140509_43.md â€” AI Model Security and Protection Platform

> **Theme:** AI for CyberSecurity & CyberSecurity for AI, Training Data Confidentiality, Containerization & Isolation
> **Mission:** Safeguard AI models against adversarial attacks, data poisoning, extraction, and unauthorized access with real-time detection, robust defenses, watermarking, lineage, and secure serving.

README (Problem Statement)

Summary: Build a comprehensive platform that protects AI models from adversarial attacks, data poisoning, and unauthorized access while ensuring model integrity.
Problem Statement: AI models face threats including adversarial attacks, model extraction, and poisoning. Create a platform that implements adversarial defense, model watermarking, and access control. The system should detect attacks in real-time, provide integrity verification, and enable secure deployment.

Steps: adversarial detection/defense; watermarking; secure serving; lineage & poisoning detection; tampering detection; secure training.

Suggested Data: adversarial samples; watermarking validation sets; authentication logs; audit & compliance requirements.

---

1) Vision, Scope, KPIs

Vision: Deliver trusted AI deployments with provable integrity and resilience against malicious actors.
Scope:
- v1: secure model serving, adversarial detection, watermark verification, lineage.
- v2: poisoning detection, advanced adversarial defenses, federated secure training.
- v3: red-team testing suite, compliance dashboards, continuous monitoring.

KPIs:
- Block â‰¥95% known adversarial patterns.
- Poisoning detection recall â‰¥0.9 @ FPR â‰¤0.05.
- Watermark verification â‰¥98% success.
- Serving latency overhead â‰¤15%.

---

2) Personas & User Stories

- **ML Engineer:** â€œI want to deploy models securely without worrying about adversarial exploits.â€
- **Security Officer:** â€œI need continuous monitoring and audit logs for compliance.â€
- **Researcher:** â€œI want watermarking to prove model ownership.â€
- **CISO:** â€œI want guarantees of integrity before using AI outputs in critical workflows.â€

User Stories:
- US-01: â€œAs an ML engineer, I want adversarial detection wrapping my model service.â€
- US-05: â€œAs a researcher, I want to insert and later validate watermarks.â€
- US-09: â€œAs a CISO, I want dashboard metrics on model integrity.â€

---

3) PRD

Capabilities:
1. **Adversarial Detection:** entropy checks, Mahalanobis distance, autoencoder reconstruction error.
2. **Defense Mechanisms:** randomized smoothing, feature denoisers, adversarial training.

3. **Watermarking:** black-box (trigger set) and white-box (weight perturbations).
4. **Secure Serving:** RBAC, payload inspection, rate limiting, encrypted transport.
5. **Lineage:** signed checkpoints, dataset fingerprinting, provenance ledger.
6. **Poisoning Detection:** influence functions, gradient anomaly detection.
7. **Monitoring:** canary inputs, extraction heuristics, integrity checks.
8. **Training Security:** isolated containers, seccomp/AppArmor sandboxing.

---

4) FRD

- **Ingress Gateway:** TLS1.3, JWT/OIDC authentication, payload inspector.
- **Defense Layer:** ensemble detectors wrapping inference requests.
- **Watermark Module:** API `POST /watermark/verify`.
- **Lineage Ledger:** blockchain-style append-only store for model/data signatures.
- **Poison Scan:** retraining-time module analyzing label distributions, gradients.
- **Monitor:** metrics pushed to SIEM/Splunk.

---

5) NFRD

- **Latency:** additional inference cost â‰¤15%.
- **Scale:** 1k RPS per model, horizontal scaling.
- **Availability:** 99.9%.
- **Compliance:** SOC2, ISO27001, HIPAA.
- **Audit:** immutable logs, 7-year retention.

---

6) Architecture (Logical)

```
[Clients] -> [API Gateway/AuthZ] -> [Payload Inspector] -> [Defense Layer] -> [Model Serving]
                                  |-> [Watermark Service]
                                  |-> [Lineage Ledger]
                                  |-> [Poison Detector]
                                  |-> [Monitoring/SIEM]
```

---

7) HLD

- **Gateway:** Envoy + OPA for policy.
- **Defense:** ONNXRuntime wrappers calling detection models.
- **Watermark:** trigger-set queries; white-box watermark verifier.
- **Lineage:** append-only ledger (Hyperledger Fabric or immudb).
- **Training Isolation:** Kubernetes pods w/ seccomp.

---

8) LLD Examples

Adversarial Score:
- Features: softmax entropy, Mahalanobis distance, AE reconstruction error.
- Thresholds: score > Ï„ â†’ adversarial.

Watermark Verification:
- Input trigger set X.
- Prediction pattern Y.
- Compare vs expected signature.

Poison Detection:
- Influence function outliers.
- Gradient cosine similarity checks.

---

9) Pseudocode

```
function secure_infer(request):
  if not verify_signature(request): reject()
  if payload_inspector.blocks(request): deny()
  adv_score = defense_ensemble(request.input)
  if adv_score > Ï„: return safe_response()
  y = model(request.input)
  if watermark_enabled: watermark_verify(y)
  log_lineage(y, request.meta)
  return y
```

---

10) Data & Evaluation

- **Data:** ImageNet-C, CIFAR-adv, TrojAI, watermark datasets.
- **Eval Metrics:** robust accuracy, AUC of adversarial detection, watermark verification power, poisoning detection recall.
- **Validation:** red-team attack sims (FGSM, PGD, DeepFool, Trojan triggers).

---

11) Security & Governance

- RBAC + ABAC.
- All payloads logged, anonymized.
- Immutable lineage for audits.
- Compliance mapping to NIST SP800-53.

---

12) Observability & Cost

- Metrics: % blocked queries, detection latency, watermark integrity rate.
- Cost: defense models only on suspicious payloads.

---

13) Roadmap

- **M1 (4w):** Secure serving + watermark verify.
- **M2 (8w):** Poison detection + adv training.
- **M3 (12w):** Extraction monitoring + ledger.
- **M4 (16w):** Federated secure training + red-team suite.

---

14) Risks & Mitigations

- **False blocks:** allow human override.
- **Latency hit:** selective routing to defense models.
- **Watermark removal attacks:** hybrid watermarks (black+white box).
- **Insider threats:** RBAC + audit logs.