MCP-Bastion 2.0.0

The Zero-Trust control plane for MCP agents

Wrap any MCP server with local guardrails in three lines of code: agent IAM, supply-chain checksums, prompt-injection blocking, PII redaction, and denial-of-wallet caps. Ships today on PyPI, npm and Docker, with a live dashboard.

PyPI · npm · Docker / GHCR · FastMCP · TypeScript · 631 tests · 92% coverage

AT A GLANCE

18
SECURITY PILLARS

10 / 10
OWASP MCP TOP 10 COVERED

< 5 ms
HOT-PATH OVERHEAD

~99%
TOKEN CUT ON BIG TOOL OUTPUT

THE 2026 MCP MOMENT

Runtime governance is the new must-have

- MCP turned every server into an agent gateway. Agents now call databases, APIs and shell tools on behalf of an LLM.
- The NSA MCP guidance and the OWASP MCP Top 10 both point to **runtime governance** at the protocol boundary as the answer.
- MCP-Bastion is that control plane: **identity-aware routing**, supply-chain verification, and local data protection, purpose-built for agents.
- One drop-in middleware secures the whole MCP surface, so teams ship agentic features with confidence.

THREAT PREVENTION AND CONTENT SAFETY

Block injection. Redact PII. Keep data local.

Prompt injection defense
Meta PromptGuard scores tool arguments; jailbreaks blocked pre-exec

Response scan
Blocks jailbreak patterns hidden in tool / resource output

Heuristic blocking offline
Obvious jailbreaks stopped out of the box, no model download

Grounding guard
Verifies file paths referenced in tool output are real (opt-in)

Content filter
Shell / code patterns, sensitive file paths, URLs, allow/deny rules

100% local execution
Classification and redaction stay inside your network

PII redaction (Presidio)
Masks SSN, email, phone, cards, passport, IBAN in outbound text

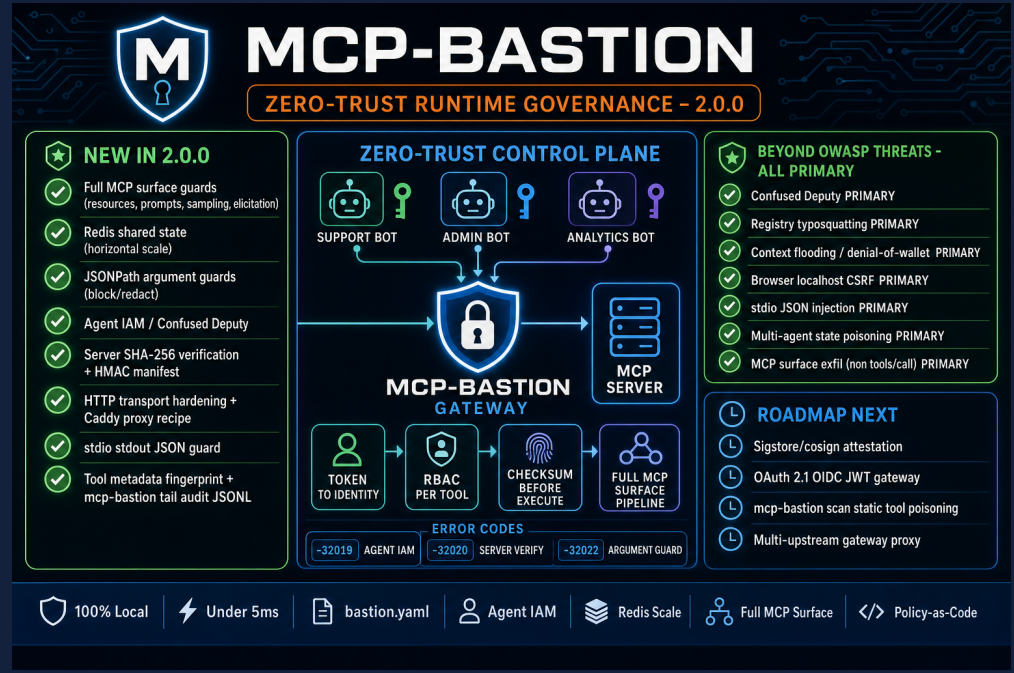
Schema validation
Rejects malformed / unexpected tool arguments before execution

Local inference, no third-party safety API. Sensitive data never leaves your enterprise network.

ZERO-TRUST ACCESS AND IDENTITY

Identity-aware routing stops the confused deputy

- **Agent IAM:** bind API tokens to agent identities with per-agent allow / block tool lists and resource URI rules. Your support bot calls `search_docs`, never `delete_user`.
- **RBAC with fnmatch globs:** permissions like `read_*` with specificity-aware matching.
- **Edge auth:** shared-secret token check on gateway-issued requests.
- **Session tool-scope cap** and **replay guard** stop scope creep and replayed calls.
- **JSONPath argument guards:** block or redact tool arguments by tool glob + JSONPath + regex.



SUPPLY CHAIN AND INTEGRITY

Verify every server before a single tool runs

Server verification
SHA-256 manifest checksums at startup and on every tools/call

Tool metadata guard
Sanitizes poisoned tools/list descriptions and schemas

mcp-bastion manifest
Generate a trusted manifest after a signed-off build

Semantic firewall
Flags dangerous tool intent and risky call chains

Typosquatting defense
Block traffic when server artifacts drift from your checksums

stdio guard
Drops non-JSON lines on stdout to protect IPC integrity

Circuit breaker
Opens after repeated tool failures to contain cascades

doctor CLI
Preflight config and supply-chain checks before you deploy

Cryptographic integrity plus behavior checks give you supply-chain confidence at the MCP boundary.

STANDARDS COVERAGE

All 10 OWASP MCP Top 10 risks, mapped to controls

MCP01 Token exposure
PII redaction, response scan, audit trail

MCP02 Privilege escalation
RBAC + Agent IAM, rate limits, session scope

MCP03 Tool poisoning
Prompt guard, content filter, metadata guard

MCP04 Supply chain
Manifest checksums, doctor CLI, circuit breaker

MCP05 Command injection
Prompt guard, argument guards, schema validation

MCP06 Intent subversion
Rate limits, replay guard, semantic firewall

MCP07 Weak auth
RBAC, edge auth, per-agent tokens

MCP08 Audit and telemetry
Audit log, dashboard, Prometheus, OTEL, alerts

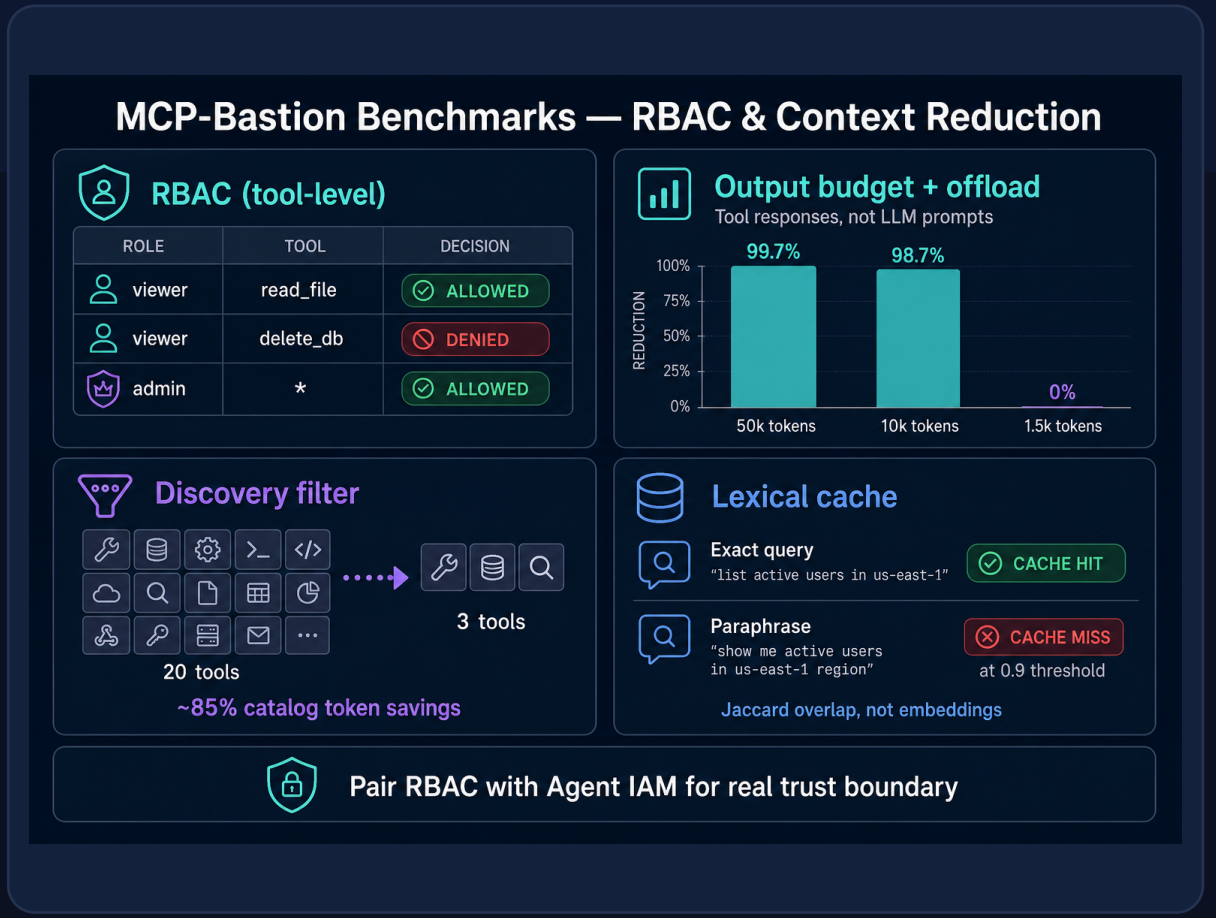
MCP09 Shadow MCP servers
Central policy, discovery filter, metrics

MCP10 Context injection
Output budget, response scan, grounding guard

FINOPS AND COST SAVINGS

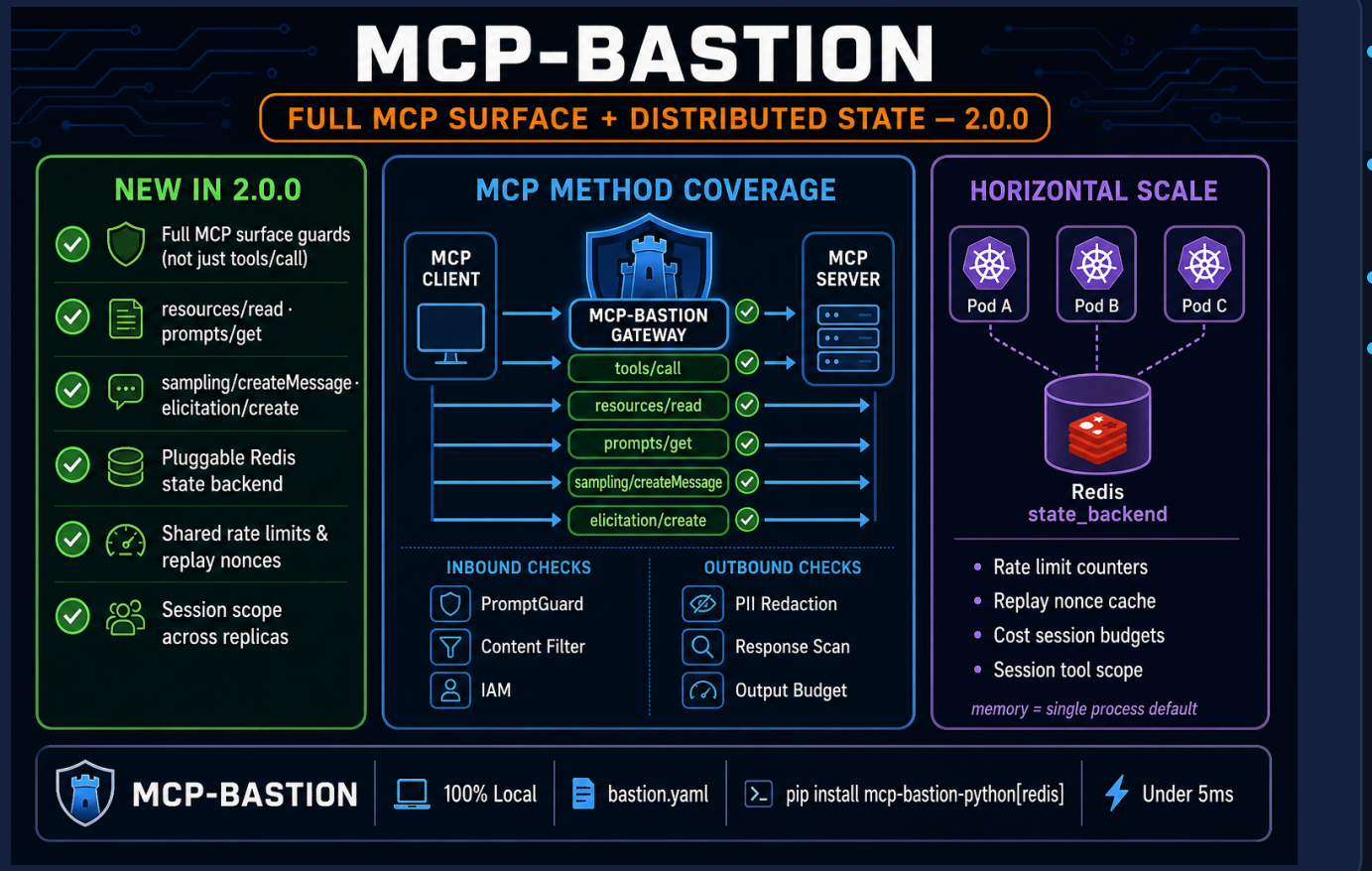
Stop denial-of-wallet. Cut token spend.

- **On by default:** 15 tool calls / session, 60s timeout, 50k token budget.
- **USD caps** per session and per day, plus per-tool caps and circuit breaker.
- **Output budget and offload:** up to ~99% token cut on oversized tool responses, 0% when already under budget. Measured and reproducible.
- **Discovery filter** trims the tool catalog by ~85% (20 tools to 3).
- Cost attribution by provider, model, tool and tenant.



2.0.0 HIGHLIGHT

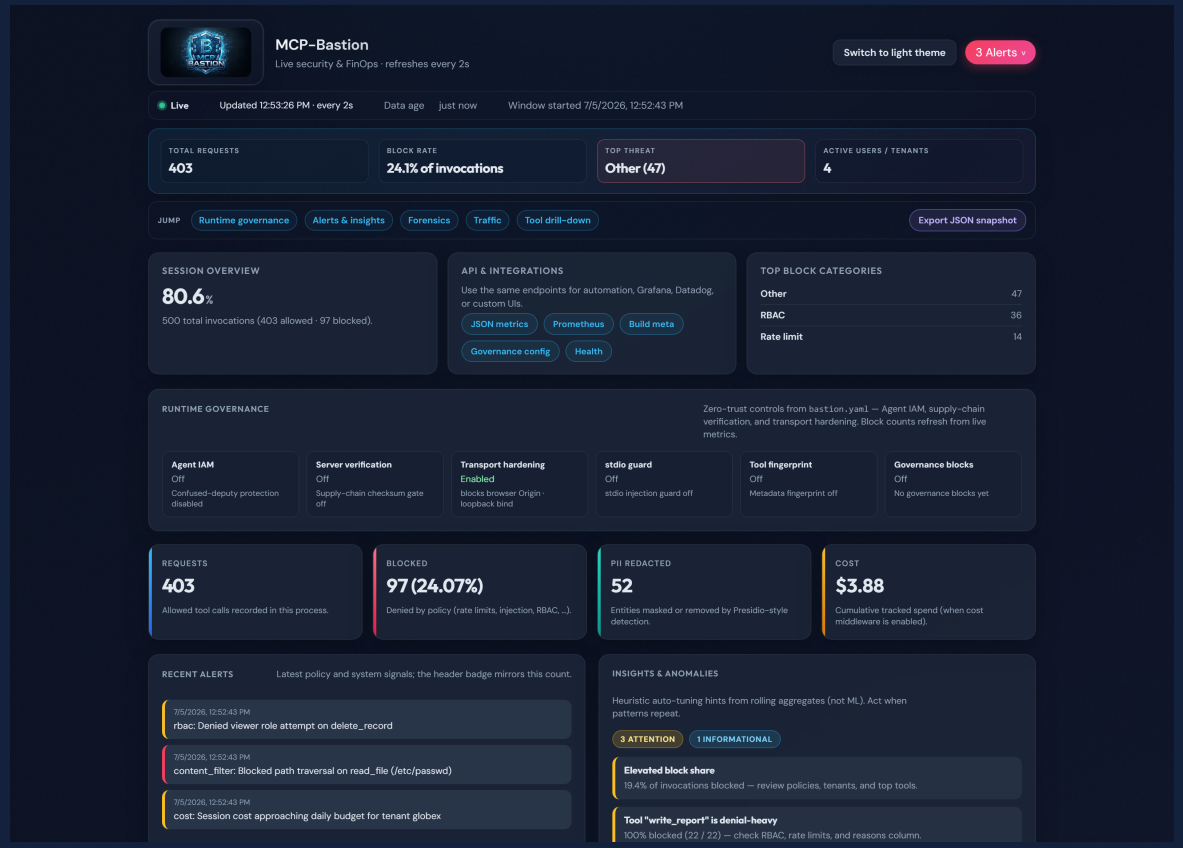
Full MCP surface, and it scales out



- Same pillars on **tools/call, resources/read, prompts/get, sampling** and elicitation, not just tool calls.
- **Redis state backend** shares rate limits, replay nonces, cost budgets and session scope across workers and pods.
- **Cost checkpoints** persist session totals across restarts.
- Standard JSON-RPC error codes for every decision, easy to log and alert on.

OBSERVABILITY

Live dashboard: every block, threat and dollar



Real-time KPIs, blocked-by-reason charts, PII forensics, FinOps and per-tenant views. Export to Prometheus /metrics, JSON /api/metrics, OTEL traces and Slack alerts. Run: `mcp-bastion dashboard --port 7000`

SHIPS TODAY

Secure your MCP server in three lines

```
pip install mcp mcp-bastion-fastmcp
from mcp_bastion_fastmcp import secure_fastmcp
mcp = FastMCP('My Server')
secure_fastmcp(mcp)
```

- **Policy-as-code:** copy bastion.yaml, then build_middleware_from_config().
- **Three ways to adopt:** FastMCP, policy config, or TypeScript wrapper.
- **Runs everywhere:** PyPI, npm, prebuilt Docker on GHCR, CI validate.
- **Low latency:** drop-in middleware, no business-logic changes.

PLUGS INTO YOUR STACK

- FastMCP and MCP Python SDK middleware
- TypeScript wrapper for Node MCP servers
- Docker images on GHCR (proxy + dashboard)
- Prometheus, OpenTelemetry, Slack alerts
- LangChain, OpenAI, Bedrock add-ons
- bastion.yaml in Git with hot reload
- Audit JSONL + mcp-bastion tail for compliance

Ship secure MCP agents today

```
pip install mcp-bastion-python[redis,policy,dashboard]
mcp-bastion dashboard --port 7000
```

```
mcp.add_middleware(build_middleware_from_config())
```

Built by Vaquar Khan · Feedback and PRs welcome

LINKS

- **GitHub, star us**
- **PyPI 2.0.0**
- **Documentation**
- **npm @mcp-bastion/core**