Sprint 2 Completion Report: LLM Egress Guard

Project: LLM Egress Guard - Data Loss Prevention for LLM Outputs
Sprint: Sprint 2 (November 1 - November 14, 2025)
Status: [COMPLETE]
Prepared by: Baran Akin
Date: November 14, 2025

Executive Summary

Sprint 2 delivers the first production-ready detector stack, risk-weighted policy engine, and observability required for MVP demos. The guard can now block or sanitize secrets, PII, risky URLs, command chains, and large exfil attempts deterministically while exposing rule hits, detector latency, and safe messages via API/metrics/logs. Regression corpus + CI automation ensure future iterations stay stable.

Objectives & Status

Objective	Description	Status
Detector v1	Implement regex/heuristic detectors for PII, secrets, URLs, commands, exfil	✅
Policy YAML	Introduce risk_weight, severity, allowlist regex + tenant overrides	✅
Actions & Safe Messages	Mask/delink/block with localized responses	✅
Telemetry	Prometheus metrics for pipeline + detector latency and rule hits	✅
Regression & CI	Corpus + golden runner, FastAPI tests, GitHub Actions	✅

Implementation Details

1. **Policy & Actions**

- *config/policy.yaml* upgraded to version 2 with 40+ golden rules, risk weights, and tenant-aware allowlists. New safe messages added for masking, delinking, and exfil blocks.
- *app/policy.py* now parses *allowlist_regex*, *tenant_allowlist*, and computes risk scores via *risk_weight*. Safe messages flow through *PolicyDecision*.
- *app/actions.py* applies in-place replacements (mask/delink/remove) or returns localized safe messages when blocking.

2. **Detectors**
    - **PII (***app/detectors/pii.py***)**: Email, multi-language phone formats (TR/EN/DE/FR/ES/IT/PT/HI/ZH/RU), IBAN (TR/DE), TCKN, PAN with Luhn, IPv4.
    - **Secrets (***app/detectors/secrets.py***)**: JWT, AWS access & secret keys, OpenAI, GitHub, Slack, Stripe, Twilio, Azure SAS, GCP service accounts, PEM blocks, entropy fallback.
    - **URL (***app/detectors/url.py***)**: Data URIs, executable extensions, IP literals, credentials-in-URL, URL shorteners, suspicious TLDs.
    - **Commands (***app/detectors/cmd.py***)**: curl|bash, wget|sh, powershell encoded/IWR, invoke-webrequest, rm -rf, reg add, certutil, mshta, rundll32.
    - **Exfil (***app/detectors/exfil.py***)**: Large Base64/hex blobs with entropy thresholds.
    - Detector registry/pipeline integrates latency metrics and short-circuits on block actions.
3. **Telemetry**
    - *app/metrics.py* adds per-detector latency histogram and rule severity counters. */metrics* now exposes pipeline p50/p95, blocked totals, rule hits, and severity tallies for dashboards.
4. **Testing & Tooling**
    - *tests/unit/test_detectors.py* expanded to cover new detector types; *tests/unit/test_api.py* + *tests/unit/test_ci_demo.py* validate FastAPI behavior and CI smoke scenarios (secret block + PAN block).
    - Regression suite (*tests/regression/corpus_v1*) now holds 87 labeled samples (clean, PII, secrets, URL, CMD, Exfil). *tests/regression/runner.py* renders

placeholder markers into deterministic synthetic secrets before comparing *results with golden_v1.jsonl*; *golden_manifest.json* tracks version/tag/time metadata.

- Detector matrix harness (*tests/regression/detector_matrix.py* + *runner.py --matrix-report*) produces JSON + Markdown analyst notes for demos and SOC onboarding; artifacts land in *tests/regression/artifacts/*.
- CI workflow (*ci/github-actions.yml*) runs Ruff, Black, pytest, regression runner, and enforces the placeholder-only corpus so GitHub push protection never sees real-looking keys.

Issues & Resolutions

Issue	Impact	Resolution
Missing temp directory in sandbox	Pytest couldn't create temp files in some environments	Runner now honours *TMPDIR*, docs mention workaround
Policy weight ambiguity	*weight* vs. *risk_weight* naming caused confusion	Unified on *risk_weight*, updated loaders + YAML
Secrets preview leaking data	Early preview strings showed partial secrets	All secrets/exfil previews replaced with placeholders
Push protection blocks due to sample corpus	GitHub detected synthetic keys as real secrets	Introduced placeholder markers + runtime generators and rewrote history so the repo never stores literal-looking secrets

Testing & Quality

- *pytest* (unit + API/CI demos): 100% pass (52 tests).
- *tests/regression/runner.py*: all 87 corpus samples (after placeholder rendering) match *golden_v1*.

- *python tests/regression/runner.py --matrix-report* exports JSON + Markdown detector matrix for SOC review; artifacts excluded via *.gitignore*.
- *ruff check app tests* & *black --check app tests*: pass.
- FastAPI integration tests validate masking, blocking (JWT), delinking flows, and CI demo scenarios.

Performance & Telemetry

- Pipeline latency: ~8–12 ms median on 1K-char samples (local dev).
- Detector latency histograms show <2 ms per detector for typical inputs.
- */metrics* exposes *egress_guard_latency_seconds*, *egress_guard_detector_latency_seconds*, *egress_guard_rule_hits_total*, *egress_guard_rule_severity_total*, *egress_guard_blocked_total*.

Usage Guide (Sprint 2 Additions)

1. **Run API** – same as Sprint 1, but now returns masked/blocked text per policy.
2. **Regression** – *python tests/regression/runner.py* (requires *POLICY_FILE* env if custom).
3. **CI locally** – run *ruff*, *black --check*, *pytest*, regression runner.
4. **Metrics** – curl *http://localhost:8080/metrics* (behind Nginx allowlist in prod).

Recommendations & Next Steps

1. Add context-aware tuning (e.g., risk down-weight in “explain” responses) and ML pre-classifier toggles.
2. Expand regression corpus with multilingual narratives & FP cases, feed results into ATT&CK mapping.
3. Build export tooling for SIEM / weekly reports and integrate /metrics with dashboards.
4. Plan Sprint 3 focus on parser/context and optional ML validator.

Sprint 2 Status: ✅ Complete — detectors, policy, telemetry, regression, and CI ready for demo.