Miner Capabilities and Validation Strategies

AML Subnet: What Miners Can Do and How to Validate It

Date: 2025-10-25
Context: Analyzing SOT data schemas to define miner value propositions

---

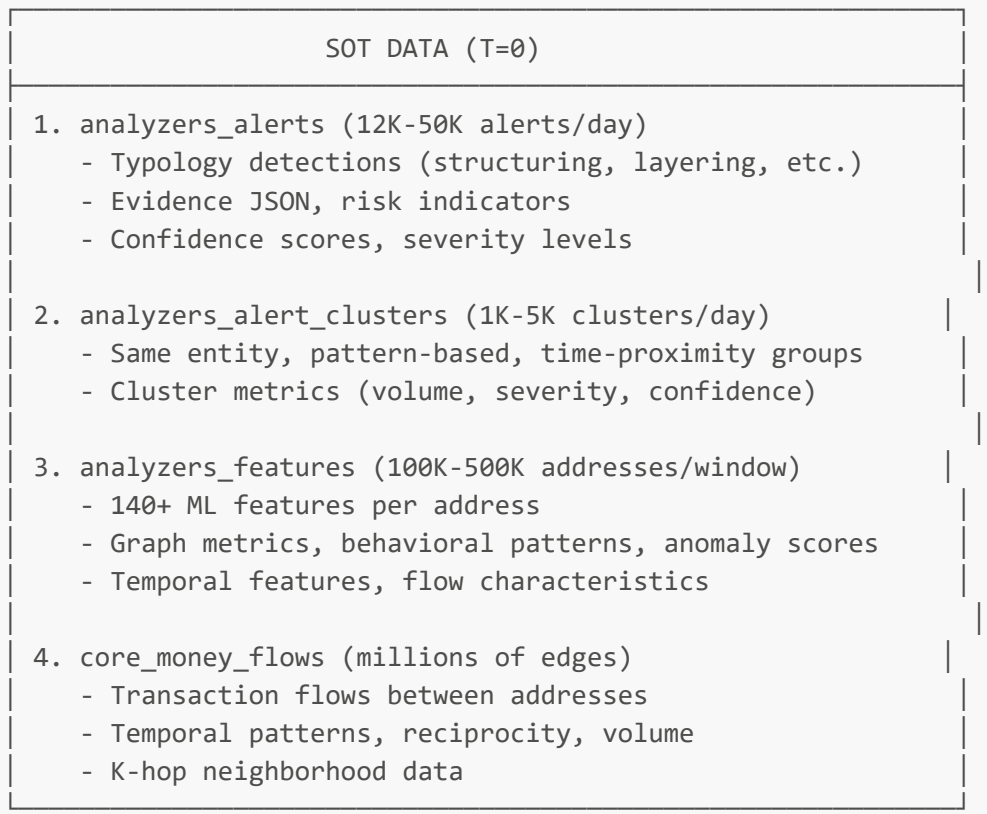
Executive Summary

Based on the SOT data schemas (`analyzers_alerts`, `analyzers_alert_clusters`, `analyzers_features`, `core_money_flows`), miners can contribute in **5 distinct ways**, each with specific validation strategies.

Key Insight: The subnet provides rich, temporal data enabling miners to compete on ML quality, not data infrastructure.

SOT Data Overview

What Validators Provide to Miners (Daily Batch)



What Changes Over Time (T+τ Ground Truth)

```
┌─────────────────────────────────────────────────────────┐
│              GROUND TRUTH (T+τ, τ=7-21 days)            │
├─────────────────────────────────────────────────────────┤
│ - New alerts triggered by addresses                     │
│ - Flow to known sanctioned/mixer/scam addresses         │
│ - Address label discoveries (exchange, validator, etc.) │
│ - Cluster evolution (addresses joining/leaving)         │
│ - Pattern confirmations (structural patterns validated) │
└─────────────────────────────────────────────────────────┘
```

Proposal 1: Alert Risk Scoring (Primary Use Case)

What Miners Do

Input: Single alert from `analyzers_alerts`
Output: Risk score ∈ [0,1] representing P(illicit outcome within τ days)
Computation: ML model using features, flows, cluster context

Model Approaches

1.1 Supervised ML Models

```
# Train on historical alerts with confirmed outcomes
model = LightGBM(
    features=[
        # From analyzers_features
        'behavioral_anomaly_score',
        'graph_anomaly_score',
        'total_volume_usd',
        'degree_total',
        # From alert
        'alert_confidence_score',
        'typology_type',
        # From cluster
        'cluster_severity_max',
        'cluster_confidence_avg'
    ],
    target='confirmed_illicit'  # from T+τ
)
```

Validation Strategy:

```
T+0 Checks (Immediate):
├─ Range validation: score ∈ [0,1]
├─ Variance check: std(scores) > threshold
├─ Monotonicity: high-severity alerts → higher median scores
├─ Pattern traps: embedded canary alerts with known outcomes
```

```
└─ Consistency: correlation with previous day's model ≥ 0.85

T+τ Settlement (21 days later):
├─ Brier Score: mean((predicted - actual)²)
├─ Log Loss: -mean(y*log(p) + (1-y)*log(1-p))
├─ Calibration: ECE (Expected Calibration Error)
├─ AUC-ROC: ranking quality
└─ Early warning bonus: exp(-λ * days_to_event)
```

1.2 Ensemble Approaches

- Combine multiple weak learners
- Stack typology-specific models
- Use cluster-level features as meta-features

Validation: Same as 1.1, plus diversity bonus for uncorrelated predictions

---

Proposal 2: Alert Prioritization (Ranking)

What Miners Do

Input: Batch of N alerts (10K-50K)
Output: Ranked list by urgency for investigation
Goal: Top K alerts should have highest confirmed illicit rate

Ranking Strategies

2.1 Learning-to-Rank

```
# Pairwise or listwise ranking
ranker = LambdaMART(
    features=alert_features + cluster_features,
    objective='ndcg'  # Normalized Discounted Cumulative Gain
)
```

2.2 Multi-Objective Ranking

```
# Balance multiple criteria
score = w1 * risk_score
      + w2 * severity_weight
      + w3 * cluster_importance
      + w4 * freshness_decay
```

Validation Strategy:

```
T+0 Checks:
├─ Top-K coverage: top 500 alerts cover ≥80% of known bad addresses
├─ Severity alignment: critical alerts appear in top decile
└─ Stability: Kendall-τ correlation with baseline ≥ 0.75

T+τ Settlement:
├─ Precision@K: fraction of top K that confirmed illicit
├─ NDCG@K: quality of ranking order
├─ Recall@K: coverage of illicit alerts in top K
└─ MRR: Mean Reciprocal Rank of first true positive
```

Proposal 3: Cluster Risk Assessment

What Miners Do

Input: Alert cluster from `analyzers_alert_clusters`
Output: Cluster-level risk score considering:

- Aggregate alert severity
- Address relationship patterns
- Temporal evolution signals
- Cross-cluster connections

Cluster Analysis Methods

3.1 Graph-Based Scoring

```
# Analyze cluster as subgraph
cluster_score = f(
    internal_density,           # how connected internally
    external_connectivity,      # links to other clusters
    temporal_burst_factor,      # activity concentration
    member_anomaly_scores,      # avg member features
    pattern_consistency         # same pattern across cluster
)
```

3.2 Temporal Cluster Evolution

```
# Track cluster changes over windows
evolution_score = f(
    growth_rate,                # new members joining
    stability,                  # core member retention
    volume_acceleration,        # transaction volume trend
    severity_escalation         # alert severity increase
)
```

Validation Strategy:

```
T+0 Checks:
├─ Cluster coherence: score variance across members ≤ threshold
├─ Size bias check: score not simply proportional to cluster size
└─ Type consistency: pattern clusters scored differently than entity clusters

T+τ Settlement:
├─ Cluster outcome rate: % of clusters with ≥1 confirmed illicit member
├─ Coverage metrics: high-risk clusters contain X% of illicit addresses
├─ False positive rate: low-risk clusters have <Y% illicit
└─ Propagation accuracy: predicted cluster expansion matches actual
```

Proposal 4: Feature Engineering & Enrichment

What Miners Do

Input: Base features from `analyzers_features` + flows
Output: Enhanced feature vectors with:

- Derived features (ratios, combinations)
- Graph embeddings
- Temporal derivatives
- Domain-specific signals

Feature Categories

4.1 Graph Embeddings

```python
# Learn low-dimensional representations
embeddings = Node2Vec(
    graph=money_flows,
    dimensions=64,
    walk_length=10,
    num_walks=20
)
# Use embeddings as additional features
```

4.2 Temporal Features

```python
# Compute derivatives across windows
features_enriched = {
    'volume_7d_vs_30d': volume_7d / volume_30d,
    'degree_acceleration': (degree_7d - degree_30d) / 23,
    'behavior_shift': cosine_distance(pattern_7d, pattern_30d),
```

```
    'anomaly_trend': linear_fit(anomaly_scores_history)
}
```

4.3 Domain-Specific Signals

```
# AML-specific feature engineering
aml_features = {
    'structuring_indicator': count_txs_just_below_10k / total_txs,
    'rapid_movement_score': volume_per_active_hour,
    'mixer_proximity': min_hops_to_known_mixer,
    'exchange_affinity': volume_to_exchanges / total_volume
}
```

Validation Strategy:

```
Feature Quality Metrics:
├─ Information gain: mutual information with target
├─ Redundancy check: correlation with existing features
├─ Stability: consistency across time windows
└─ Coverage: non-null rate, distribution properties

Performance Impact:
├─ Ablation testing: model performance with/without features
├─ Feature importance: SHAP values, permutation importance
└─ Generalization: features work across different typologies

Submission Format:
miners submit both:
1. Enhanced features for addresses
2. Alert scores using those features
Validators compare scores from enhanced vs base features
```

Proposal 5: Anomaly Detection & Pattern Discovery

What Miners Do

Input: Full feature matrix + money flows
Output: Novel anomaly signals or pattern detections not in original alerts

Detection Methods

5.1 Unsupervised Anomaly Detection

```
# Discover new anomaly types
detectors = [
```

```
    IsolationForest(contamination=0.01),
    LocalOutlierFactor(n_neighbors=20),
    DBSCAN(eps=0.3, min_samples=5),
    Autoencoder(hidden_dims=[64,32,16,32,64])
]

# Ensemble predictions
novel_anomalies = voting(detectors, threshold=3/4)
```

5.2 Structural Pattern Mining

```
# Find new graph motifs
patterns = {
    'fan_out_chains': detect_fan_out(flows, min_recipients=10),
    'circular_flows': detect_cycles(flows, min_cycle_length=4),
    'timing_clusters': detect_temporal_bursts(flows, window='1h'),
    'value_laddering': detect_amount_patterns(flows, tolerance=0.05)
}
```

Validation Strategy:

```
Novel Pattern Validation:
├─ Novelty check: pattern not in original alerts
├─ Significance: pattern occurs in <5% of addresses (rare)
├─ Consistency: pattern detected across multiple miners
└─ Actionability: pattern has discriminative power

T+τ Confirmation:
├─ Pattern outcome rate: % of detected patterns confirmed illicit
├─ False discovery rate: novel patterns that were false positives
├─ Precision-Recall: balance of discovery vs accuracy
└─ Incremental value: improvement over baseline alerts

Reward Structure:
├─ Discovery bonus: first miner to find confirmed pattern
├─ Quality multiplier: based on pattern's discriminative power
└─ Consensus requirement: ≥3 miners must confirm pattern
```

Integrated Validation Framework

Multi-Tier Validation Strategy

```
┌─────────────────────────────────────────────────────────────┐
│ Tier 1: Immediate Integrity (T+0, 100% of submissions)      │
├─────────────────────────────────────────────────────────────┤
```

```
│ ✓ Format validation (schema, ranges, cardinality)          │
│ ✓ Hash verification (input data unchanged)                 │
│ ✓ Determinism check (same input → same output)             │
│ ✓ Basic sanity (variance, monotonicity, consistency)       │
│ Weight: 20% of score                                       │
└────────────────────────────────────────────────────────────┘

┌────────────────────────────────────────────────────────────┐
│ Tier 2: Pattern Traps (T+0, embedded in data)              │
├────────────────────────────────────────────────────────────┤
│ ✓ Canary alerts with known outcomes (commit-reveal)        │
│ ✓ Synthetic anomalies (behavioral watermarks)              │
│ ✓ Edge cases (boundary conditions, rare patterns)          │
│ Weight: 30% of score                                       │
└────────────────────────────────────────────────────────────┘

┌────────────────────────────────────────────────────────────┐
│ Tier 3: Ground Truth Settlement (T+τ, τ=7-21 days)         │
├────────────────────────────────────────────────────────────┤
│ ✓ Actual outcomes: alerts → illicit confirmations          │
│ ✓ Probabilistic metrics: Brier, LogLoss, calibration       │
│ ✓ Ranking metrics: NDCG, Precision@K, AUC                  │
│ ✓ Early warning: time-discounted rewards                   │
│ Weight: 50% of score                                       │
└────────────────────────────────────────────────────────────┘
```

Micro-Audit System (Rare, Deep Verification)

```python
# 10% of miner-days, triggered by:
triggers = [
    'model_version_change',     # new model deployed
    'performance_outlier',       # sudden improvement/degradation
    'consistency_violation',     # correlation drop
    'random_sampling'            # baseline audit rate
]

# Micro-audit process:
def micro_audit(miner_id, batch_sample):
    """
    1. Validator sends micro-bundle (1-5% of alerts + traps)
    2. Miner returns scores within SLO (2 min)
    3. Validator runs miner's Docker image on same data
    4. Compare outputs bit-for-bit (must match exactly)
    """
    miner_scores = request_scores(miner_id, batch_sample)
    validator_scores = run_docker_image(miner_image, batch_sample)

    assert np.array_equal(miner_scores, validator_scores), \
        "Audit failed: non-deterministic or forged output"
```

Miner Capability Matrix

Capability	Complexity	Validation Difficulty	Value Proposition	Recommended For
Alert Risk Scoring	Medium	Low	High - Core use case	All miners
Alert Prioritization	Low-Medium	Low	High - Operational value	All miners
Cluster Assessment	High	Medium	Medium - Specialized insight	Advanced miners
Feature Engineering	Very High	High	Medium - Incremental gains	Expert miners
Pattern Discovery	Very High	Very High	Low-Medium - Moonshot potential	Research miners

Example Miner Specialization Strategies

Strategy A: Generalist (Recommended for Most)

```
Focus:
  - Alert risk scoring (Proposal 1)
  - Alert prioritization (Proposal 2)

Approach:
  - Robust gradient boosting models
  - Ensemble of typology-specific models
  - Conservative, well-calibrated predictions

Expected Performance:
  - Top 30% of miners
  - Consistent payouts
  - Low variance
```

Strategy B: Graph Specialist

```
Focus:
  - Cluster assessment (Proposal 3)
  - Graph embeddings (Proposal 4.1)

Approach:
  - GNN-based models
  - Multi-hop neighborhood analysis
  - Temporal graph features
```

```
Expected Performance:
  - Top 20% on cluster-heavy batches
  - Higher variance
  - Premium for specialized expertise
```

Strategy C: Research/Innovation

```
Focus:
  - Pattern discovery (Proposal 5)
  - Novel feature engineering (Proposal 4)

Approach:
  - Unsupervised learning
  - New graph algorithms
  - Cross-typology patterns

Expected Performance:
  - Highly variable
  - Occasional discovery bonuses
  - Long-term reputation building
```

Container & Docker Requirements

Miner Image Specification

```
FROM python:3.11-slim

# Determinism requirements
ENV PYTHONHASHSEED=0
ENV NUMEXPR_MAX_THREADS=4
ENV OMP_NUM_THREADS=4
ENV MKL_NUM_THREADS=4

# No network access during scoring
RUN apt-get update && apt-get install -y --no-install-recommends \
    tini ca-certificates && rm -rf /var/lib/apt/lists/*

# Model and dependencies
COPY requirements.txt /app/
RUN pip install --no-cache-dir -r /app/requirements.txt

COPY model/ /app/model/
COPY score.py /app/

WORKDIR /app
ENTRYPOINT ["/usr/bin/tini", "--"]
CMD ["python", "-u", "score.py"]
```

Resource Limits (SLO)

```
Resources:
  cpu: 4 vCPU
  memory: 8 GB RAM
  disk: 10 GB (model + temp data)

Performance:
  throughput: ≥5000 alerts/minute
  p95_latency: ≤2 minutes per 10K alerts
  determinism: 100% (bit-exact reproducibility)
```

Fraud Prevention & Game Theory

Attack Vectors & Mitigations

Attack	Description	Mitigation
Model Copying	Copy top miner's outputs	Similarity detection (cosine <0.98), timing analysis
Overfitting to Traps	Learn pattern trap signatures	Commit-reveal, rotating trap patterns
Score Manipulation	All 0s or all 1s	Variance checks, monotonicity tests
Non-determinism	Different outputs per run	Micro-audits, Docker image verification
Sybil Attack	Multiple miners, same model	Plagiarism detection, diversity rewards

Incentive Alignment

```
# Reward function balances multiple objectives
total_reward = (
    0.50 * ground_truth_accuracy     # T+τ performance
  + 0.30 * trap_performance          # Anti-gaming
  + 0.10 * consistency_bonus         # Stable predictions
  + 0.05 * early_warning_bonus       # Speed of detection
  + 0.05 * diversity_bonus           # Unique predictions
) * uptime_multiplier
```

Implementation Roadmap

Phase 1: Core Scoring (Months 1-2)

- ☑ Alert risk scoring (Proposal 1)
- ☑ Basic validation (T+0 + T+τ)
- ☑ Pattern trap system
- ☑ Miner leaderboard

Phase 2: Ranking & Clusters (Months 3-4)

- Alert prioritization (Proposal 2)
- Cluster risk assessment (Proposal 3)
- Enhanced validation metrics
- Micro-audit system

Phase 3: Advanced Features (Months 5-6)

- Feature engineering submissions (Proposal 4)
- Graph embeddings
- Ablation testing framework
- Feature marketplace

Phase 4: Discovery (Months 7+)

- Pattern discovery bonuses (Proposal 5)
- Consensus-based validation
- Research miner track
- Community curation

Questions for Stakeholders

1. **Scope Priority**: Should we start with just Proposal 1 (alert scoring) or include Proposal 2 (ranking) from day 1?
2. **τ Parameter**: What's the optimal validation horizon? 7 days (fast feedback) vs 21 days (more confirmations)?
3. **K Parameter**: For Precision@K/NDCG@K, what's the realistic investigation capacity? 100, 500, or 1000 alerts/day?
4. **Feature Submission**: Should miners be allowed to submit enhanced features (Proposal 4), or only use provided features?
5. **Pattern Discovery**: Is there appetite for discovery bonuses (Proposal 5), or focus purely on scoring/ranking quality?
6. **Container Strategy**: Require all miners use Docker images, or allow direct API submission?

Conclusion

The SOT data schemas enable miners to contribute value in **5 distinct ways**, from basic alert scoring to advanced pattern discovery. The validation strategies are robust and multi-layered, making gaming unprofitable while rewarding genuine ML innovation.

Recommended Starting Point:

- **Phase 1**: Proposals 1 & 2 (Alert scoring + Prioritization)

- **Validation**: Tier 1-3 framework with pattern traps
- **Timeline**: 2-3 months to production
- **Expansion**: Add Proposals 3-5 based on demand and miner sophistication

The architecture is scalable, fraud-resistant, and provides clear value to both miners (revenue opportunity) and validators (improved AML detection).