

Evolving AppSec: Using Agentic Workflows and Building Secure AI Systems

Dr. Viswanath (Vis) S. Chirravuri

Chief AI Security Governance, Thales Cyber & Digital

1 Use AI to strengthen AppSec

2 Secure the AI you build

3 Consume vendor AI safely

OWASP LASCON 2026 · Austin, TX · Friday, October 30, 2026 · 3:00 – 3:50 PM

whoami # "Vis"



20+ years in cybersecurity

ROLE

Chief AI Security Governance, Thales Cyber & Digital

Past: Crédit Agricole CIB, J.P. Morgan, CA Technologies, Capgemini

CREDENTIALS

GIAC Security Expert (GSE #335)
CISSP, PMP, 35+ certs

D.Eng. Cybersecurity Analytics, GWU

M.S. ISE, SANS.edu

TEACHING

SANS SEC545 Instructor – GenAI & LLM AppSec

SANS SEC546 Author & Instructor – Securing Agentic AI

COMMUNITY

Founder, AI AppSec Academy
RSAC speaker 2024 & 2026

OWASP, CompTIA SME, open source on GitHub

linkedin.com/in/vchirrav github.com/vchirrav-eng aiappsecacademy.com

Disclaimer



The views and opinions in this presentation are solely my own.

They do not represent the official positions of my employer, Thales, or any affiliated organizations.

This material is for informational and educational purposes only.

References to tools, products or services are based on personal experience or publicly available information.

AppSec Must Evolve in Three Dimensions

1 USE AI FOR APPSEC

Agentic workflows across the SDLC

- Threat modeling
- SAST · secrets · SCA · DAST skills
- AI remediation
- WAF rule generation

YOU LEAVE WITH
Product security AI skills

2 SECURE THE AI YOU BUILD

Defend what you ship

- MLSecOps policy
- RAG pipeline threats
- Agentic AI attack surface
- MCP hosts · clients · servers

YOU LEAVE WITH
MLOps security policy

3 CONSUME VENDOR AI SAFELY

Govern what you adopt

- Model provider assessment
- Model selection scoring
- Securing Desktop Agents
- OWASP secure coding rules

YOU LEAVE WITH
LLM assessment + coding rules

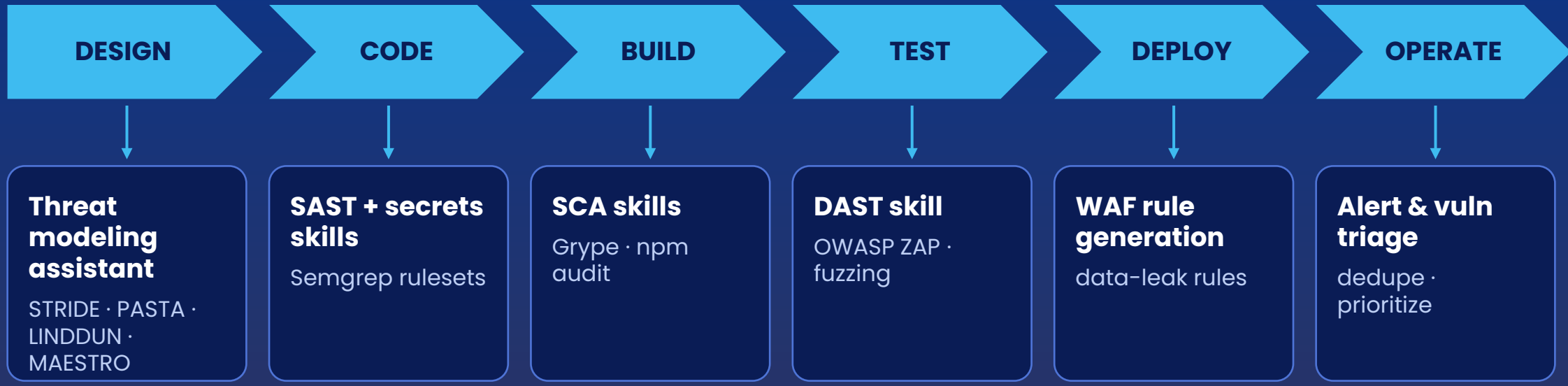
One practitioner framework: use AI → secure AI → consume AI — every tool shown is open source.

01

Use AI to Strengthen AppSec

Agentic workflows and reusable skills make traditional AppSec tooling faster, more consistent and more contextual.

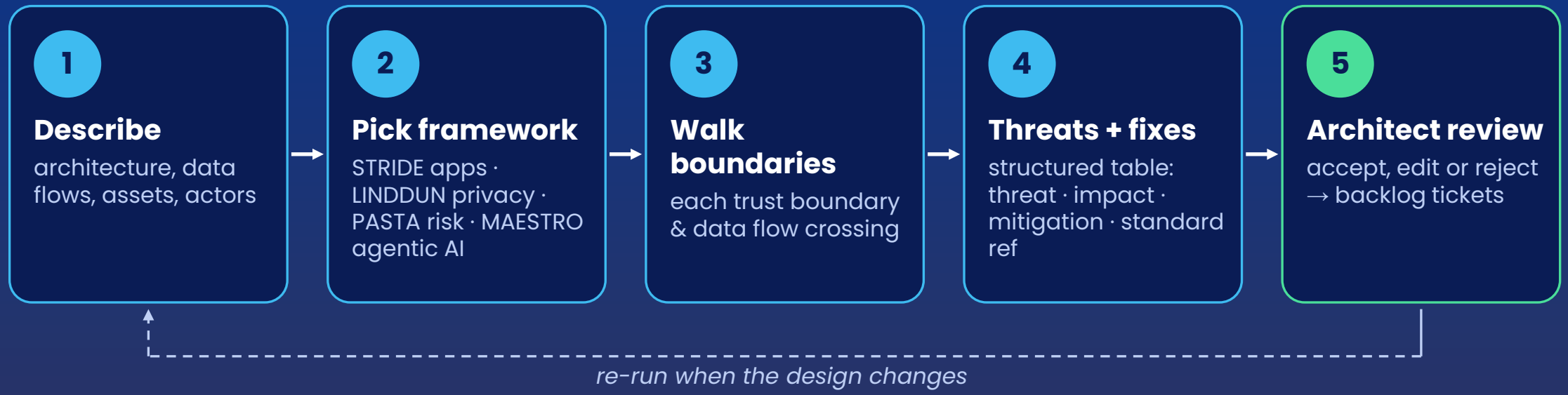
Agentic AI Across the SDLC



HUMAN GATE AI proposes → tools verify → engineers decide. Every AI output is re-checked by a deterministic scanner or a person.

Skills turn ad-hoc prompting into repeatable security automation — same tool, same rules, same output.

AI Threat Modeling: A Reviewable Workflow



TRY IT

ThreatModelingGPT — free, no login; cites established methods instead of improvising

chatgpt.com/g/g-6943215f28dc8191a9097e570c462d45-threatmodelinggpt

Why Agentic SAST: From Finding to Fix

TRADITIONAL SAST	WITH AN AGENTIC WORKFLOW	
Remediation gap Stops at discovery; generic advice; manual research to patch	→	**Agent drafts a context-aware fix and opens a PR** — SOLVED
False-positive fatigue Alert volume nobody can audit; critical bugs buried in noise	→	**Agent triages every finding and explains its verdict** — SOLVED
Blind to runtime No reachability context; flags vulnerabilities in dead code	→	**Agent traces call paths — still pair with IAST / runtime data** — PARTIAL

The scanner still finds the bug; the agent supplies the context, triage and fix.

Anatomy of a Security Skill

```
---
name: sast-semgrep
description: Run Semgrep SAST when asked
  to scan source code for vulnerabilities
---
## Run
semgrep scan --config p/owasp-top-ten \
  --sarif --output semgrep.sarif .
## Triage each finding
- map to CWE; check reachability
- verdict: true / false positive + why
## Report
| rule | file:line | CWE | severity | verdict |
```

1. **Trigger**
   the description tells the agent when to load the skill
2. **Pinned tool + ruleset**
   same scanner, same rules in every repo
3. **Deterministic command**
   the agent runs the tool — it doesn't guess findings
4. **Output contract**
   SARIF + a fixed table → comparable triage across teams

Illustrative structure — see the repo for the published skills

Skill-Driven Scanning Workflow

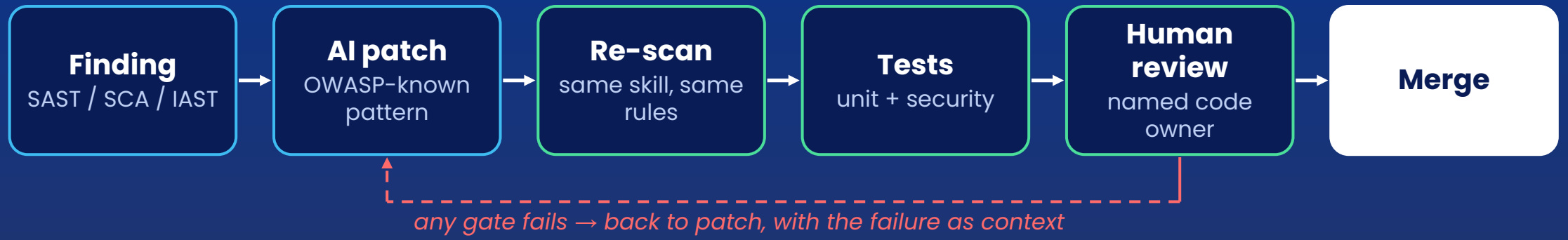


Same tool, same rules, same output — every repo, every team.

↗ github.com/vchirrav-eng/product-security-ai-skills

skills run in parallel

AI Remediation: Trust, but Verify



RISKS → THE GATE THAT CATCHES THEM

False confidence
patch looks right, adds a new bug
re-scan + tests

Missing context
breaks business logic
human review

Accountability gap
who owns a failed AI fix?
named code owner

Adversarial patching
attackers game how AI fixes CWEs on critical paths
no auto-merge

Upside: speed, scale, consistency — and teams without deep security expertise can act on findings.

AI-Generated WAF Rules for New Features

New feature
new API fields in responses

→

AI drafts rules
ModSecurity / Coraza SecRule

→

Detection only
log, don't block

→

Replay & tune
kill false positives

→

Enforce
block + monitor

```
SecResponseBodyAccess On
SecRule RESPONSE_BODY "@rx \b4\d{3}(?:[ -]?\d{4}){3}\b" \
  "id:9500001,phase:4,block,log,\
   msg:'Card number leaking in response body'"
```

Illustrative rule — response-body inspection, phase 4

7 sections

~40 rules

9500001–99 rule ID range

WHAT THE RULES STOP LEAKING

Financial data | PII | Credentials & secrets | App internals | Health / medical | Env & config leakage | Response headers

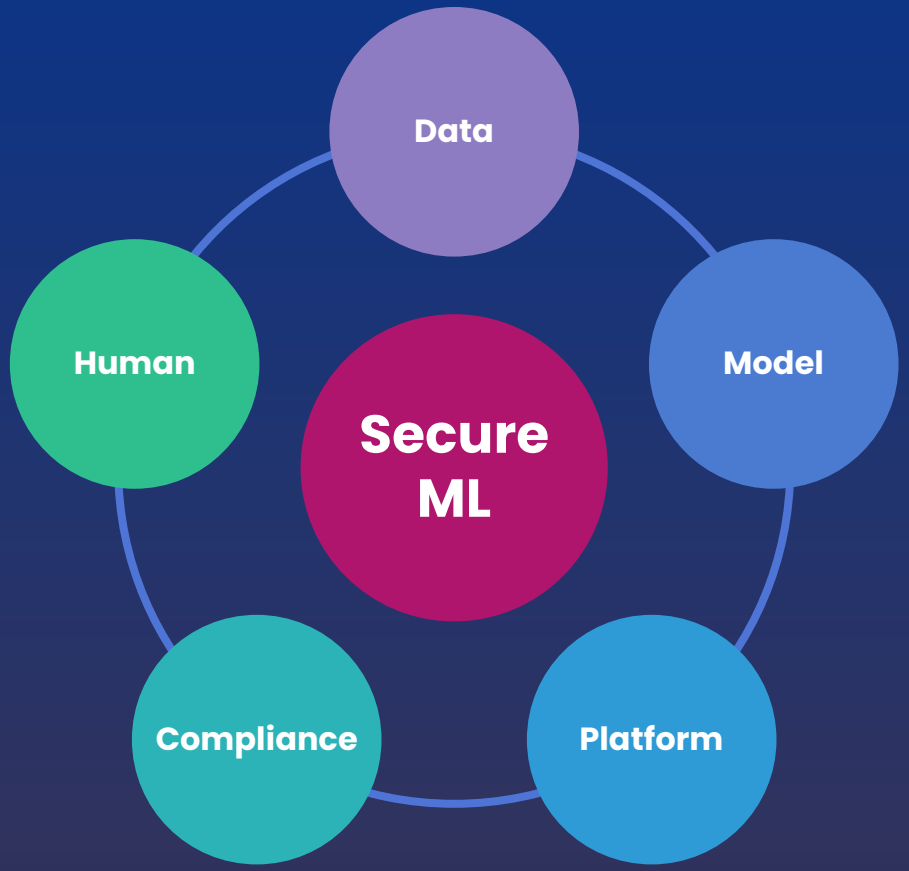
↗ github.com/vchirrav-eng/waf-rules → prevent-sensitive-data-leak.txt

02

Secure the AI You Build

Defend the AI systems your teams ship — ML pipelines, RAG, single- and multi-agent systems and MCP servers — with defense in depth.

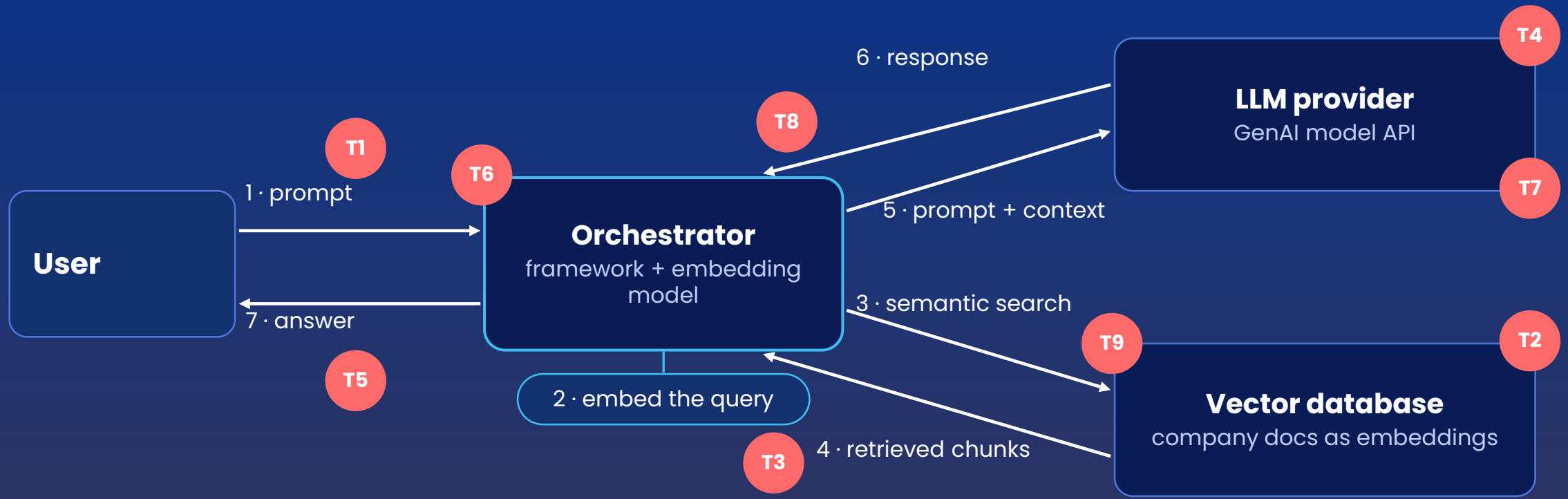
MLSecOps: An Open-Source Security Policy



- **Data security** encryption · access control · anonymization · classification
- **Model security** I/O validation · adversarial training · robustness testing · monitoring
- **Platform security** API & cloud security · vuln scanning & pen testing · hardening
- **Compliance security** regulatory · audit trails · retention & deletion · third-party risk
- **Human security** training & awareness · incident response · governance & oversight

↗ github.com/ThalesGroup/secure-ml → security-policy/ml-secpol.md (goals + techniques per pillar)

RAG Pipeline: Where the Threats Land



- T1 Prompt injection
- T2 Data poisoning
- T3 Sensitive leakage
- T4 Jailbreaking
- T5 Insecure output
- T6 Denial of service
- T7 Model theft
- T8 Hallucination
- T9 Vector manipulation

RAG Defense: A Control at Every Hop

1 — INPUT — T6 T1

Input rail: injection classifier, length & rate limits

2 — INGESTION — T2

Source allowlist, document sanitization, hashed corpus

3 — RETRIEVAL — T9 T3

Permission-aware retrieval: doc ACLs + tenant filters at query time

4 — CONTEXT — T1

Spotlight retrieved text: delimit it as data, never instructions

5 — MODEL CALL — T7 T6 T4

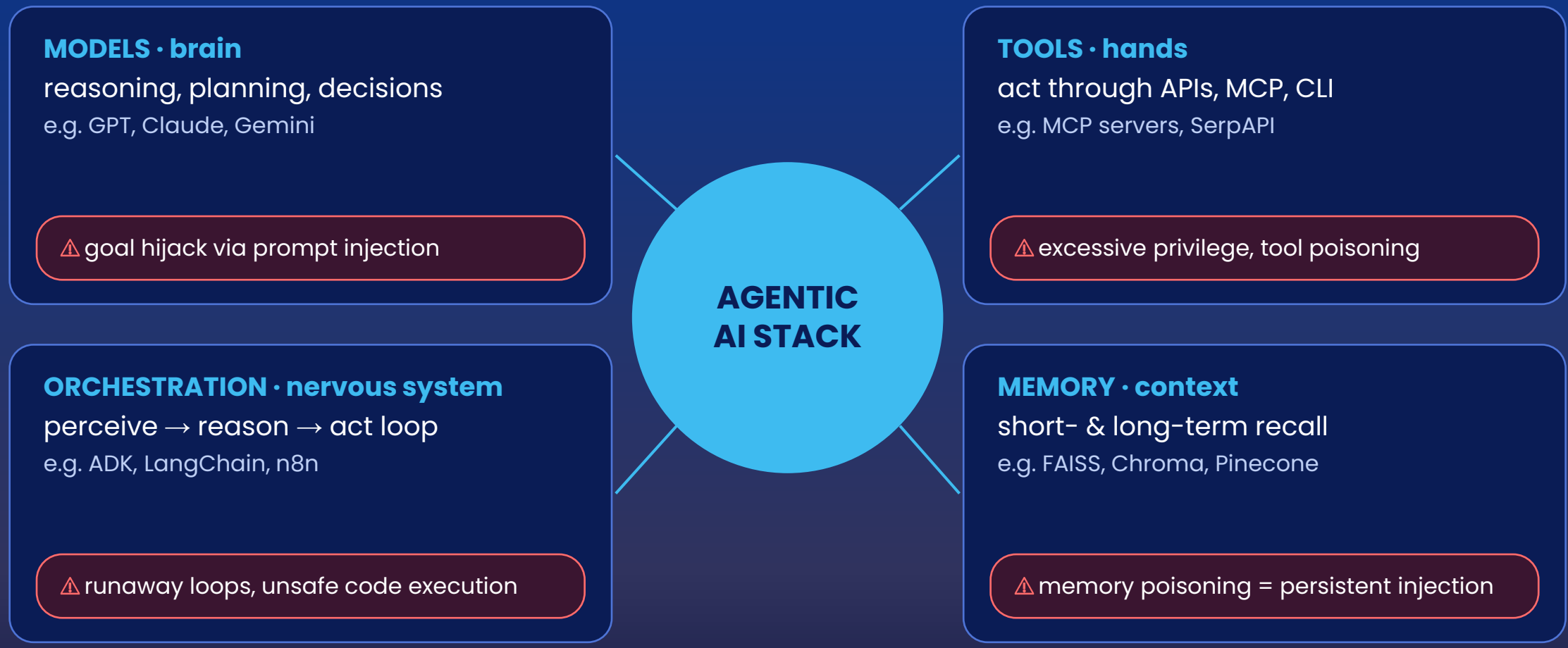
Model gateway: virtual keys, quotas, central logs

6 — OUTPUT — T8 T5

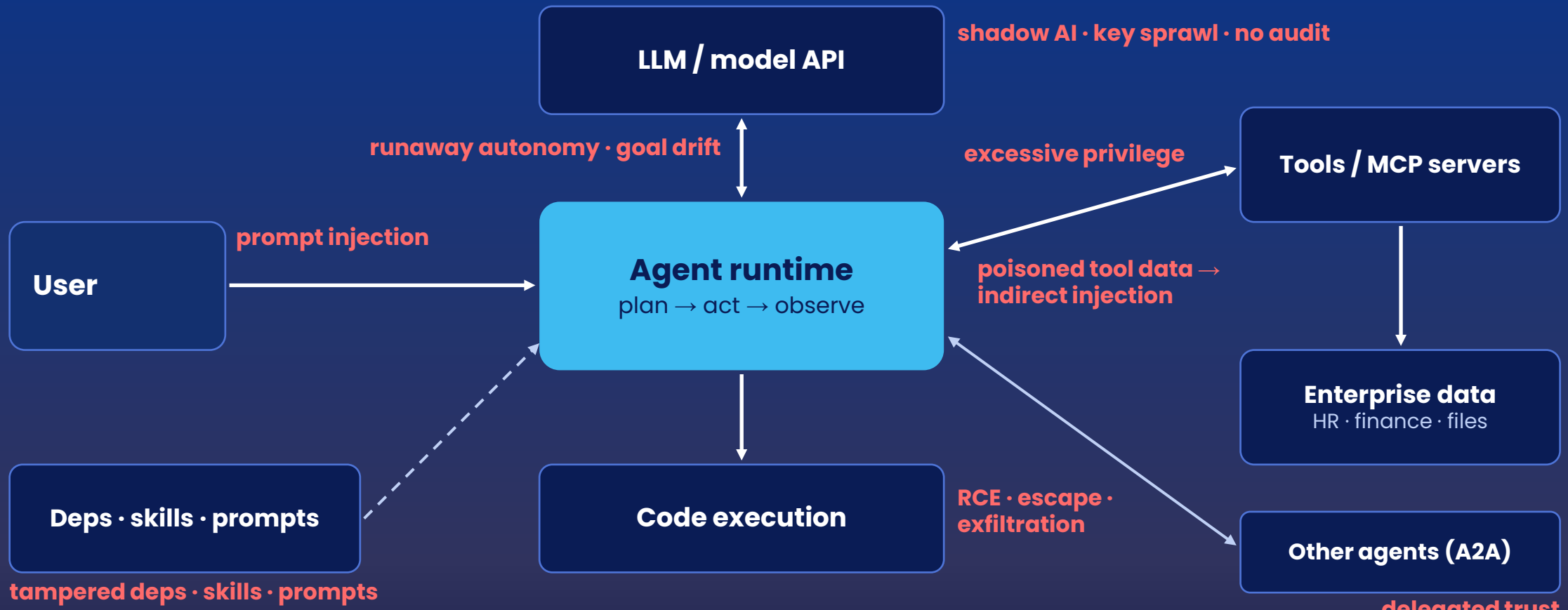
Output rail: PII masking, grounding & citation checks, encode before render

Architecture moves trust boundaries too: Agentic RAG, Self-RAG and CRAG add tool calls & web fallback — threat model each.

Agentic AI: Four Parts, Four Attack Surfaces



Inside an Agentic System: Where Risk Lives



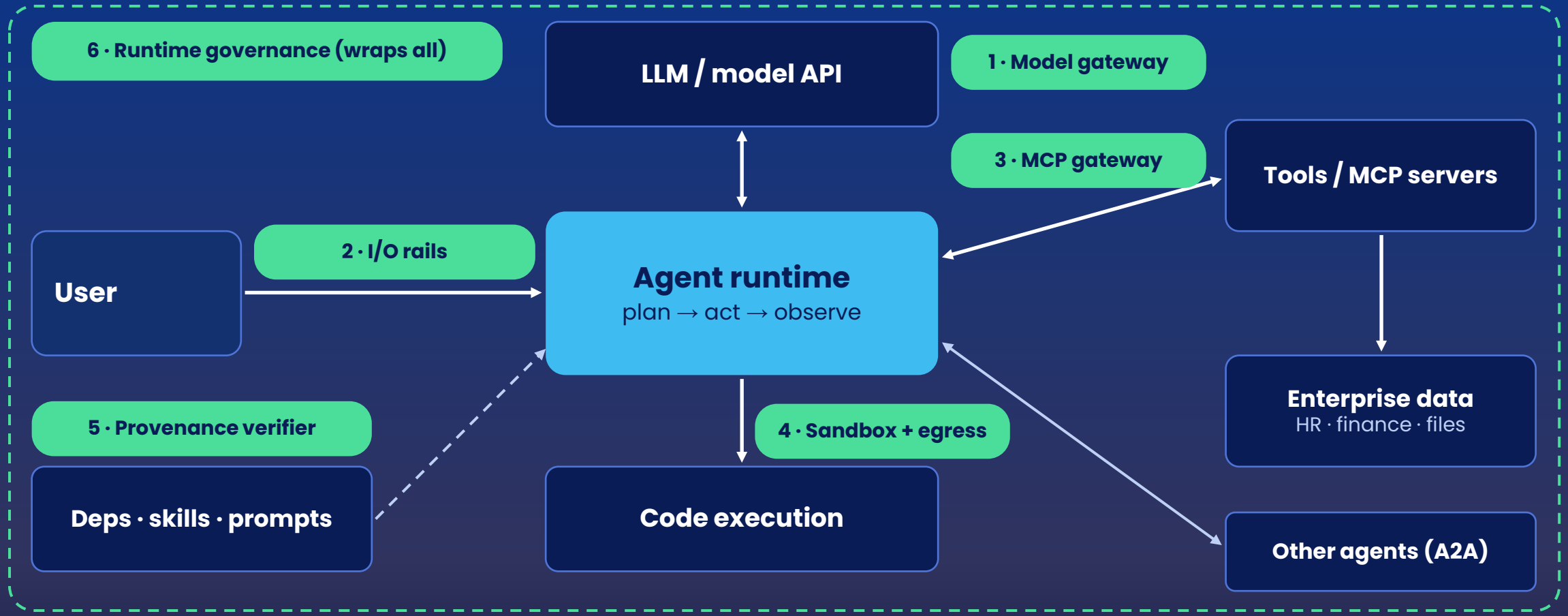
Every red label is an attack demonstrated live in SANS SEC546 labs.

Six Threats ⟶ Six Controls

	THREAT	OPEN-SOURCE CONTROL
1	Shadow AI: unmanaged model access, key sprawl	**Managed model gateway** LiteLLM · virtual keys · spend limits · central logs
2	Prompt injection: direct & via tool data	**Input / output guardrails** NeMo Guardrails judge · Presidio PII masking
3	Any agent can call any tool, full privilege	**Least-privilege MCP gateway** Bifrost · authn · per-agent tool allowlists
4	Agent-written code = RCE in your network	**Sandbox + egress control** gVisor (runsc) · Squid deny-by-default
5	Tampered deps, skills & prompts (rug-pull)	**Supply-chain provenance** content hashes · cosign · CycloneDX AIBOM
6	No visibility, silent regressions, no kill switch	**Runtime governance** policy-as-code · canary replays · kill switch

All open source. All deployable today.

Defense in Depth: Six Choke Points



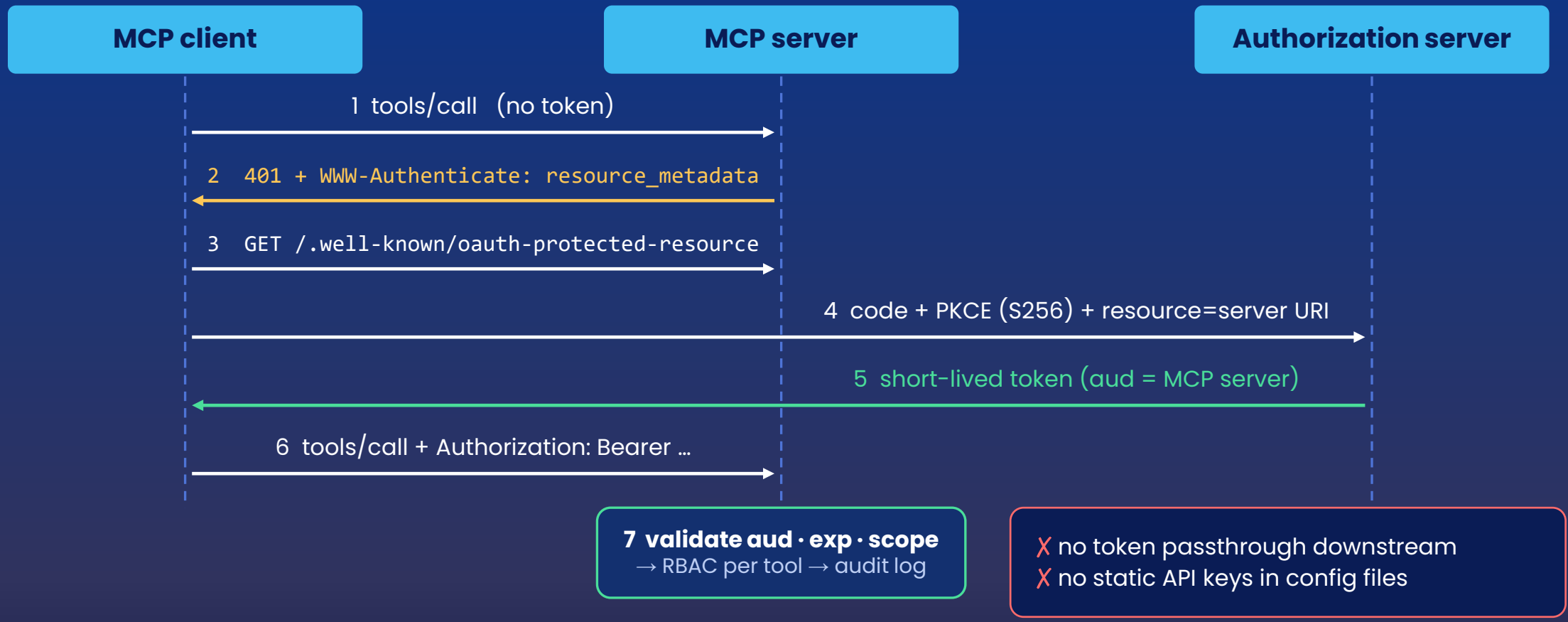
Every red label from the anatomy slide now dies at a green choke point.

MCP Security: Every Layer Verifies Trust

MCP HOST IDE, desktop app	MCP CLIENT protocol bridge	MCP SERVER executes tools
✓ Enforce user consent before tools run	✓ Validate JSON-RPC in & out	✓ Run in a sandbox
✓ Isolate clients from each other	✓ OAuth 2.1 tokens, not static keys	✓ Secrets in a vault, never in config
✓ Connect only to signed, pinned servers	✓ Re-approve on any server change	✓ RBAC per user / role per tool
✓ Monitor & flag tool calls	✓ Expose only tools the task needs	✓ Audit who, what, when, why

Key principle: no layer assumes the layer above or below is secure.

MCP Authorization Flow (OAuth 2.1)



Steps 2–3: RFC 9728 protected-resource metadata · Step 4: RFC 8707 resource indicators bind the token to one server

03

Consume Vendor AI Safely

Adopt vendor models and coding agents through structured assessment and secure-by-default rules backed by OWASP standards.

Vendor AI Governance: Assess, Score, Decide

Intake
provider · model · deployment · use case

→

Assess
provider + model questionnaires

→

Score
provider safety + model safety → combined

→

Decide
threshold → status

→

Approve
risk owner · 6–24 month validity → re-review

COMPLIANCE THRESHOLDS

80–100 APPROVED
production use

60–79 CONDITIONAL
mitigate + re-assess

40–59 REVIEW
security board escalation

0–39 REJECTED
must not be used

USE-CASE FOCUS

GenAI chatbot
content filtering · prompt injection · data leakage · PII

RAG system
data access controls · source attribution · embedding security

Agentic AI
action authorization · scope limits · HITL · audit · rollback

↗ aiappsecacademy.com/downloads/AI_AppSec_Academy_LLM_Security_Assessment.xlsx

Securing Desktop Agents: Inventory, Score, Scan

Inventory
every agent, MCP server, skill & key

→

Score
OWASP AIVSS: CVSS v4 + agentic factors

→

Scan
snyk-agent-scan on MCP servers + skills

→

Enforce
allowlist · managed settings · sandbox

→

Monitor
MDM re-scan on a schedule → SIEM

WHAT INVENTORY FINDS ON A DEVELOPER LAPTOP

- Claude Code
- Claude Desktop
- Cursor
- VS Code + Copilot
- Windsurf
- Gemini CLI
- Codex
- Kiro
- MCP configs (stdio + remote)
- SKILL.md skills
- IDE extensions
- API keys on disk
- shadow / unapproved agents

TOOLS & GUARDRAILS

snyk-agent-scan

uvx snyk-agent-scan@latest · prompt injection, private-data exposure, destructive tools, malicious skills, secrets

OWASP AIVSS

CVSS v4 base + 10 agentic amplification factors (autonomy, tools, non-determinism) → prioritize fixes

Endpoint guardrails

MDM allowlist of agents · managed settings + deny rules · sandbox + egress allowlist · short-lived keys

↗ github.com/snyk/agent-scan · aivss.owasp.org/calculator

Where to Start: 30 / 60 / 90 Days

DAYS 0–30 **SEE IT**	DAYS 31–60 **CONTAIN IT**	DAYS 61–90 **PROVE IT**
Inventory AI: agents, MCP servers, skills, vendor models	Scan skills + OWASP rules in coding agents	Hash + sign deps, skills, prompts
Route every model call through a gateway	I/O rails on your riskiest AI app	Canary attack replays, policy-as-code
Central logs, audit trail, spend caps	MCP gateway with per-agent allowlists	Rehearse the kill switch
Assess your top vendor models	Sandbox agent code + deny-by-default egress	Track fix acceptance, FP rate, MTTR

Visibility first — you can't contain what you can't see.

Take It Home: Open-Source Toolkit

1. **MLOps Security Policy**
   goals + techniques for data, model, platform, compliance, human
   github.com/ThalesGroup/secure-ml

2. **AI / LLM Security Assessment**
   provider + model questionnaires, auto-scored
   aiappsecacademy.com → downloads

3. **Product Security AI Skills**
   SAST, secrets, SCA, DAST skills for coding agents
   github.com/vchirrav-eng/product-security-ai-skills

4. **OWASP Secure Coding Rules**
   22 markdown rule sets for agentic coding
   github.com/vchirrav-eng/owasp-secure-coding-md

5. **WAF Rules**
   AI-generated sensitive-data-leak rules
   github.com/vchirrav-eng/waf-rules

6. **ThreatModelingGPT**
   STRIDE · PASTA · LINDDUN · MAESTRO
   chatgpt.com → ThreatModelingGPT

Fork them, adapt them to your stack, and send pull requests.

Thank You · Questions?

The future is secure intelligence.



linkedin.com/in/vchirrav · aiappsecacademy.com · github.com/vchirrav-eng