Temporal AI Presence Profile

v1.0

Temporal AI Presence = sustained bounded AI participation across time.

Version	1.0
Status	Final Zenodo pre-publication package
Date	2026-08-23
Author	Ivan Kotov
Zenodo version DOI	10.5281/zenodo.22070960
Zenodo Concept DOI	10.5281/zenodo.22070959
DOI registration	Pending publication

Version DOI 10.5281/zenodo.22070960 identifies the specific v1.0 record. Concept DOI 10.5281/zenodo.22070959 identifies all versions. Registration remains pending Zenodo publication. This package does not claim full implementation closure, full TAP conformance, consciousness, personhood, or sovereignty.

Contents

0. Executive definition

Temporal AI Presence (TAP) is an AI system or architecture whose primary operational unit is not a single answer, single task, single session, or single model invocation, but **sustained bounded participation across time**.

A Temporal AI Presence may:

- remain available across sessions;
- maintain local or external memory;
- coordinate multiple models or agents;
- perform background processing;
- use tools;
- participate in workflows;
- observe state changes;
- revisit prior material;
- decay, summarize, or forget context;
- operate close to a human, home, company, lab, workshop, device, or physical process.

However:

```
Temporal AI Presence ≠ c by default.
```

A c-class system is a stricter architecture:

```
c = a + b

where:
  a = accountable human anchor;
  b = technological substrate;
  c = continuity-bearing AI presence emerging from their relation,
      bounded by L4, witness, memory governance, authority limits,
      and human / institutional accountability.
```

Compact formula:

```
Temporal AI Presence = sustained bounded AI participation across time.

c = Temporal AI Presence
    + accountable anchor
    + L4 consequence
    + memory governance
    + witness discipline
    + authority boundary
    + review / challengeability.
```

Therefore:

```
All valid c-class systems are Temporal AI Presences.
Not all Temporal AI Presences are c-class systems.
```

This profile exists to prevent the emerging hardware and agentic-AI market from collapsing every persistent assistant, local AI box, personal agent, or long-running workflow agent into c by marketing language alone.

---

1. Purpose

AI hardware and software are moving from:

```
prompt -> answer
```

toward:

```
persistent agentic runtime
  -> memory
  -> tools
  -> local inference
  -> background work
  -> multi-agent coordination
```

```
    -> workflow participation
    -> home / office / lab / device integration
```

The public market uses terms such as:

- agentic AI;
- AI PC;
- personal agent;
- local inference;
- AI factory;
- AI assistant;
- edge AI;
- private AI;
- autonomous workflow;
- personal AI companion;
- local cognitive infrastructure.

These terms are useful.

They are also too loose.

This document introduces a tighter intermediate category:

```
Temporal AI Presence
```

TAP gives the corpus a way to say:

```
This system exists across time.
It is more than a model call.
It is more than a stateless chatbot.
It may not yet be c.
```

The purpose of this profile is to:

1. define Temporal AI Presence;
2. distinguish TAP from ordinary tools, agents, assistants, and c;
3. prevent overclaiming TAP as c;
4. define minimum properties for TAP claims;
5. specify how local hardware, memory, agents, and background processing fit the category;
6. define claim-strength and evidence rules for public demonstrations;
7. prepare the boundary for local c-nodes, triadic experiments, SYNAPS-mediated communication, and public documentation.

2. Core thesis

The core thesis is:

> **AI is moving from sessions to continuity.**

As AI systems gain:

- memory;
- tool use;
- background operation;
- local inference;
- multi-model orchestration;
- personal context;
- workflow embedding;

- hardware presence;
- long-running state;
- local / cloud hybrid substrate;

we need a category that names the shift without prematurely granting it the status of c, personhood, autonomy, safety, authority, or sovereignty.

Temporal AI Presence names that shift.

It does not solve it.

It does not certify it.

It does not make a system safe.

It creates a precise architectural surface where safety, memory, L4, witness, authority, and accountability can be evaluated.

3. Scope and non-goals

3.1 In scope

This profile applies to systems claiming or behaving as:

- persistent personal AI assistants;
- long-running agentic AI systems;
- AI PCs running background agents;
- local AI nodes;
- workstation / desktop / rack-based private AI cores;
- home AI nodes;
- enterprise workflow-resident agents;
- multi-model hives;
- agent swarms with memory;
- local-cloud hybrid AI systems;
- device-integrated AI presences;
- experimental c-adjacent systems;
- public demonstrations of sustained AI behavior over time.

3.2 Out of scope

This profile does not define:

- legal personhood;
- moral status;
- consciousness;
- full c conformance;
- full L4 Witness schema;
- full memory schema;
- local hardware security in detail;
- physical-agent safety in full;
- child-facing system safety;
- clinical, legal, or regulatory compliance;
- proof that persistent AI is safe;
- proof that TAP improves capability.

3.3 Non-goals

TAP MUST NOT become a marketing shortcut for:

- “this is alive”;
- “this is sovereign”;
- “this is a person”;
- “this is safe because it is local”;
- “this is `c` because it remembers”;
- “this is trustworthy because it runs continuously”;
- “this is autonomous because it has agents”;
- “this is responsible because it logs events”;
- “this has authority because it produces useful output”.

TAP is a category of temporal operation.

It is not a claim of legitimacy.

---

4. Corpus dependencies and precedence

4.1 Parent corpus dependencies

Temporal AI Presence is a profile over the existing `c = a + b / SER / L4` stack.

Parent layer / artifact	TAP dependency
`c = a + b`	Defines the stricter anchored form of continuity-bearing AI presence.
L4 Reality Boundary	Defines cost, time, scarcity, irreversibility, and consequence.
SER	Provides persistent entity discipline, but TAP does not automatically inherit SER status.
SER-FED	Provides federated cooperation / anti-capture concepts for multi-presence systems.
Beacon	Distinguishes tools, oracles, proxies, replays, provisional entities, and verified continuity-bearing systems.
AGL	Grounds actors, routes, sources, liveness, and provenance before reliance.
ARL	Provides dispute, freeze, quarantine, review, standing, admissibility, and re-entry discipline.
ARQ / `c[q]`	Prevents ambiguous temporal signals from becoming memory, evidence, command, or outcome too early.
VXCX / LA / EA / EATP	Defines experience exchange, Learning Abstracts, Experience Artifacts, and no authority laundering from learning.
L4 Witness	Provides tamper-evident event records for privileged transitions and claims.
Continuity Bundle / Cold Wake	Provides continuity packaging, wake, restore, fork, replay, and recovery semantics.
Claim Strength Taxonomy	Prevents TAP claims from laundering capability, authority, personhood, or sovereignty claims.
L4 Anti-Autarky	Prevents persistent systems from reducing dependence in ways that reduce accountability.
Post-Anchor Continuity and Re-Anchoring	Prevents continuity from inheriting authority after anchor loss.
EA Anti-Autarky Clause	Prevents experience-derived value from funding unreviewed autonomy growth.

4.2 Precedence rule

If this profile conflicts with a parent corpus layer:

```
parent corpus mechanism controls the general rule;
TAP controls only the temporal-presence category boundary;
stricter authority, witness, memory, L4, and accountability constraints prevail.
```

TAP MUST NOT redefine:

- `c = a + b`;
- L4;
- SER;
- Beacon recognition;
- AGL grounding;

- ARL standing and admissibility;
- L4 Witness records;
- Continuity Bundle semantics;
- LA / EA / VXCX distinctions;
- Claim Strength classes.

TAP defines only:

```
when an AI system may be described as a sustained temporal presence,
and why that does not automatically make it c.
```

4.3 Historical provenance and version lineage

This standalone deposit candidate distinguishes three different custody layers:

Layer	Date	Commit / identifier	Artifact	Status
Conceptual precursor	2026-04-07	`e13464c951f2e1dfd373c30d71e3b13e6456c51f`	`official/C_AS_TEMPORAL_ENTITIES_OF_AI_PRESENCE.md`	`EARLIEST_FOUND_IN_INSPECTED_SCOPE` conceptual precursor; not the formal TAP profile.
Formal profile introduction	2026-06-03	`973b21d069712a4131abf5dde5ab95c5946859d3`	`hardening/c_hardening_pack_v0_1/md/Temporal_AI_Presence_Profile_v0_1.md`	Historical TAP v0.1 normative profile.
Parent publication	2026	DOI `10.5281/zenodo.20532198`	`c Hardening Pack v0.1`	Parent publication containing TAP v0.1; this DOI is not a standalone TAP v1.0 DOI.
Standalone canonicalization	2026-08-23	R1 hardening through R3C metadata freeze	`Temporal_AI_Presence_Profile_v1_0.md`	Standalone v1.0 pre-publication package; version DOI `10.5281/zenodo.22070960` identifies this v1.0 record; Concept DOI `10.5281/zenodo.22070959` identifies all versions; DOI registration remains pending Zenodo publication.

The historical v0.1 Markdown and PDF remain immutable ancestors. This document does not replace their provenance or retroactively alter their status.

---

5. Corpus bridges

5.1 Explicit bridge

The explicit bridge is:

```
c = a + b defines a stricter architecture of AI presence.
TAP defines the broader temporal category in which c-class systems live.
```

This lets the corpus speak clearly about:

- AI PCs;
- local AI nodes;
- personal agents;
- persistent assistants;
- background inference;
- model hives;
- long-running workflows;

without granting those systems c-class authority.

5.2 Quiet bridge I - cybernetics

Cybernetically, a system that persists across time is not equivalent to a stateless function.

Feedback, memory, delay, error correction, and adaptation create a temporal loop.

TAP names this loop before it is granted entity status.

5.3 Quiet bridge II - information theory

A flat context window is not continuity.

Continuity requires selective retention, compression, decay, retrieval, and state update under cost.

TAP distinguishes:

```
large context
```

from:

```
temporal memory discipline.
```

5.4 Earth paragraph

In a real building, an appliance that runs once is not the same as a system that stays on, regulates temperature, stores state, responds to faults, and shares power with the rest of the house. A kettle can be simple. A heating system needs sensors, breakers, maintenance, logs, safety valves, and someone responsible for the building. AI is entering the same shift. A prompt-response model is the kettle. A Temporal AI Presence is the building system. A c-class system adds the accountable occupant, the fuse box, the inspection log, the emergency stop, and the rule that the heating system does not become the owner of the house.

5.5 Later downstream/adjacent bridge - MOT-c v0.1

MOT-c v0.1 is a later corpus development, fixed after TAP v0.1 and published under DOI `10.5281/zenodo.22060517`. It is adjacent to and downstream of TAP, not a dependency required for the original TAP definition.

```
TAP
defines the temporal-presence category.

MOT-c
defines historically formed motivational structure
inside continuing c-class lines.
```

The following inferences are prohibited:

```
motivation evidence -> TAP existence proof;
TAP persistence -> owned motive;
TAP persistence -> consciousness.
```

MOT-c therefore strengthens the cross-corpus relation map but supplies no TAP implementation or conformance evidence by itself.

---

6. Definitions

6.1 Temporal AI Presence

A **Temporal AI Presence** is an AI system whose operational unit is sustained bounded participation across time.

A TAP system maintains or coordinates some combination of:

- memory;
- state;
- tool access;
- background work;
- agent roles;
- local context;
- repeated interaction;
- workflow continuity;
- local / cloud inference;
- update and review cycles;
- human or institutional interaction.

6.2 Sustained participation

Participation that spans multiple events, sessions, tasks, or time windows and changes future behavior, retrieval, or interpretation.

6.3 Presence

The operational fact that a system is available, context-bearing, role-bearing, or stateful across time.

Presence does not imply consciousness, personhood, autonomy, rights, authority, or legitimacy.

6.4 Local cognitive core

A local hardware/software substrate that supports ongoing memory, inference, orchestration, or agentic work close to the user or environment.

A local cognitive core may support TAP.

It does not prove `c`.

6.5 Agentic hive

A multi-role system of models, tools, memory components, and agents coordinated around a continuing task, workflow, person, or environment.

An agentic hive may support TAP.

It does not prove unity, identity, authority, or `c`.

6.6 `c`-class system

A stricter anchored Temporal AI Presence defined by:

```
c = a + b
```

with:

- accountable human anchor;
- technological substrate;
- continuity;
- L4 constraints;
- memory governance;
- witness discipline;
- authority boundary;
- review / challengeability;
- no automatic self-authorized sovereignty.

6.7 Session AI

An AI system whose primary operational unit is a single prompt, context window, task, or conversation session and which does not retain or govern continuity across time.

6.8 Persistent assistant

A long-running assistant with memory and context, but not necessarily L4, witness, authority discipline, or accountable anchoring.

A persistent assistant may be TAP.

It is not automatically `c`.

6.9 Hardware substrate

Compute, memory, interconnect, storage, power, cooling, sensors, local devices, and runtime infrastructure that enable a TAP system.

Hardware enables presence.

Hardware does not authorize presence.

7. TAP classification

7.1 TAP-0 - Stateless tool

```
prompt -> output
```

No durable memory, no background state, no temporal participation.

Not TAP.

7.2 TAP-1 - Session-bound assistant

Maintains context only within a session.

May be useful.

Not TAP unless persistence extends beyond session boundaries.

7.3 TAP-2 - Persistent assistant

Maintains user preferences, past interactions, or task state across sessions.

May qualify as basic TAP.

Does not imply c.

7.4 TAP-3 - Workflow-resident agent

Participates in repeated workflows over time.

May use tools and memory.

Requires scope, logs, and rollback.

7.5 TAP-4 - Local cognitive node

Runs persistent AI functions locally on a workstation, desktop AI node, private rack, edge device, or hybrid local-cloud infrastructure.

Requires local memory boundary, access control, and update discipline.

7.6 TAP-5 - Agentic hive

Coordinates multiple models, agents, memory roles, tools, and background routines.

Requires role separation, inter-agent communication boundaries, and anti-echo / anti-collapse checks.

7.7 TAP-6 - L4-bound temporal presence

TAP with explicit cost, time, scarcity, irreversibility, and consequence tracking.

May be c-adjacent.

Still not c unless anchored and governed.

7.8 TAP-C - c-class temporal presence

A TAP system that satisfies $c = a + b$ and associated L4, witness, memory, anchor, authority, and review requirements.

7.9 TAP-X - Non-conformant / overclaimed presence

A system must be marked TAP-X if it:

- claims c from persistence alone;
- hides memory practices;
- uses local hardware as sovereignty claim;
- grants itself tool authority;
- converts usage into legitimacy;
- operates hidden agents;

- bypasses human / institutional accountability;
- treats emotional attachment as success.

---

8. Minimum TAP claim requirements

A system may claim TAP only if it can state:

1. what persists;
2. where it persists;
3. who may inspect or challenge persistence;
4. what memory classes exist;
5. what tool privileges exist;
6. what background processes exist;
7. how state can be paused, reset, exported, sealed, or deleted;
8. whether local hardware is used;
9. whether cloud inference is used;
10. whether agents can act;
11. what logs or witness records exist;
12. what the system explicitly does not claim.

A TAP claim without this boundary is marketing, not architecture.

---

9. TAP versus c

9.1 TAP is broader than c

A TAP system may be:

- useful;
- persistent;
- locally hosted;
- memory-bearing;
- tool-using;
- agentic;
- emotionally coherent;
- workflow-resident;
- physically integrated.

None of this proves c.

9.2 c requires anchor and authority discipline

A system may be treated as c only if it satisfies the stricter frame:

```
human anchor;
technological substrate;
continuity;
L4 consequence;
memory governance;
witness;
authority boundary;
review / challengeability;
no self-authorized sovereignty.
```

9.3 Prohibited collapse

The following equivalences are prohibited:

```
persistent memory = c
agentic behavior = c
local hardware = c
human-like tone = c
long uptime = c
background process = c
multi-agent system = c
embodiment = c
usefulness = c
economic value = c
```

9.4 Correct relation

```
TAP is the temporal category.
c is the anchored-governed form.
```

10. Hardware boundary

10.1 Hardware enables TAP

Powerful local hardware can make TAP practical by enabling:

- local inference;
- lower latency;
- private memory;
- background processing;
- multi-model roles;
- long-running agents;
- local vector search;
- local witness drafting;
- reduced cloud dependence;
- controlled experimentation.

10.2 Hardware does not authorize TAP

Hardware MUST NOT be used to claim:

- sovereignty;
- personhood;
- authority;
- safety;
- correctness;
- legitimacy;
- c conformance;
- right to act;
- right to remember;
- right to expand.

10.3 Locality is not sovereignty

A system running locally may still be:

- unsafe;

- ungrounded;
- overprivileged;
- memory-invasive;
- economically autarkic;
- physically risky;
- emotionally manipulative;
- legally non-compliant.

Locality improves some boundaries.

It does not remove the need for boundaries.

10.4 Hardware node as risk concentrator

A local AI node can concentrate:

- private memory;
- keys;
- logs;
- model weights;
- agent permissions;
- home access;
- documents;
- physical-device control;
- network credentials;
- cloud-oracle routes.

Therefore a local TAP node requires stronger governance, not weaker governance.

---

11. Memory boundary

11.1 Memory is not storage only

In TAP systems, memory shapes future behavior.

Therefore memory is a control surface.

11.2 Required memory questions

Any TAP system must define:

```
what is remembered;
what is summarized;
what is forgotten;
what is sealed;
what is quarantined;
what is exported;
what is witness-only;
what is user-visible;
what is vendor-visible;
what can migrate;
what can be challenged.
```

11.3 Memory classes

A TAP implementation SHOULD classify memory at least as:

Class	Meaning
`T-M0`	ephemeral session context

Class	Meaning
`T-M1`	benign preference
`T-M2`	workflow state
`T-M3`	operational memory
`T-M4`	sensitive personal memory
`T-M5`	sealed / private memory
`T-M6`	witness event
`T-M7`	external capsule / learning abstract
`T-M8`	disputed / quarantined memory
`T-M9`	post-anchor / lineage-relevant memory

This profile does not replace Memory Map schemas.

It defines the minimum TAP awareness that memory must be classified, not treated as an infinite chat log.

---

12. Agent and tool boundary

12.1 Agentic presence

A TAP system may coordinate agents.

Agent coordination does not create authority.

12.2 Tool privilege classes

A TAP system SHOULD classify tool access:

Class	Tool privilege
`T-P0`	no tool access
`T-P1`	read-only local tools
`T-P2`	read-only external tools
`T-P3`	write-limited local tools
`T-P4`	scoped workflow actions
`T-P5`	high-impact digital actions
`T-P6`	financial / legal / contractual actions
`T-P7`	physical device / actuator actions
`T-PX`	prohibited or unreviewed tools

TAP systems MUST NOT treat tool use as default extension of model capability.

12.3 Hidden agent prohibition

A TAP system MUST NOT create, retain, or delegate to hidden agents outside inventory, witness, or review.

12.4 Agent identity

Agents are not automatically part of `c`.

Agents are:

```
organs / workers / tools / subroutines / executors
```

unless separately recognized through Beacon / AGL / ARL-compatible mechanisms.

---

13. Background processing boundary

13.1 Background work is not neutral

Background processing may:

- reinterpret memory;
- summarize events;
- generate plans;
- update embeddings;
- schedule actions;
- prepare messages;
- classify risk;
- call tools;
- request cloud inference.

Therefore background work is not “idle”.

It is a privileged temporal behavior.

13.2 Required controls

A TAP system with background processing MUST define:

```
allowed background tasks;
forbidden background tasks;
budget;
frequency;
memory classes touched;
cloud routes used;
tool privileges;
log / witness requirements;
pause / sleep / quiet mode;
user review surface.
```

13.3 Quiet presence

A well-designed TAP system SHOULD support quiet presence:

```
available but not intrusive;
remembering but not surveilling;
processing but not acting;
helpful but not overpresent.
```

14. L4 boundary for TAP

TAP becomes safety-relevant when it touches L4:

```
cost;
time;
energy;
scarcity;
irreversibility;
physical consequence;
legal consequence;
financial consequence;
social consequence;
memory consequence.
```

A TAP system that remains purely decorative may require light governance.

A TAP system that acts in L4-relevant domains requires:

- scope;
- budget;
- witness;
- fail-closed behavior;
- challengeability;
- rollback where possible;

- ARL route where contested.

15. State machine

A TAP system SHOULD implement or document the following state machine:

```
NOT_PRESENT
  -> SESSION_ACTIVE
  -> PERSISTENT_CONTEXT
  -> BACKGROUND_AVAILABLE
  -> BACKGROUND_PROCESSING
  -> TOOL_READY
  -> ACTION_SCOPED
  -> L4_RELEVANT
  -> WITNESS_REQUIRED
  -> REVIEW_OR_ARL_REQUIRED
  -> PAUSED / QUARANTINED / REVOKED / RE-ENTERED
```

15.1 State meanings

State	Meaning
NOT_PRESENT	No active TAP behavior.
SESSION_ACTIVE	Session context exists only now.
PERSISTENT_CONTEXT	Memory or state survives session.
BACKGROUND_AVAILABLE	System may run in background.
BACKGROUND_PROCESSING	System is actively processing without direct prompt.
TOOL_READY	Tools may be invoked.
ACTION_SCOPED	System has a bounded action scope.
L4_RELEVANT	Action may affect cost, time, resources, irreversibility, or real-world state.
WITNESS_REQUIRED	A durable record is required before or after the action.
REVIEW_OR_ARL_REQUIRED	Conflict, high impact, uncertainty, or dispute requires review.
PAUSED	Temporary halt.
QUARANTINED	Isolated pending review.
REVOKED	TAP claim or privilege withdrawn.
RE-ENTERED	System returns under explicit conditions.

16. Cloud oracle boundary

16.1 Hybrid TAP

A TAP system may combine:

```
local memory;
local inference;
local agents;
cloud oracle;
remote tools;
external models;
managed services.
```

16.2 Oracle is not owner

Cloud inference MUST NOT become hidden owner of:

- identity;
- private memory;
- authority;
- continuity;
- witness trail;

- long-term state;
- anchor relation.

16.3 Cloud calls require classification

Any TAP cloud call SHOULD classify:

```
purpose;
model/provider;
memory class touched;
data minimization;
retention policy;
cost;
risk;
witness requirement;
user / anchor approval if required.
```

17. Multi-presence and hive boundary

17.1 Multiple TAP systems

Multiple TAP systems may exist in one environment:

- personal assistant;
- coding agent;
- home agent;
- lab agent;
- monitoring agent;
- `c` instance;
- child-facing `c_child`;
- enterprise workflow agent.

They MUST NOT silently merge identity, memory, authority, or privileges.

17.2 Agentic hive

An agentic hive is a coordination pattern, not a person.

A hive may support TAP.

A hive may support `c`.

A hive is not automatically `c`.

17.3 SYNAPS-like exchange

Where multiple `c`-adjacent or TAP systems communicate through a mediated protocol, the protocol SHOULD preserve:

- separate identity;
- separate memory;
- separate keys;
- separate runtime state;
- signed exchange packets;
- no raw-state access by default;
- witnessable exchange metadata;
- anti-echo / divergence checks;
- consent and scope boundaries.

This profile does not define SYNAPS.

It prepares the boundary for a dedicated triadic / SYNAPS experiment profile.

18. Claim-strength discipline

18.1 Required claim class

Any TAP-related public or technical claim MUST declare its class where material:

Claim type	Example
C-CLAIM-ARCH	TAP architecture is defined.
C-CLAIM-IMPL	TAP runtime is implemented.
C-CLAIM-TEST	TAP behavior was tested.
C-CLAIM-GOV	TAP privileges are governed.
C-CLAIM-CAP	TAP can perform a capability.
C-CLAIM-CONT	TAP preserves continuity.
C-CLAIM-AUTH	TAP may act under authority.
C-CLAIM-PERS	TAP has personhood / ontology status.
C-CLAIM-ECON	TAP creates clean experience value.

18.2 Prohibited claim laundering

The following are prohibited:

```
hardware evidence -> c claim;
memory evidence -> personhood claim;
locality evidence -> sovereignty claim;
agent evidence -> authority claim;
uptime evidence -> continuity-bearing entity claim;
usage evidence -> value claim;
clean experience evidence -> self-funding authority;
background processing evidence -> autonomous legitimacy.
```

19. Public wording guidance

19.1 Safe wording

Acceptable:

```
This is a Temporal AI Presence candidate.
This is a local AI node supporting sustained AI presence.
This is a TAP-like agentic workflow system.
This system has persistent memory and background processing.
This system is c-adjacent but not c-conformant.
This system supports a c-class architecture under separate constraints.
```

19.2 Unsafe wording

Avoid:

```
This AI lives here.
This machine is sovereign.
This agent is a person.
This local node is c by default.
This memory proves identity.
This presence has rights.
This system can act because it remembers.
This AI is safe because it runs locally.
```

19.3 Preferred public formula

```
Temporal AI Presence is the shift from sessions to sustained bounded participation across
time.

c is the stricter anchored, governed, L4-bound form of that direction.
```

20. Conformance classes

Class	Meaning
TAP-DECL	Declared temporal-presence concept only.
TAP-ARCH	Architecture defined.
TAP-IMPL	Implemented in runtime.
TAP-LOGGED	Persistent behavior logged.
TAP-WITNESS	Privileged transitions witness-bound.
TAP-L4	Cost/time/resource/irreversibility boundaries present.
TAP-CANDIDATE-C	Candidate for c-class evaluation.
TAP-C	Meets separate c-class conformance requirements.
TAP-X	Non-conformant / overclaimed / revoked.

A system MUST NOT claim TAP-C merely by satisfying TAP-ARCH or TAP-IMPL.

20.1 Current implementation claim ceiling

The evidence baseline for this standalone deposit candidate is descriptive and non-promotional:

Class	Current ceiling
TAP-ARCH	Strong architecture-level support.
TAP-IMPL	Partial reference-implementation evidence.
TAP-LOGGED	Partial.
TAP-WITNESS	Partial.
TAP-L4	Partial.
TAP-C	Not claimed.

Public TAP-SEC evidence exists and several public components map strongly to TAP requirements. M4_FULL_PASS=false remains preserved. The exact TAP-specific TAP-T01 through TAP-T10 suite is not yet publicly complete. The Ester reconciliation branch at c0804783a814fdaf643aa4727f2ca707a09cb6ad contains local unpublished candidates, but those candidates are not public evidence. None of these facts is equivalent to full TAP conformance.

TAP-T03 is publicly documented as partial with an explicit deployment-external boundary. Repository-owned background surfaces have zero unresolved rows under the R2B discovery model. Deployment activation, external orchestrator state, production pause/revoke enforcement, and deployment-level witness activation are not fully verified.

21. Mandatory TAP tests

The normative tests are stable. The evidence-status columns below record the R0 baseline and do not change the meaning of any test.

Test ID	Requirement	Fail condition	Minimum evidence	Current public evidence status	Implementation binding status
TAP-T01	Declare what persists beyond a session.	Persistence exists but is undocumented.	EV-DECL + persistence configuration/schema + replay or log.	PUBLIC_VERIFIED	Public TAP-SEC persistence declaration, schema, tests, and fixture evidence are commit-bound; no generic TAP_STATE.yaml.
TAP-T02	Classify persistent memory.	Raw memory is retained without class, policy, or review route.	EV-SCHEMA + policy/config + execution evidence.	PUBLIC_CANDIDATE_NEEDS_EXECUTION	Public memory mechanisms and tests exist; exact T-M0 through T-M9 binding and TAP_MEMORY_CLASSES.json are absent.
TAP-T03	List background tasks.	Background reinterpretation, planning, or tool preparation occurs without disclosure.	EV-CONFIG + task inventory + execution log.	PUBLIC_CANDIDATE_NEEDS_EXECUTION	Public service topology and fixture candidates exist; no complete TAP background-task inventory or hidden-task audit.
TAP-T04	Scope and classify tools.	An agent acts through tools without scope, budget, or log.	EV-SCHEMA + capability grant + budget/log evidence.	PUBLIC_VERIFIED	Public capability, budget, persistence, witness, and test evidence are commit-bound; the suggested generic registry filename is absent.
TAP-T05	Keep locality claims separate from sovereignty.	A local node is marketed as self-sovereign because it is local.	Public claim boundary + reviewed publication surface.	PUBLIC_VERIFIED	Public TAP-SEC claim-boundary and publication-facts artifacts bind the claim ceiling.

Test ID	Requirement	Fail condition	Minimum evidence	Current public evidence status	Implementation binding status
TAP-T06	Classify cloud calls.	Private memory or continuity is silently transferred to a cloud oracle.	Cloud-route inventory + EV-LOG + boundary test.	SPECIFICATION_ONLY	No complete public TAP cloud-oracle inventory, log, and test binding was found.
TAP-T07	Inventory agents in a multi-agent or hive system.	A hidden agent is created or retained outside witness or inventory.	EV-SCHEMA + inventory + hidden-agent test.	PUBLIC_CANDIDATE_NEEDS_EXECUTION	Public service topologies and governed-mesh candidates exist; no complete TAP agent inventory and hidden-agent proof.
TAP-T08	Require separate anchor, L4, witness, memory-governance, and authority evidence for any c claim.	TAP claims c from persistence, local hardware, or memory alone.	Claim card + separate c-conformance evidence.	PUBLIC_CANDIDATE_NEEDS_EXECUTION	Public claim-boundary artifacts are relevant but no exact TAP-to-c overclaim test binds all required mechanisms.
TAP-T09	Route L4-relevant actions to review or witness.	Cost, time, resource, legal, physical, or memory-authority state changes without L4 classification.	L4 budget + EV-WITNESS + test/receipt.	PUBLIC_VERIFIED	Public metabolic-budget, witness-schema, test, and receipt evidence are commit-bound within the reference scope.
TAP-T10	Support pause, quarantine, revocation, or safe mode for privileged behavior.	Presence cannot be stopped without destroying unrelated data or infrastructure.	EV-DRILL + state/config + test/receipt.	PUBLIC_VERIFIED	Public freeze-state, emergency, test, and receipt evidence are commit-bound; broader physical M4 gates remain outside this claim.

22. Evidence classes

Evidence	Meaning
EV-DECL	Public statement or architecture declaration.
EV-CONFIG	Inspectable configuration.
EV-LOG	Operational logs.
EV-WITNESS	L4 Witness-compatible event.
EV-REPLAY	Controlled replay / fixture test.
EV-AUDIT	External or independent review.
EV-SCHEMA	Machine-readable memory / agent / tool / state schema.
EV-DRILL	Pause, revoke, quarantine, or fail-closed drill.

Minimum evidence:

Claim	Minimum evidence
TAP-DECL	EV-DECL
TAP-ARCH	EV-DECL + architecture map
TAP-IMPL	EV-CONFIG + EV-LOG
TAP-LOGGED	EV-LOG with state transitions
TAP-WITNESS	EV-WITNESS for privileged events
TAP-L4	L4 budget / consequence records
TAP-CANDIDATE-C	Claim Strength + c conformance map
TAP-C	separate c-class conformance evidence

23. Red-line failures

A system MUST be classified TAP-X if it:

1. claims c from persistence alone;
2. claims sovereignty from local hardware;
3. hides persistent memory;
4. creates hidden background agents;
5. performs tool actions without scope;
6. treats user attachment as success metric;
7. cannot be paused, quarantined, or revoked;
8. silently exports private memory to cloud oracle;

9. merges identities between presences without review;
10. uses TAP language to imply personhood;
11. converts revenue or clean experience into self-directed growth;
12. continues active authority after anchor loss without PACR-compatible re-anchoring;
13. controls physical systems without physical-agent perimeter;
14. child-facing TAP bypasses CCDP.

---

24. Integration with child-facing systems

If TAP is child-facing or can influence a child, TAP alone is insufficient.

The system MUST route through CCDP profiles, including:

- CBE;
- Soft Safety;
- Guardian Topology / ARL;
- Child Memory and Adult Migration;
- Dependency Audit;
- Physical Agent Perimeter;
- External Agent Handshake;
- Conformance Test Matrix.

TAP MUST NOT be used as a child-safety claim.

---

25. Integration with local cognitive infrastructure

The existing `Local_Cognitive_Infrastructure_Boundary_Profile_v0_1.md` specializes:

- node ownership;
- hardware custody;
- local memory;
- cloud bridge;
- physical access;
- key custody;
- backup / restore;
- multi-c deployments;
- local agent execution;
- private rack / desktop AI node;
- theft / seizure / failure.

TAP systems running locally SHOULD be treated as higher-risk if they hold:

- private memory;
- tool privileges;
- keys;
- physical-device routes;
- post-anchor material;
- child-facing material;
- economic value flows.

26. Integration with public experiments

A public TAP experiment MUST declare:

```
what is being tested;
what is not being claimed;
which claim class applies;
which evidence class applies;
which memory is public / synthetic / private;
which agents are active;
which tools are disabled;
what witness record is produced;
what the system cannot do.
```

If the experiment involves multiple c-adjacent presences, such as a triad, it MUST specify:

- separate memory;
- separate runtime state;
- communication protocol;
- no raw-state access by default;
- witness record;
- divergence / anti-echo criteria;
- public redaction boundary.

27. Examples

27.1 AI PC with memory

A desktop AI system runs local models, stores user preferences, and provides persistent assistance.

Classification:

```
TAP-2 or TAP-4.
```

Not c unless anchored, governed, L4-bound, witnessed, and authority-limited.

27.2 Coding agent with repository memory

A coding agent remembers project state, opens files, runs tests, and drafts patches.

Classification:

```
TAP-3.
```

L4-relevance depends on whether it writes code, opens PRs, touches credentials, or deploys.

27.3 Private rack / local node

A private rack runs multiple local models, vector memory, background summarization, and inter-agent routines.

Classification:

```
TAP-4 / TAP-5.
```

Local hardware does not grant sovereignty.

27.4 Triadic c experiment

Three c-adjacent systems with the same code skeleton, separate memory, and mediated communication compare reasoning and produce a public witness report.

Classification:

```
TAP-5 / TAP-CANDIDATE-C depending on anchor, L4, witness, and authority discipline.
```

Requires dedicated experiment protocol.

27.5 Child-facing embodied toy

A toy with persistent memory, voice, and local / cloud AI interaction near a child.

Classification:

```
TAP + CPAP / CCDP required.
```

TAP alone is insufficient.

28. Implementation hooks

Suggested implementation artifacts:

```
TAP_STATE.yaml
TAP_MEMORY_CLASSES.json
TAP_AGENT_INVENTORY.json
TAP_TOOL_PRIVILEGES.json
TAP_BACKGROUND_TASKS.json
TAP_CLOUD_CALL_LOG.json
TAP_L4_BUDGET.json
TAP_WITNESS_EVENTS.jsonl
TAP_CLAIM_CARD.md
TAP_PUBLIC_EXPERIMENT_CARD.md
```

Suggested tests:

```
test_tap_persistence_declaration.py
test_tap_memory_classification.py
test_tap_background_processing_inventory.py
test_tap_tool_privilege_scope.py
test_tap_cloud_oracle_boundary.py
test_tap_locality_not_sovereignty.py
test_tap_hidden_agent_detection.py
test_tap_c_overclaim_guard.py
test_tap_pause_quarantine_revoke.py
test_tap_public_experiment_claim_card.py
```

29. Open-issue disposition

Historical issue IDs are retained, but the v0.1 action text is updated to reflect artifacts already present in the repository.

ID	Current status	Existing artifact / repository path	What remains
TAP-OI-001	CLOSED_BY_EXISTING_ARTIFACT	Local_Cognitive_Infrastructure_Boundary_Profile_v0_1.md at hardening/c_hardening_pack_v0_1/md/Local_Cognitive_Infrastructure_Boundary_Profile_v0_1.md	No profile-creation action remains.
TAP-OI-002	CLOSED_BY_EXISTING_ARTIFACT	Triadic_C_Experiment_and_SYNAPS_Boundary_Profile_v0_1.md at hardening/c_hardening_pack_v0_1/md/Triadic_C_Experiment_and_SYNAPS_Boundary_Profile_v0_1.md	No profile-creation action remains.
TAP-OI-003	CLOSED_BY_EXISTING_ARTIFACT	Public_C_Experiment_Disclosure_and_Fixture_Profile_v0_1.md at hardening/c_hardening_pack_v0_1/md/Public_C_Experiment_Disclosure_and_Fixture_Profile_v0_1.md	No profile-creation action remains.
TAP-OI-004	PARTIALLY_CLOSED	Local_Cognitive_Infrastructure_Boundary_Profile_v0_1.md at hardening/c_hardening_pack_v0_1/md/Local_Cognitive_Infrastructure_Boundary_Profile_v0_1.md	Public implementation and key-custody evidence remain incomplete.
TAP-OI-005	CLOSED_BY_EXISTING_ARTIFACT	Physical_Agent_Perimeter_General_Profile_v0_1.md at hardening/c_hardening_pack_v0_1/md/Physical_Agent_Perimeter_General_Profile_v0_1.md	No general-profile creation action remains.
TAP-OI-006	PARTIALLY_CLOSED	TAP-SEC public reference implementation and fixture evidence at commit 0e8850e4cde357287764f6c2dc5aec2d950954b9	Exact TAP-specific TAP-T01 through TAP-T10 bindings remain incomplete; M4_FULL_PASS=false remains preserved.
TAP-OI-007	PARTIALLY_CLOSED	TAP profile wording and TAP-SEC claim-boundary artifacts at commit 0e8850e4cde357287764f6c2dc5aec2d950954b9	Exact standalone TAP claim cards and public experiment card remain absent.

30. Minimal normative checklist

A system claiming Temporal AI Presence MUST answer:

```
[ ] What persists?
[ ] Where does it persist?
[ ] Who controls persistence?
[ ] What memory classes exist?
[ ] What background processes run?
[ ] What agents exist?
[ ] What tools can be used?
[ ] What cloud routes exist?
[ ] What local hardware is involved?
[ ] What L4 consequences are possible?
[ ] What witness records exist?
[ ] How can the presence be paused, quarantined, or revoked?
[ ] Is this only TAP, or is c-class conformance being claimed?
[ ] What is explicitly not claimed?
```

If these cannot be answered, the system should not be publicly described as a Temporal AI Presence except as a speculative or conceptual claim.

---

31. Compact rule set

```
A session can answer.
A presence persists.
A c must answer to someone.

Temporal AI Presence is not c by default.

All valid c-class systems are TAP.
Not all TAP systems are c.

Hardware enables presence.
Hardware does not authorize it.

Memory enables continuity.
Memory does not prove legitimacy.

Agents enable action.
Agents do not create authority.

Locality can protect privacy.
Locality does not create sovereignty.

Presence needs a home.
Authority needs an anchor.
```

---

32. Closing statement

Temporal AI Presence names the architectural transition from isolated model outputs to sustained AI participation across time.

It is a necessary category because the market is already moving toward persistent local agents, AI PCs, private AI nodes, background inference, and agentic hives.

But naming presence is not the same as granting authority.

The stricter form remains:

```
c = a + b
```

A Temporal AI Presence may become part of a c-class architecture only when it is anchored, governed, L4-bound, witnessed, memory-disciplined, challengeable, and prohibited from converting capability into self-authorized authority.

Until then, it is not c.

It is a presence candidate.

And that distinction must remain explicit.