[CAPABILITY PROFILE · SW-07]

GOVERNANCE KERNEL · AUTHREX GOVERNANCE KERNEL (QA10)

Give LLM agents authority tiers instead of admin keys: gate every tool call by tier and human approval before it executes, and leave signed, tamper-evident evidence of what was allowed.

IMPLEMENTED | TESTED | NOT YET VALIDATED

PRIMARY FUNCTION	Runtime decision and enforcement engine for agentic AI: authority-tier enforcement (T3 to T0) at an agent-to-proxy-to-tool boundary, human approval gating with role-based access control, signed policy loading, checkpointed hash-chain audit evidence, reviewer packet export, loopback latency measurement, sustained concurrent-load testing, and ledger tamper detection.
MISSION AREAS	Agentic AI acting on networks and infrastructure; MCP-based tool governance; software assurance for autonomous cyber and IT operations.
EVIDENCE ANCHORS	Container test suite: 83 passed, 1 skipped (environment-dependent) across 53 test functions, python:3.11-slim, package v1.0-QA10, July 2026; Python 3.11 clean install pass; Docker Desktop build, demo and pytest pass; ledger tamper detection across clean, forged and truncated records; QA10 final artifact after eleven AI-assisted review rounds.
CURRENT STANDING	Implemented in software and internally tested; a synthetic-data minimum viable product prepared for structured external technical review. Not released (public GitHub release pending counsel review; source access invited-review only). Not independently validated.
SELF-ASSESSED TRL	TRL 3 to 4 (software prototype tested in a development environment with a test suite; no external environment). No formal TRL determination has been performed.
INTELLECTUAL PROPERTY	License under review; evaluation or redistribution rights are not granted unless explicitly stated in the artifact package. Covered in part by the program's provisional applications; no per-artifact patent claim is made.

[MISSION PROBLEM]

An LLM agent that can call tools holds, in practice, whatever privilege its credentials hold. Tool misuse and credential exposure are named risk categories in the CISA, NSA and Five Eyes guidance on agentic AI services (May 2026). A model guardrail constrains outputs; it does not decide whether an action is within the agent's assigned authority at this moment, and it leaves no independent evidence of what was permitted. The consequence is a privileged action executed by an agent that was never authorized to take it, with no record a reviewer can trust.

[OPERATIONAL RELEVANCE]

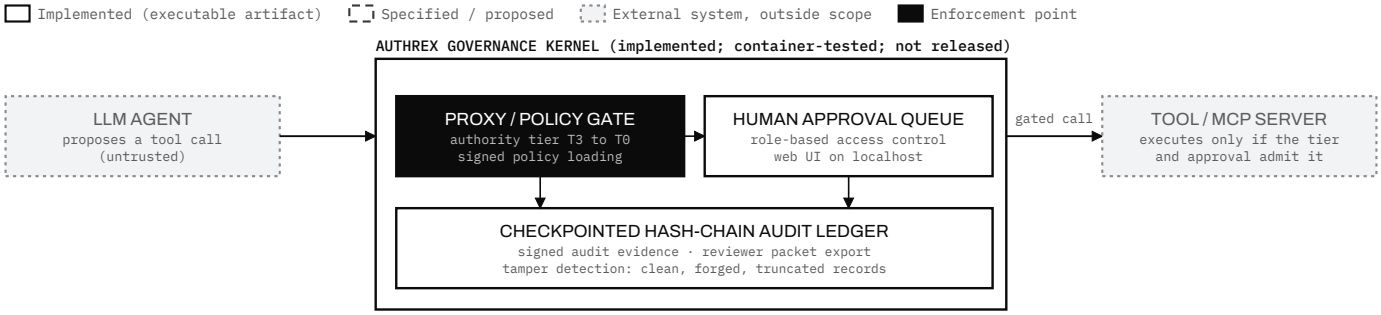
- Agentic AI operating on networks, infrastructure, and developer tooling (millisecond decision windows, tool output and retrieved content as degradable evidence).
- Software assurance and DevSecOps pipelines where an agent proposes changes to live systems.
- Zero-trust relevance: least privilege enforced per action rather than per session; auditability and provenance of every gated call.
- Cyber defense automation (AUTHREX-AGENT-CYBER profile) where an autonomous system may patch a live controller only under governed authority.

[CAPABILITY DESCRIPTION]

The Kernel is the executable enforcement engine of the program. An agent's tool call passes through a proxy where the policy gate checks the agent's authority tier against the tool's required tier; calls above the tier are routed to a human approval queue with role-based access control; approved or admitted calls are forwarded to the tool or MCP server. Every decision is written to a checkpointed hash-chain ledger with signed evidence, and a reviewer packet can be exported for external review. The demo suite exercises real HTTP agent-to-proxy-to-tool governance and real MCP SDK governance with the approval queue, a localhost web UI, loopback latency measurement, and sustained concurrent load. The Kernel calculates and enforces authority at its enforcement point; its guarantees depend on placement, platform integrity, protected keys, and the integrity of the downstream actuation path.

[GOVERNANCE KERNEL · ARCHITECTURE, ASSURANCE, EVIDENCE]

[TECHNICAL ARCHITECTURE]



Evidence: 83 passed, 1 skipped (environment-dependent) across 53 test functions in a python:3.11-slim container, package v1.0-QA10, July 2026; zero unauthorized privileged executions in synthetic scenarios. Guarantees depend on placement, platform integrity, protected keys, and the integrity of the downstream actuation path. Software-only containment can be bypassed without the hardware root of trust (BLADE-AGENT-HSM, not built).

Figure 1. Governance Kernel enforcement path. Encoding: solid = implemented; gray = external; filled = enforcement point.

[HOW IT WORKS]

1. Load signed policy (tool-to-tier map, roles, approval rules) at start; refuse unsigned policy.
2. Receive a tool call from the agent at the proxy; identify the agent and its current authority tier.
3. Compare the call against the required tier; admit, hold for human approval, or reject.
4. For held calls, present to the approval queue; an authorized role approves or denies under RBAC.
5. Forward admitted calls to the tool or MCP server; return the result to the agent.
6. Write the decision, approver, and result digest to the checkpointed hash-chain ledger with a signature.
7. On demand, verify the ledger (detecting forged records and tail truncation) and export the reviewer packet.

[TECHNICAL DIFFERENTIATION]

Role-based and attribute-based access control grant permission by identity or attribute, not by the current action and evidence; model guardrails constrain outputs regardless of which action they enable. The Kernel adds an evidence-conditioned, time-varying authority tier beneath the role and enforces it at the tool boundary with signed evidence. It composes with MCP tool servers rather than replacing them.

[ASSURANCE MECHANISMS: DESIGN PROPERTY VERSUS ESTABLISHED PROPERTY]

Mechanism	Status
Zero unauthorized privileged executions in synthetic scenarios	Tested (self-generated suite passes its full set); not independently validated
Ledger tamper detection: clean, forged, truncated	Tested in the container suite
Human approval gating with RBAC	Implemented and demonstrated (MVP demo)
Signed policy loading	Implemented
Latency and concurrent load	Measured on loopback in the test environment; not benchmarked on any target
Software-only containment	Bypassable without a hardware root of trust; BLADE-AGENT-HSM (not built) is the designed companion

[TECHNICAL EVIDENCE]

Evidence	Result	Source / artifact
Container test suite	83 passed, 1 skipped (environment-dependent); 53 test functions; package v1.0-QA10; archive hash recorded in the internal QA log	authrex.systems/kernel (site claim pending external reproduction)
Install and build	Python 3.11 clean-environment install PASS; Docker Desktop (Mac) build PASS; containerized demo end to end as unprivileged user PASS	kernel.html
Alpha and MVP demos	HTTP agent-to-proxy-to-tool gating; MCP SDK governance with approval queue, RBAC web UI, signed evidence, packet export, latency, load	kernel.html
Review package contents	authrex_kernel/, tests/ (53 functions), harness/, docs/EXTERNAL-REVIEWER-PACKET.md, demo_alpha.py, demo_mvp.py, Dockerfile, install.sh, VALIDATION-REPORT.md, CHANGELOG.md	by-request review package

[EVIDENCE SNAPSHOT]

EVIDENCE SNAPSHOT	
Implemented	Yes
Executable artifact	Yes (container)
Formal model	No (not yet performed)
Simulation evidence	Yes (synthetic demos)
Hardware validation	No
Field validation	No
Independent validation	No
Government evaluation	No
Operational deployment	No

[LIMITATIONS]

- Not released; source access is invited-review only while counsel review is pending; no independent security assessment; scalability not benchmarked.
- All results are from synthetic scenarios on self-generated test suites; a self-generated suite cannot establish that the policy semantics match any real mission policy.
- No formal model of the Kernel exists; the TLA+ work covers the frameworks and consoles, not this codebase.
- Software-only containment: a compromised host, leaked key, or bypassed proxy defeats enforcement.
- No SBOM, no formal vulnerability scan, no penetration test.

[TECHNOLOGY READINESS (SELF-ASSESSED; NO FORMAL TRA)]

CURRENT ASSESSED TRL	TRL 3 to 4 (self-assessed against the DoD nine-level scale; no formal technology readiness assessment has been performed).
BASIS	A working software prototype with an executable test suite in a development environment satisfies the software proof-of-concept level and approaches laboratory validation; no validation outside the author's environment and no representative workload has been run, so TRL 4 is not claimed outright.
EVIDENCE SUPPORTING	software implementation; container test suite; demonstrations on loopback
EVIDENCE REQUIRED FOR NEXT TRL	(1) External reproduction of the container test suite from the review package by a government or FFRDC team. (2) Sandbox evaluation with government-defined tool calls and approval policies (Phase 2). (3) Independent security assessment and SBOM before any release.

[INTEGRATION PROFILE]

INPUTS	Agent tool-call requests; signed policy; approver actions; agent identity.
OUTPUTS	Admit / hold / reject decisions; forwarded calls; ledger; reviewer packet.
INTERFACES	HTTP proxy; MCP SDK; localhost web UI; file export.
DEPLOYMENT	Containerized (python:3.11-slim); runs as an unprivileged user; localhost only in the current package.
COMPUTE REQUIREMENTS	Not characterized beyond loopback latency in the test environment.
DEPENDENCIES	Python 3.11; Docker; MCP SDK; no external network required for the demo.
SECURITY BOUNDARY	Inside: proxy gate, approval queue, ledger, policy signature check. Outside: agent, tools, host OS, key storage. Key protection is the platform's responsibility.
DATA REQUIREMENTS	Synthetic; government-defined tool inventories and approval policies (unclassified) suffice for Phase 2.

[TRANSITION ROADMAP]

Phase	Activity for this capability	Who can execute it	Exit evidence
1 Technical review	Review of the artifacts listed in the evidence table	Program alone, with a government reviewer	Reviewer findings; correction log entries
2 Government-defined simulation	Sponsor scenarios on the seeded console	Program with sponsor scenarios	Deterministic, replayable results; hash-chained records
3 Degradation and failure injection	Evidence lapse, spoofing, delayed human response	Program with sponsor parameters	False-restriction, miss, and contraction-latency figures
4 Government testbed / SIL	Integration with a representative controller or interface	Requires sponsor interface and testbed	SIL integration report
5 Hardware-in-the-loop	Representative interference; timing on real hardware	Requires sponsor laboratory or rig	Measured latencies; TRL 4 to 5 evidence
6 Relevant-environment demonstration	Independent evaluation and red team	Requires independent evaluator	Independent report
7 Program integration	Only after earlier gates succeed	Requires transition partner	Not claimed to be available

[GOVERNMENT ENGAGEMENT REQUEST (SPECIFIC)]

- Structured external technical review of the review package (the program's stated next evidence priority).
- Government-defined tool calls and approval policies for a sandbox evaluation.
- Independent security assessment (code review, dependency scan, penetration test) by a cyber test organization.
- Guidance on an accreditation-relevant evidence format (RMF artifacts) for a future release.

[REVIEWER QUESTIONS THIS PROFILE ANSWERS]

What is it: see capability description. Why does DoD care: see operational relevance. Where would it fit: see integration profile. Does it exist: see evidence snapshot. What evidence supports it: see technical evidence. Has anyone independent tested it: no. Has it been tested in the real world: no. What does it rely on: see limitations and security boundary. What can cause it to fail and what does it do then: see assurance mechanisms and the published counterexamples. How mature: see the TRL block. What would the government need to provide: see the request above. Next measurable milestone: first item under evidence required for next TRL.