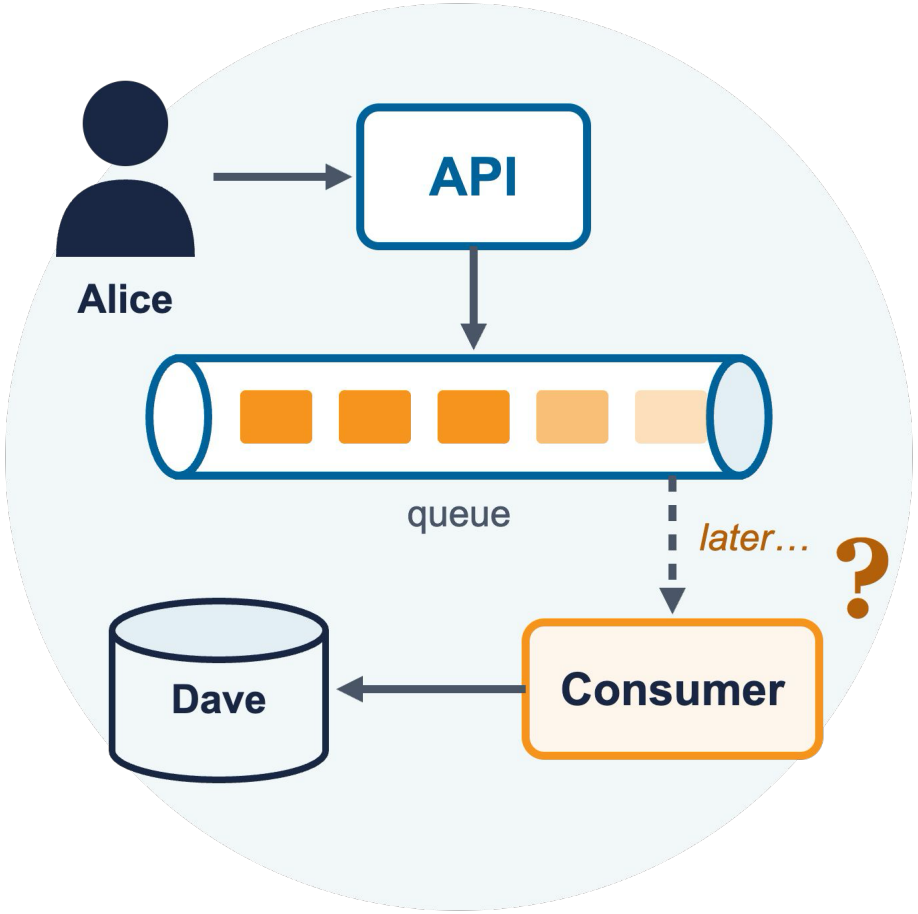


The problem becomes clear when we compose different executors. AI agents make it visible: chains of tool calls, many authority sources held at once.

An example?

Nothing exotic: **an API, a queue, a consumer.**

Business as usual,
for any enterprise.



A common use case, in any enterprise



How can a future consumer establish that its request continues the original execution, without relying on a pre-bound successor identity?

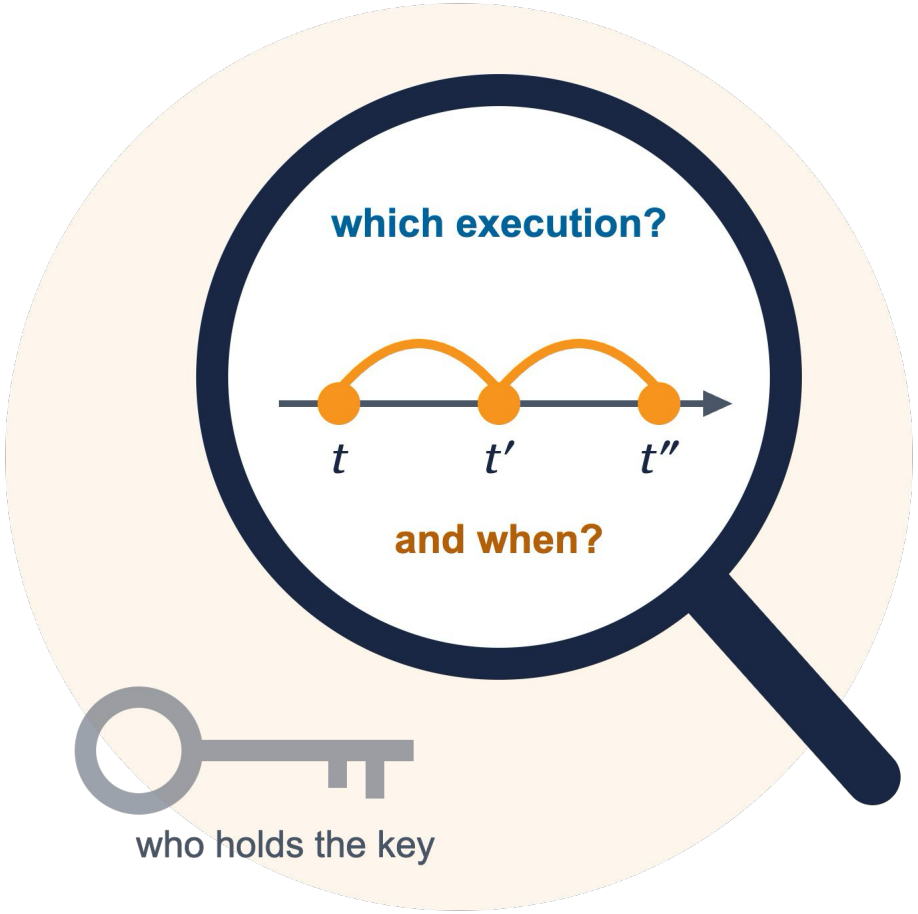
A bearer token in the log
Reusable by anyone who can read the log.

A key-bound token
Proves control of a key, but does not by itself prove continuity with the original request.

Either way
The artifact says who may present it, not which execution it continues.

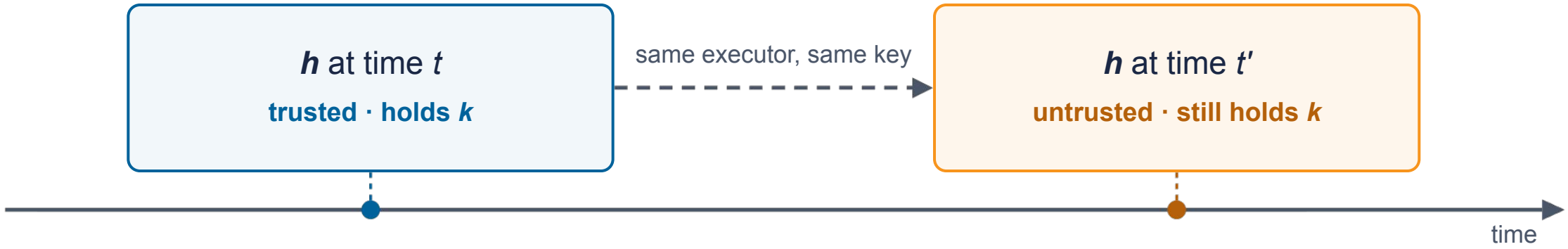
What if we look at the problem from a different perspective?

Not only who holds the key, but **within which execution, and when.**



Today's threat model

To understand the ontological shift of PIC, look at the **temporal dimension of authority**.



$Possess(h, k) \wedge Grants(k, o, r) \Rightarrow Authorized(h, o, r)$ holds at t and at t' alike.

The credentials pass through *h* in both cases: so *h* must be treated as untrusted.

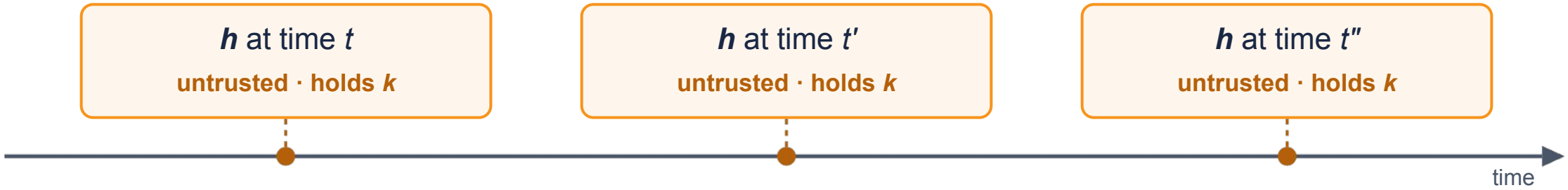
We don’t like the easy life.

Securing what we trust is easy. **The serious stuff is where we trust no one.**

So we question everything:
a new foundation,
on an untrusted threat model.



Change the threat model: what if *h* is untrusted at every moment?



On possession alone, only two options

Give *h* the key
Authorized(h, o, r), whatever *h* decides to do.

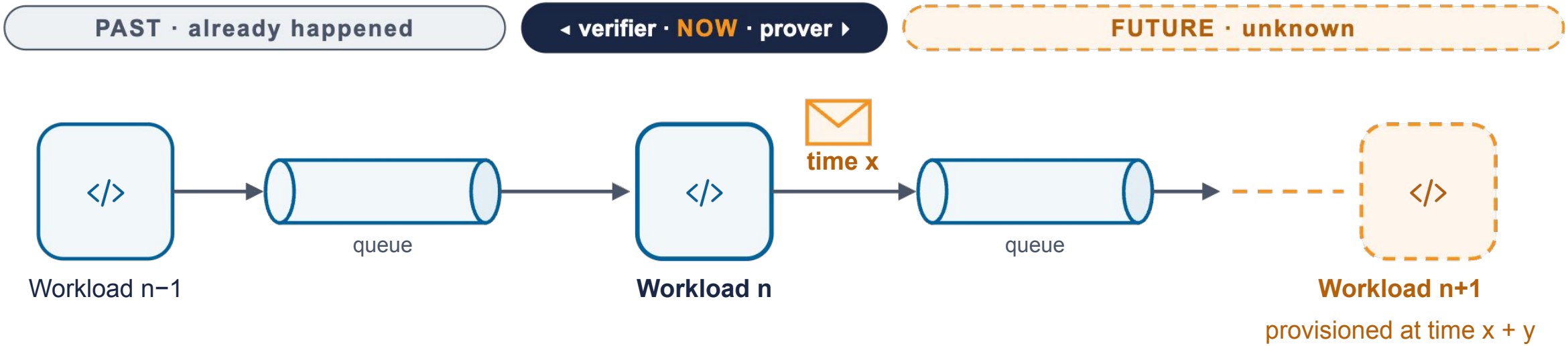
Withhold the key
h cannot be used at all.

How do you trust a holder you do not trust, do not control, and whose configuration you cannot see?

An AI agent that can hallucinate, or decide to attack you, is exactly this holder.

PIC introduces the Canonical Execution Model

One model of distributed execution, under an **untrusted threat model**: executors may be buggy, confused or adversarial.



Execution is temporal and causal, not positional.

Workload n+1 is provisioned at time **x + y**, with y > 0.

Authority flows from the origin: it is **never re-created**.

Every valid hop must prove **continuity**: possession alone is not enough.

Three problems at step N+1

Execution is temporal: what happens at N+1 **cannot be fixed at N**.

PROBLEM 1

Unknown Executor

How can I delegate to the unknown future?

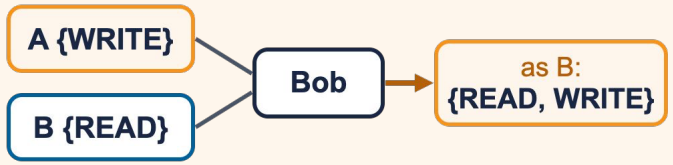


Carol may not exist yet: authority cannot be pre-bound to her key. **She must prove her step continues Bob's.**

PROBLEM 2

Invalid State

Authority mixing, with no theft.

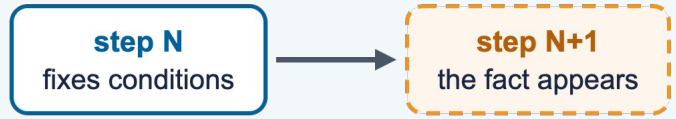


$C' \not\sqsubseteq CB$: every permission is authentic, **the composition is not a valid continuation.**

PROBLEM 3

Unknown Facts

The decision depends on a fact that does not exist yet.



It must be verified at N+1, with evidence, **within the original mandate.**

Related work of the PIC project; the paper covers 1 and 2.

All three ask the same question: does this step continue the execution?

Now get ready to change how you think about security.

A new ontology of authority: **not what you hold, but what you continue.**

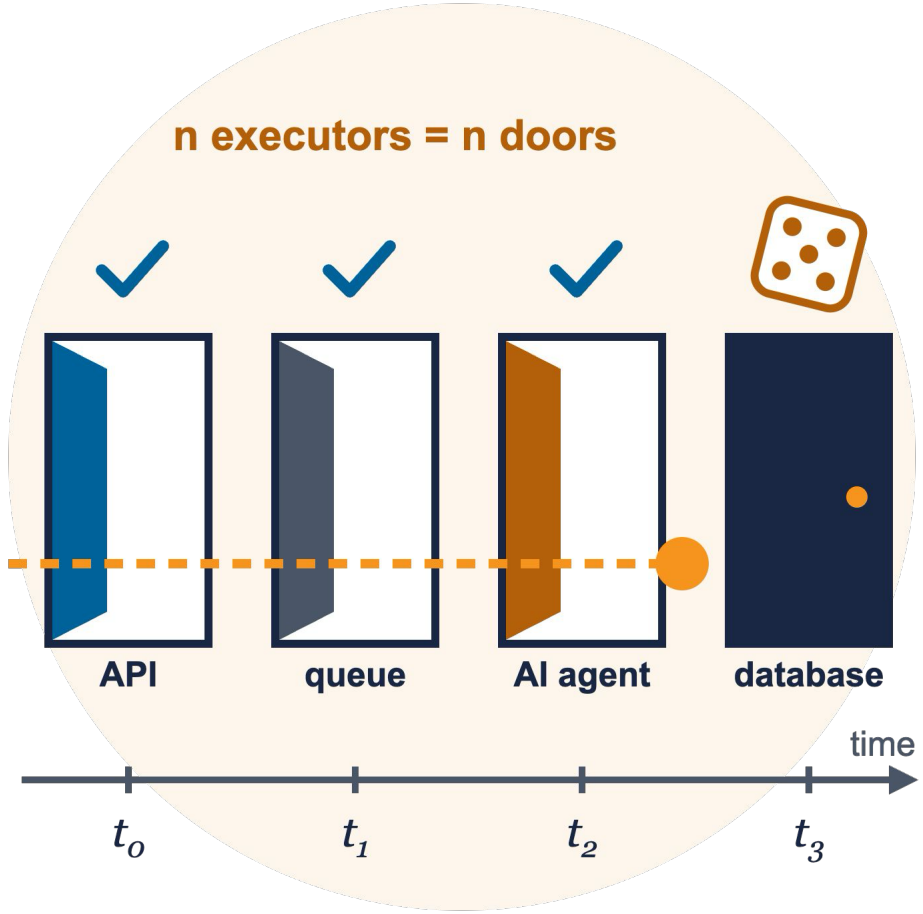


An execution has to pass through many doors over time.

> **Every executor is a door of the execution:** through n executors, it crosses n doors.

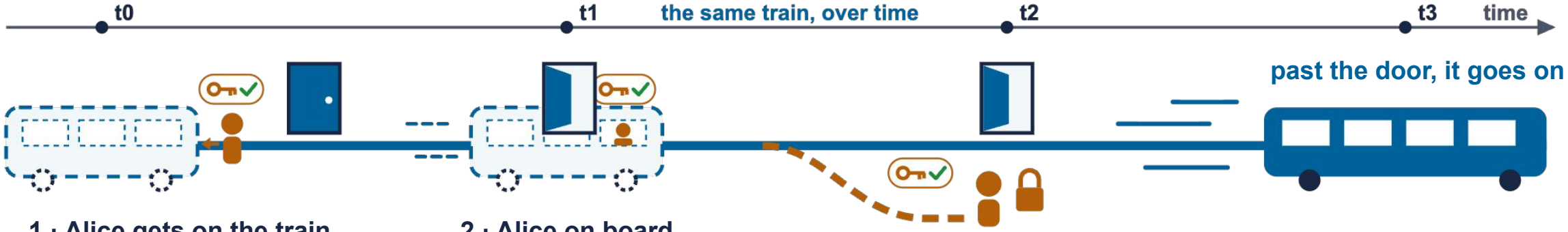
Different doors, different owners, different times: **how do I prove I may pass them all?**

Hoping nothing goes wrong at every door is a gamble, not security.



PIC: the train metaphor

The door opens to whoever **is part of the journey**, not to whoever holds a key.



The door, with possession
A valid key may remain usable across unrelated executions, unless authorization also checks the relevant context.

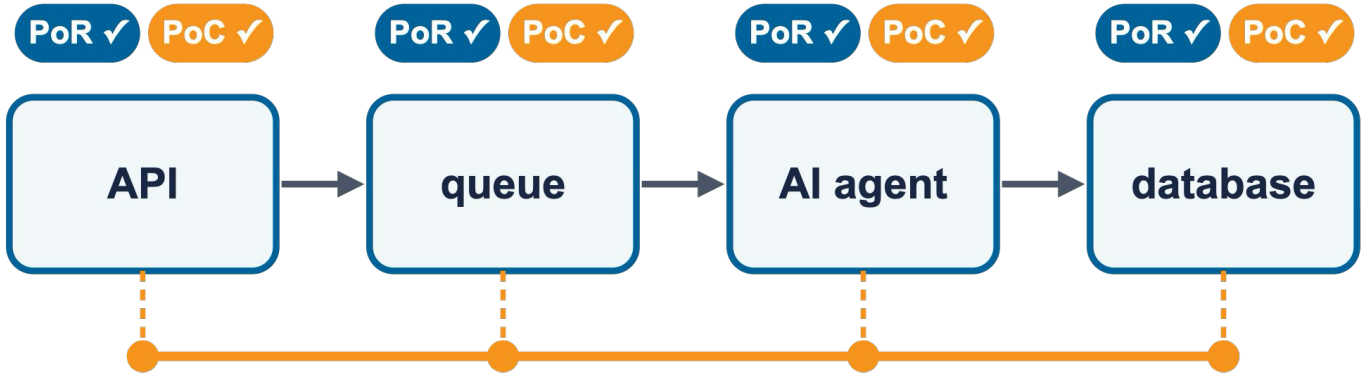
The train, with continuity
Nothing is revoked: the same key is insufficient outside a valid causal continuation.

The train is the execution.

The new architecture: simple by construction

At each node, the step proves **where it comes from** and that **its authority may continue**.

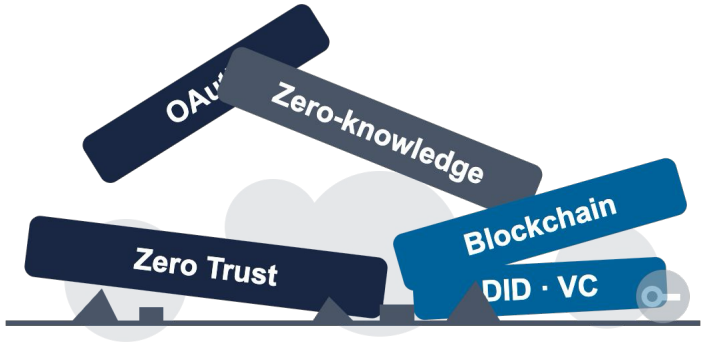
WITH PIC



one causal lineage, from the origin

PoR: the step proves it continues the one before
PoC: PoR at every hop, and authority never grows

THE STACK OF MITIGATIONS



it rested just on possession

Each step proves PoR and PoC: possession is no longer the foundation.

To solve these problems, PIC tells us…

These problems cannot be solved by possession alone.

PIC changes the ontology of authority, with two new primitives.

THE NEW FOUNDATIONS in the PIC model, in place of **Proof of Possession**

Proof of Relationship

Each step proves it continues the one before it.

Proof of Continuity

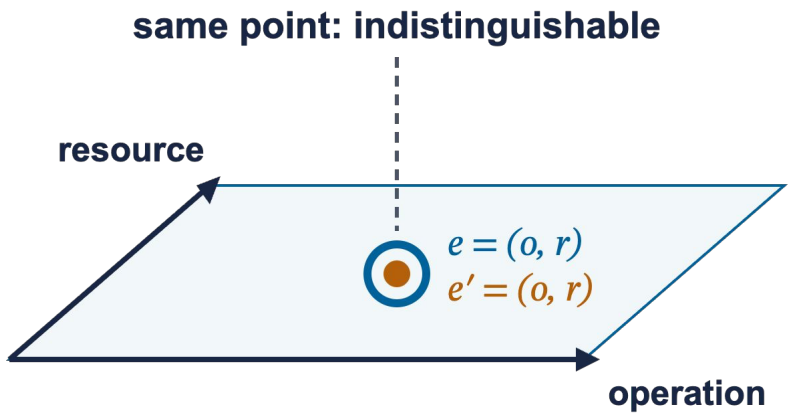
From the origin to now, every step is related, and authority never grows.

Authority continuity through execution,
where an execution is a causal, temporal order.

Essentially, **PIC adds the temporal dimension to authorization.**

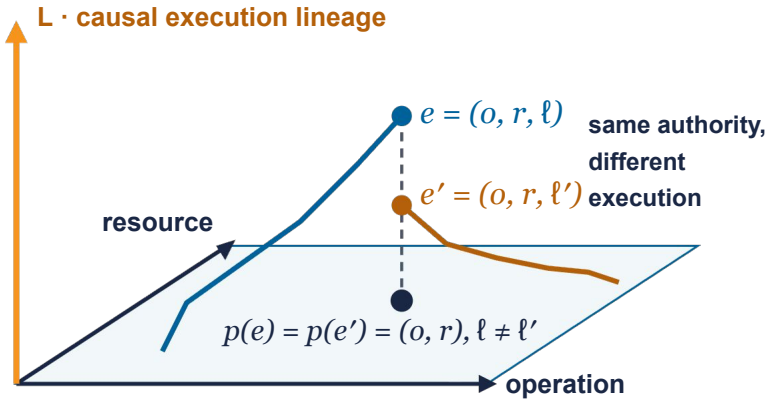
Adding Causal Time to Authorization

Possession-based view · O × R · spatial



+ L

Continuity · O × R × L · temporal: does it belong to this execution?



Definitions 1, 5, 6 *(PoR, valid transition, PoC)* · relationship is local, continuity is global

$$PoC(\pi) \Leftrightarrow \bigwedge_{i=0}^{n-1} (PoR(s_i, s_{i+1}) \wedge C_{i+1} \subseteq C_i)$$

Theorems 3 and 4 *(projection)*

Possession is the projection p : O × R × L → O × R that forgets the lineage: a lineage-invariant policy gives (o, r, ℓ) and (o, r, ℓ′) the same verdict. Distinguishing otherwise identical authority uses across lineages requires lineage-sensitive information.

With PIC, the Confused Deputy and the Temporal Confused Deputy are mathematically excluded within the model

1988 · the Confused Deputy

Using your authority on a **resource** designated by someone else.

In PIC · the Temporal Confused Deputy

Using your authority on an **execution** designated by someone else.

Now take an **untrusted executor** that makes **one logic error**: it picks an authority from outside the request.

WITHOUT PIC · THE HOLE

Possession · O × R: is this operation on this resource authorized by the presented authority? **Yes**: the authority is valid.
Same possession properties, different causal origins: **the receiver cannot tell them apart.**

The confused deputy is admissible.

Theorem 2 (possession admits authority mixing) · Theorem 3 (it cannot tell them apart) · Theorem 5 (the trade-off)

WITH PIC · SAME EXECUTOR, SAME ERROR

Continuity · O × R × L: does this authority belong to the specific execution being continued? **No**:

$$Ck \subseteq \ldots \subseteq Co,\ (o, r) \notin Co \Rightarrow (o, r) \notin Ck$$

Origin-bounded (Theorems 1, 6) · **lineage-bound** (Definition 1, PoR): a conforming receiver rejects it.

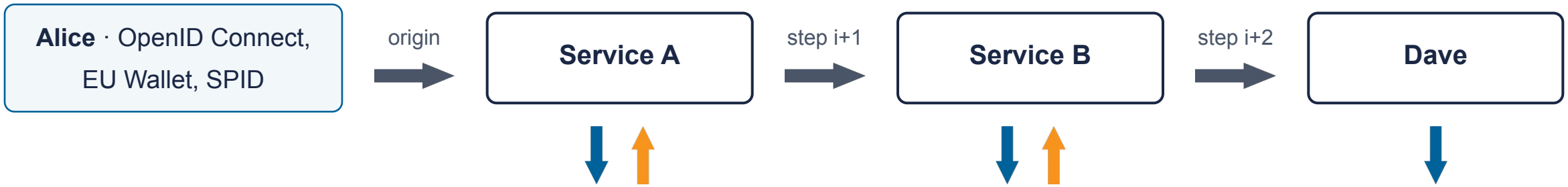
Classical and temporal: excluded by theorem.

❙ Proved in the paper · main results machine-checked in Lean 4

PIC introduces causal execution lineage as a first-class authorization dimension.

N. Hardy, “The Confused Deputy”, ACM SIGOPS OSR 22(4), 1988 · Within the model, under the paper’s stated assumptions.

Beyond the theorems: PIC-X over OAuth 2.0 Token Exchange



Trust Plane · OAuth 2.0 Token Exchange (RFC 8693)

In: subject_token, the candidate the workload signed over the previous checkpoint, with the attester's SD-JWT.

Checks: the step is linked to its exact predecessor checkpoint, and the authority does not grow.

Out: the token for the next step, attenuated. Dave verifies it.

PIC-X v0.2.3

open-source realm over RFC 8693, measured end to end

245 µs
per hop: workload proves, realm settles

≈ 50,000/s
settlements on 16 cores

16 ms
a 64-hop lineage, constant cost per hop

1.5 KB
token, the same at hop 1 and hop 64

Architecture sketch: PIC specifies invariants, not a fixed architecture (§IV). PIC-X v0.2.3 cryptographically links each step to its exact predecessor checkpoint; it does not yet prove an observed handoff. Our measurement, not the paper's: in process, one machine, *benches/token_exchange.rs* · github.com/pic-protocol/pic-x

Thank you

Questions?

Authority Continuity for Verifiable Lineage in Distributed Systems and AI Agents

Nicola Gallo, Stefano Silvestri, Antonio Radesca

Nitro Agility S.r.l. · ICAR, National Research Council of Italy (CNR)

Code · PIC-X
github.com/pic-protocol/pic-x

Protocol
pic-protocol.org

Proofs · Lean 4
github.com/ngallo/pic-model

The research continues · follow-up papers on arXiv

Beyond Proof of Possession: A Relational Theory of Authority
arxiv.org/abs/2611.07159

Full Provenance on the COSS Causal Axis: An Event-Level Closure of Causal Attribution and Confused-Deputy Exclusion
arxiv.org/abs/2611.07164