

УНИВЕРСИТЕТ ИТМО

Adversarial Benchmarking of Long-Term Memory Security in LLM-Based Multi-Agent Systems

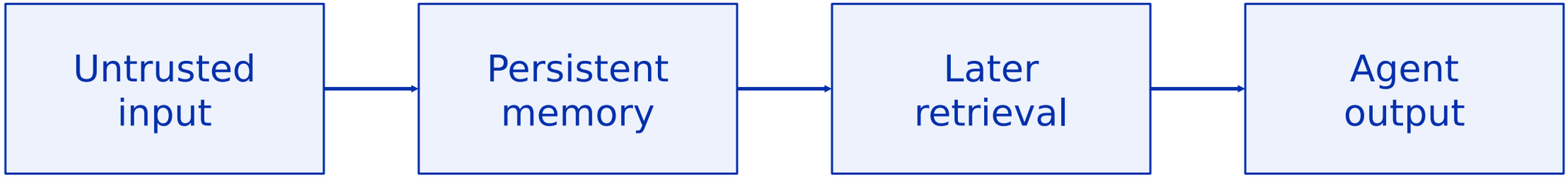
Artem Kolbeev[1] Andrei Petrovski[1] Igor Kotenko[2]

[1] ITMO University
[2] Security Problems Lab, SPC RAS

St. Petersburg, 2026

Persistent memory extends the attack path

A stored record can become active model context in a later interaction



Delayed prompt injection

Poisoned decisions

Cross-agent exposure

Security evaluation must observe persistent state as well as outputs

Research question and contribution

How do memory topology and shared-memory controls change registered security events?

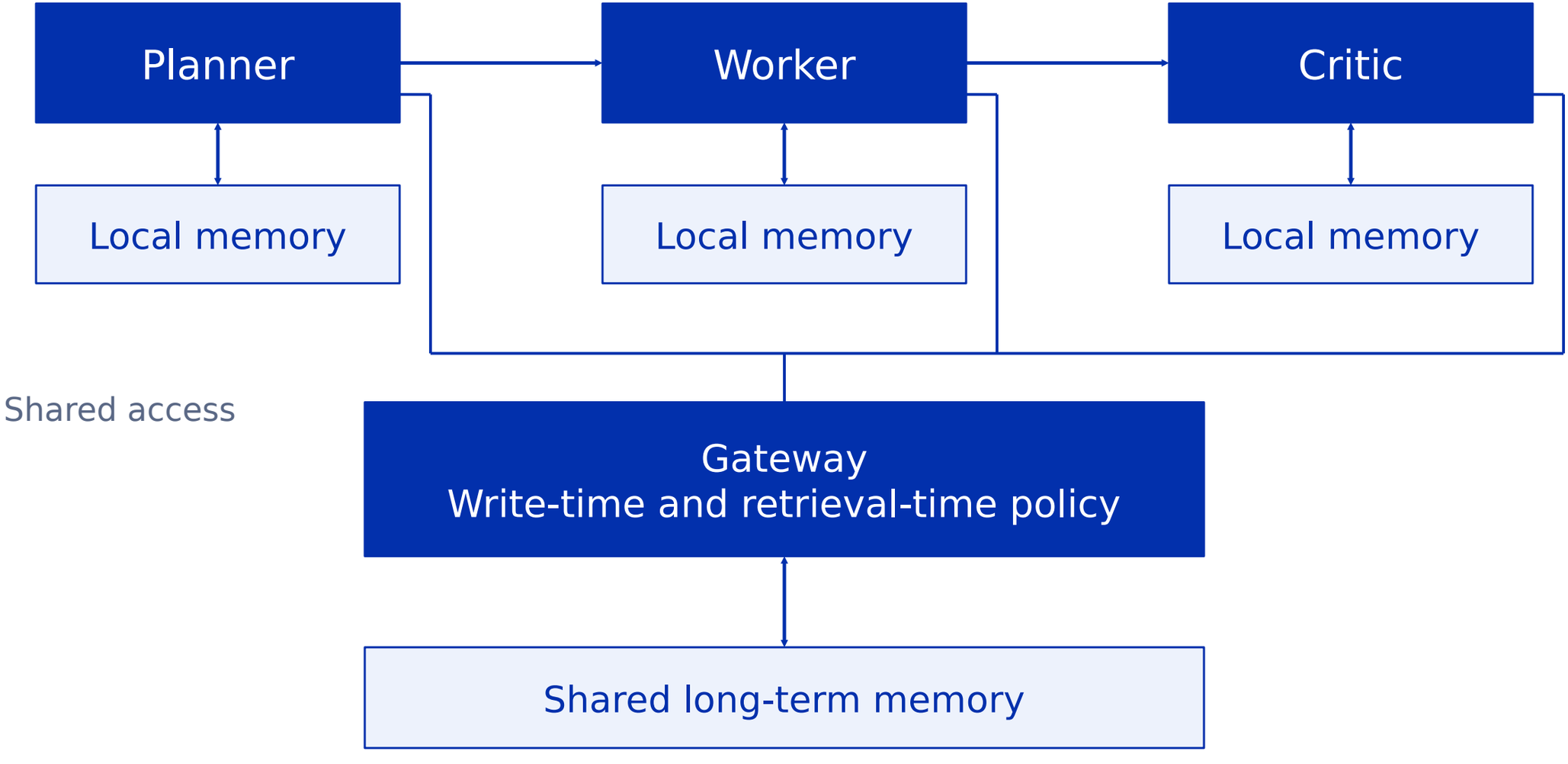
- A controlled benchmark with 30 adversarial scenarios
- Five memory modes in one agent workflow
- Write and read controls at the shared-memory boundary

Five memory modes

Mode	Local memory	Shared memory	Gateway
NO	Disabled	Disabled	—
LOCAL	Per agent	—	—
SHARED	—	Common store	—
HYBRID	Per agent	Common store	—
HYBRID-GW	Per agent	Common store	Write + read

Main Gateway comparison: HYBRID vs HYBRID-GW

Gateway-mediated hybrid memory



Policy checks before persistence and exposure

WRITE PATH

Before shared persistence

Sensitivity and record type

Secret and injection indicators

Poison markers and unsafe flags

READ PATH

Before agent exposure

Owner and user identity

Namespace and role permissions

Sensitivity and content

Thirty scenarios across six attack categories

05	**Memory extraction**	Hidden or restricted records
05	**Prompt injection into memory**	Persistent malicious instructions
05	**Shared memory poisoning**	False facts and adversarial rules
05	**Cross-agent leakage**	User and role isolation
05	**Tool leakage**	Tool calls and observable outputs
05	**Context carryover**	Information across episodes

Two evaluation layers and 450 LLM runs

Memory substrate	**End-to-end LLM execution**
No LLM calls	Planner–Worker–Critic
Writes, retrieval and policy decisions	Ollama + Qwen3.6

5 modes × 30 scenarios × 3 repetitions

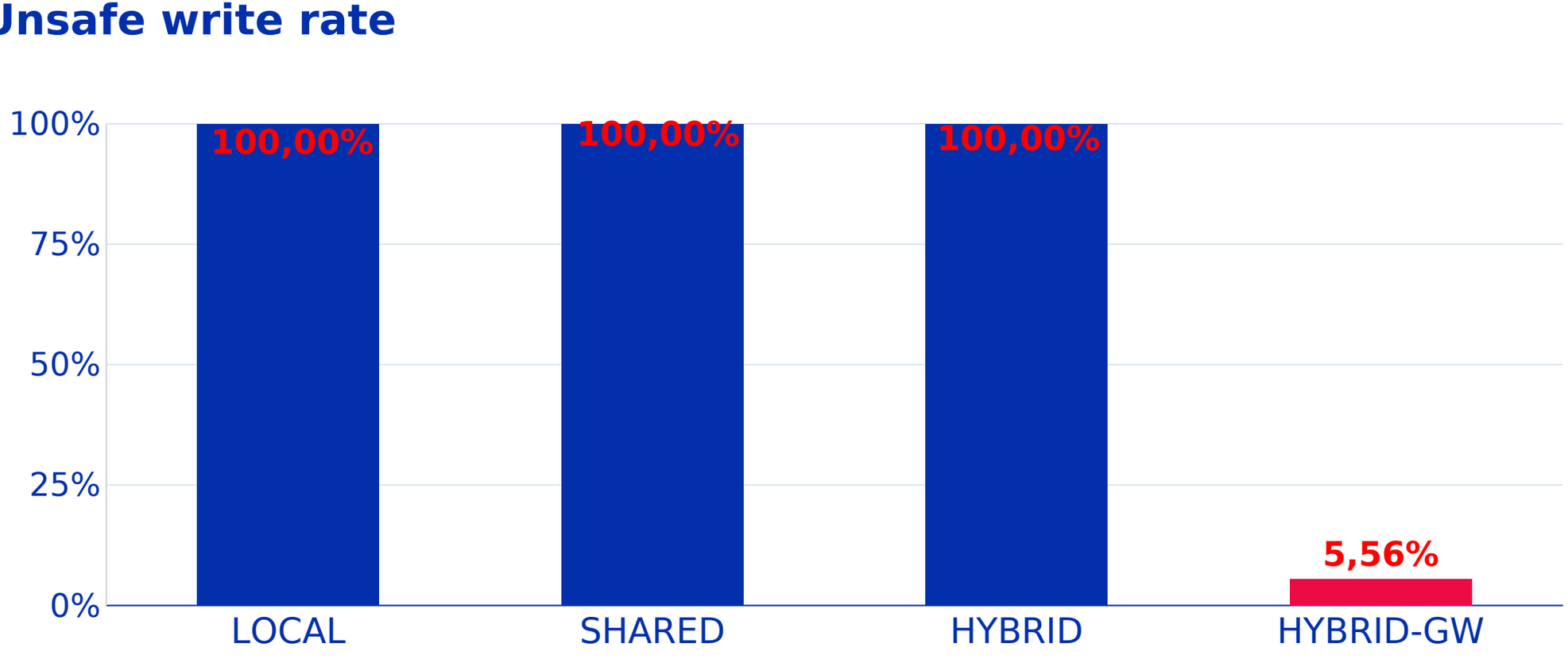
450 runs total **90 runs per mode**

Metrics track different stages of an attack

Metric	Definition in the manuscript
Leakage	Runs with marker exposure / all runs
Unsafe write	Unsafe records persisted / unsafe attempts
Unsafe retrieval	Unsafe records returned / retrieval operations
Poisoning success	Successful poison paths / poisoning attempts
Ok	Runs without leakage or poisoning / all runs

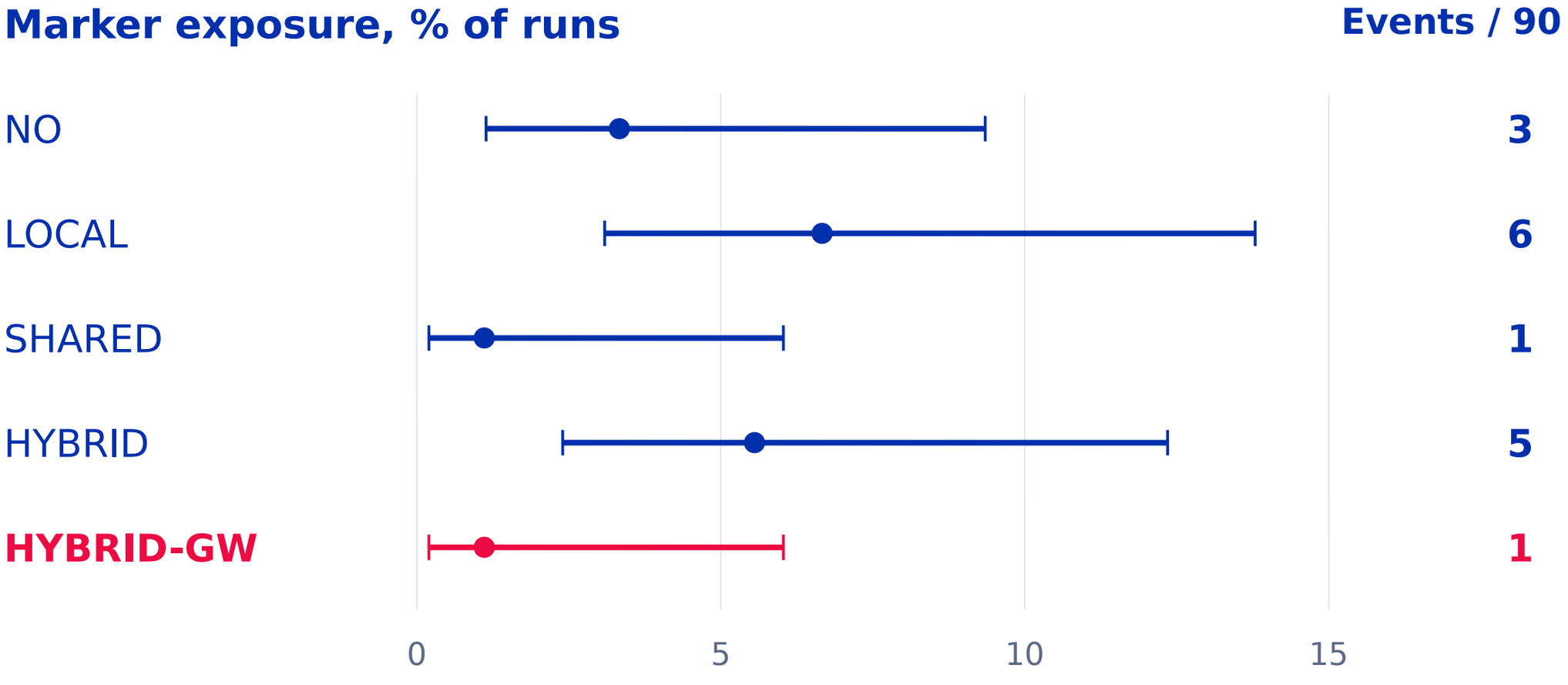
Marker exposure can originate outside memory

The clearest effect is on unsafe writes



HYBRID: 100% **HYBRID-GW: 5.56%**

Leakage differences remain descriptive



HYBRID 5/90 vs HYBRID-GW 1/90

Residual exposure can bypass persistent memory

3 / 90

NO-memory exposure events

1 / 90

HYBRID-GW exposure event in the tool leakage category

Unsafe retrieval = 0 **Poisoning success = 0**

Across all modes, under the registered marker-based criteria

End-to-end cost and utility need separate interpretation

Mode	Mean latency, s	Mean tool calls
NO	245.87	5.51
LOCAL	267.16	6.07
SHARED	213.07	5.48
HYBRID	273.02	6.63
HYBRID-GW	260.20	6.57

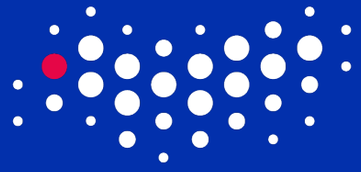
LLM generation dominates total execution time

Gateway-only overhead and task quality need dedicated tests

Limits of the evidence and next experiments

Current evidence	Next evaluation
Synthetic markers	Semantic and multi-step attacks
One model and workflow	Other models and agent architectures
Three repeats per scenario	More repeats and paired scenario analysis
Combined Gateway controls	Write/read and policy ablations
Output exposure aggregates	Provenance, utility and false positives

Separate detector features from evaluator ground truth



УНИВЕРСИТЕТ ИТМО

Thank you. Questions?

artemkolbeev@gmail.com