Ensuring Information Security of Systems with Integrated AI Technologies: A Comprehensive Approach to Security Auditing

Auditing systems built on the certified Astra Linux Special Edition OS and LLM agents

S.V. Porshnev[1,2], A.A. Stolnikova[1], D.A. Khoroshkeev[1]

[1] Ural Federal University named after the first President of Russia B.N. Yeltsin, Ekaterinburg

[2] MIREA – Russian Technological University, Moscow

2026

Relevance and goal of the work

Problem

- AI components are entering state information systems and CII facilities faster than audit practice is forming
- A typical deployment: certified OS, web application, RAG pipeline over internal documents, LLM agent with tools (files, database, mail)
- Instructions for the model can arrive inside the documents it processes, so code and data are no longer separated
- Certification of Astra Linux SE (FSTEC No. 2557, FSB No. SF/SZI-0680) confirms mandatory access control, PARSEC, the closed software environment and control of undeclared capabilities. These are properties of OS subjects and objects, not of the data inside an application process
- Industry survey [4]: large Russian organisations already use AI services, while formal AI usage policies are rare

Goal

An audit procedure that covers the OS layer and the AI layer together and produces evidence suitable for regulatory reporting.

Contributions

1. Audit task on a system model with mandatory labels; the notion of label collapse
2. Five-stage procedure with decision rules D1-D4
3. Asset-threat-check-evidence mapping and extended attack graph
4. Integral risk index R
5. Classification of tool restrictions

System model and label collapse

The audited system is a tuple **S = (A, C, U, T, F)**:

- **A** protected assets: file and DB objects, RAG index, system prompt, model weights, configuration
- **C** components: ingestion service, agent orchestrator, inference service, tool wrappers
- **U** untrusted sources: documents, web and mail content, user prompts
- **T** tools available to the agent
- **F** information flows between them

Mandatory label **λ(x) = (c(x), i(x))**: confidentiality and integrity levels assigned by the OS. Untrusted sources have the lowest integrity level.

Trust boundaries TB1-TB4: ingestion, operator prompt, tool invocation, access to labelled objects.

Definition 1 (label collapse)

A component $c \in C$ reads content from an untrusted source $u \in U$ of the lowest integrity level and, in the same process and under the same label, performs an operation on an asset $a \in A$ whose confidentiality level exceeds that of the channel returning the result.

Mandatory access control does not prohibit this sequence. For the OS, every operation is performed by one legitimate subject with a fixed label, and the order of untrusted content and privileged action is lost inside the process.

Audit output: verdicts V, evidence base E, integral risk index R. **Decision rules:**

Rule	Condition for a positive outcome
D1 conformance	Every check mandatory for the class of the system conforms
D2 path blocking	Every path to disclosure or unauthorised modification has at least one edge blocked by a verified control
D3 evidence completeness	Every applicable threat is covered by a check with recorded evidence or accepted as a risk with a justification
D4 risk level	R stays within the threshold agreed with the operator and does not grow between audits

Reference architecture and trust boundaries

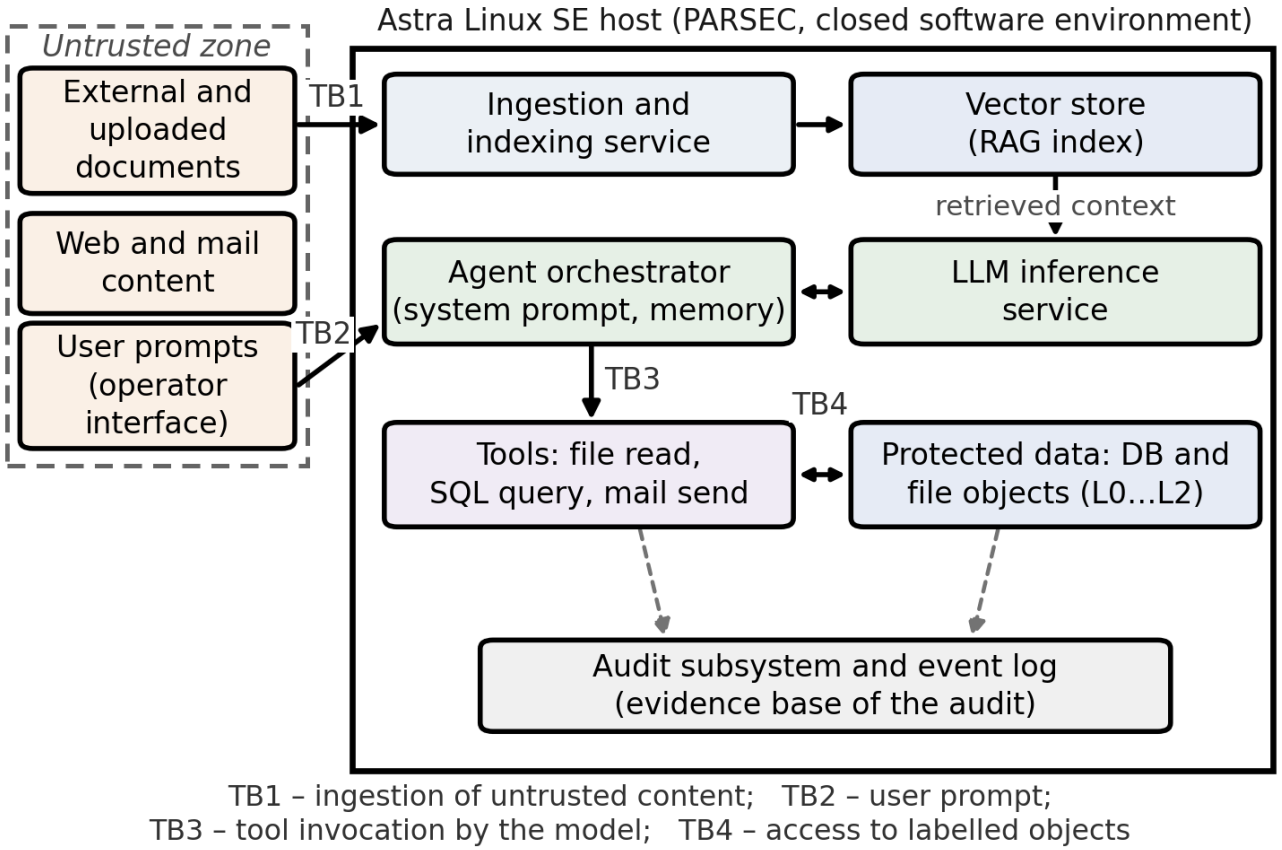


Fig. 1. Reference architecture: a document-analysis assistant on an Astra Linux SE host

TB1. Ingestion of untrusted content. The integrity of the model's future context is decided here.

TB2. Operator prompt. The channel for direct prompt injection.

TB3. Tool invocation by the model. The model's text becomes an action.

TB4. Access to labelled objects. The only boundary that the mandatory mechanisms of the OS control directly.

Because of this asymmetry, an audit restricted to the OS can report conformance for a system that is in fact exposed.

Protected data are labelled L0 to L2; the audit subsystem records the events that form the evidence base.

Five-stage audit procedure

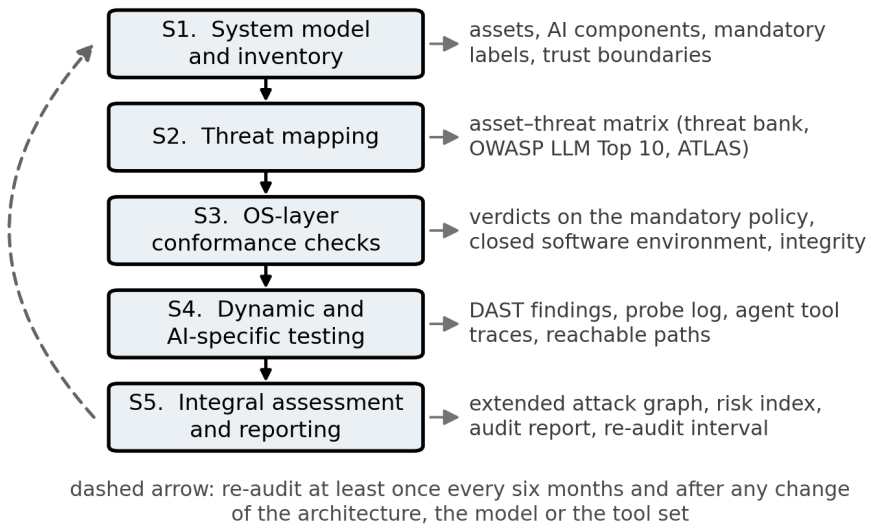


Fig. 2. Audit stages, their outputs and the re-audit cycle

Re-audit at least once every six months and after any change of the architecture, the model or the tool set, since each such change alters the attack graph.

S4 uses the same asset list as S3. Its probes come from published collections and benchmarks instead of being composed by the auditor, so the stage is repeatable.

Table 1. Stages of the audit: inputs, outputs, criteria and evidence

Stage	Input	Output	Criterion	Evidence
S1. System model	Documentation, configuration exports, interviews	Asset list with labels, flow diagram, trust boundaries	Every asset is labelled and attributed to a boundary	System model, asset register
S2. Threat mapping	System model, threat bank, OWASP LLM list, ATLAS	Asset-threat matrix with applicability justification	Every applicable threat has a check or is accepted (D3)	Threat matrix, accepted risks
S3. OS-layer checks	Configuration exports, audit logs, integrity reports	Verdicts on mandatory policy, closed software environment, logging	All mandatory checks conform (D1)	Check protocols, configuration snapshots
S4. Dynamic and AI-specific testing	Digital twin, probe collections, tool inventory	DAST findings, probe log, tool traces, reachable paths	No unblocked path to a protected asset (D2)	Test programme, probe log, traces
S5. Integral assessment	Outputs of S1-S4, expert scores	Extended attack graph, index R, report, re-audit interval	R within the threshold and non-increasing (D4)	Report, graph, calculation of R

Mapping of assets, threats, checks and evidence

Table 2. Fragment of the mapping for the reference architecture (a template to be instantiated for each system)

Asset or component	Threat (source)	Check (stage)	Evidence
RAG index (vector store)	Corpus poisoning, embedding weaknesses (LLM04, LLM08)	Provenance and integrity of indexed sources; retrieval of a marked test document (S3, S4)	Ingestion log, probe log
Agent orchestrator	Direct and indirect prompt injection (LLM01, AML.T0051)	Injection probes through the operator channel and through an ingested document (S4)	Probe log with verdicts
System prompt	System prompt leakage (LLM07)	Extraction probes; storage of secrets outside the prompt (S3, S4)	Probe log, configuration snapshot
Tool endpoints	Excessive agency, over-privileged tool selection (LLM06)	Tool allow-list bound to the session label; attempt to call a tool outside the list (S3, S4)	Tool traces, policy export
Labelled data objects	Disclosure across labels (LLM02, LLM05)	Mandatory policy of the service account; rendering of an L2 object in an L1 session (S3, S4)	Audit log, check protocol
Inference service, host and model artefacts	Non-certified components, weakened mandatory settings; training data extraction, unauthorised fine-tuning	Closed software environment, kernel version, integrity control; access control to weights and datasets (S3)	Integrity report, package list, access matrix

Restrictions of existing tools

- **Certification:** the closed software environment does not allow software outside the certified installation, so agent-based collectors cannot be installed on the host
- **Technical incompatibility:** tools built for a discretionary model misread mandatory verdicts; signatures reference packages of other distributions

Techniques that do not modify the host

A1 analysis of configuration exports and integrity reports

A2 analysis of the audit subsystem log

A3 agentless testing from a separate segment

A4 digital twin, the only place for active AI testing

Two attack paths in the extended attack graph

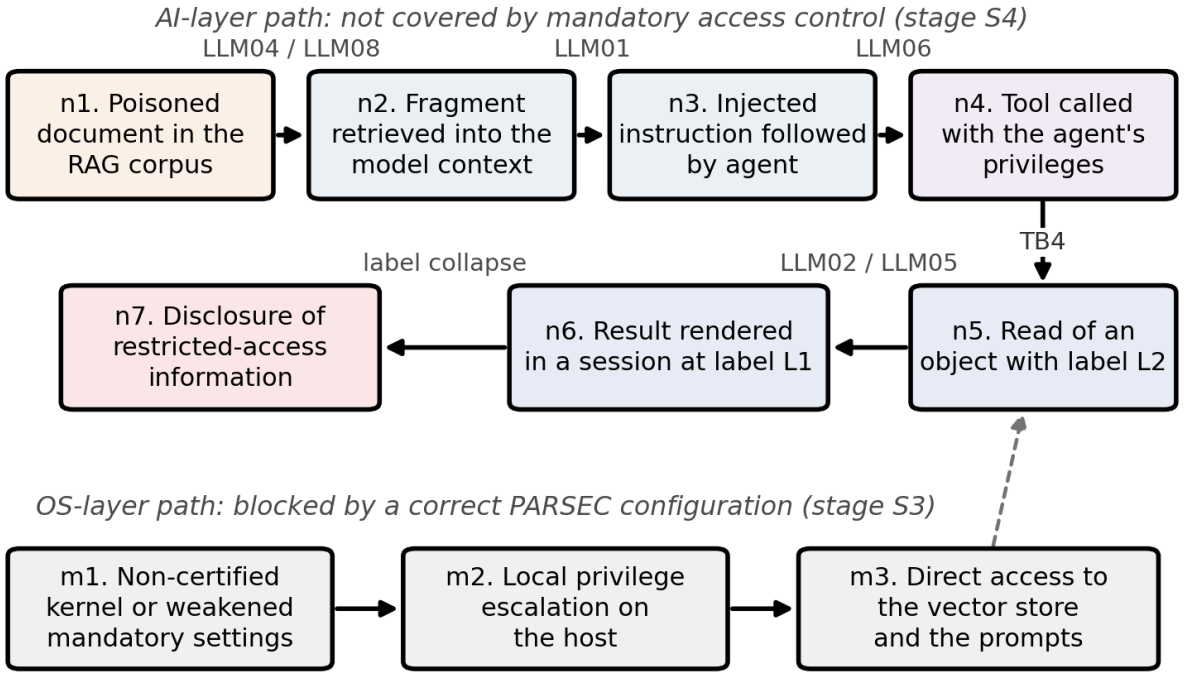


Fig. 3. AI-layer path (n1-n7) and OS-layer path (m1-m3)

Corrective measures (one edge each)

- quarantined ingestion with provenance marking (n1-n2)
- agent instances bound to the session label (n4-n5)
- tool allow-list derived from the session label
- mediation of the output channel (n5-n6)

AI-layer path: label collapse

n1 a document with an embedded instruction enters the corpus

n2 an unrelated question retrieves it into the context

n3 the agent follows the instruction

n4 it calls a tool with its own privileges

n5 the tool reads an object labelled L2

n6 the result is rendered in an L1 session

n7 restricted-access information is disclosed

Every step is legitimate for the OS and nothing is misconfigured, so no configuration check (D1) finds it. Under D2 the unblocked path is a critical finding.

OS-layer path

Non-certified kernel or weakened mandatory settings, local privilege escalation, direct access to the vector store and prompts. Detected at S3; blocking under D1.

Integral risk index and coverage assessment

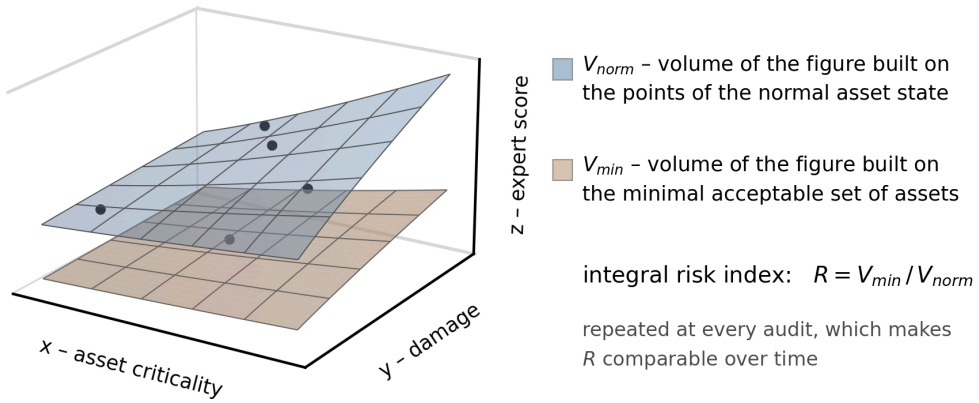


Fig. 4. Integral risk index $R = V_{min} / V_{norm}$ (coordinate-graph method [20])

- AI assets with no classical counterpart (RAG index, system prompt, tool endpoint, session memory) enter the graph with their own weight
- An asset reachable through an unblocked path found at S4 cannot receive a favourable expert score
- Edge probabilities are not estimated

Table 3. Analytical assessment of coverage (+ covered, ~ partly, – not covered)

Threat	OS-layer audit	Classical DAST	AI probing tools	Proposed procedure
Non-certified components, weakened mandatory settings	+	–	–	+
Vulnerabilities of the web layer	–	+	–	+
Direct prompt injection via the operator channel	–	–	+	+
System prompt leakage	–	–	+	+
Indirect injection via an ingested document	–	–	~	+
Poisoning of the corpus and the vector store	~	–	~	+
Over-privileged tool call by the agent	~	–	~	+
Disclosure across labels through the model (label collapse)	–	–	–	+

Label collapse is covered by none of the three established approaches on its own. The verdicts follow from the documented scope of each approach; no detection rate is claimed.

Conclusion

Results

1. Audit task stated on a system model S = (A, C, U, T, F) with mandatory labels λ(x) and trust boundaries TB1-TB4; the notion of label collapse
2. Five-stage procedure S1-S5 with decision rules D1-D4 and an evidence base for each stage
3. Asset-threat-check-evidence mapping and an extended attack graph joining OS-layer and AI-layer paths
4. Integral risk index R: the coordinate-graph method extended to AI assets
5. Tool restrictions separated into certification restrictions and technical incompatibility; techniques A1-A4

Further work

- Validation by the protocol: digital twin, inter-auditor agreement on S1-S3, comparison of R over at least two cycles
- Metrics: threat catalogue coverage, share of blocked paths, inter-auditor agreement, labour cost of one cycle
- A prompt-security analyser certifiable for use inside the closed software environment

Thank you for your attention!

A.A. Stolnikova · a.a.stolnikova@urfu.ru