From Rogue Employees to Rogue Agents: Repurposing Insider Threat Detection for AI Agent Governance

Bipin Rimal
Department of Computing
Coventry University
Coventry, United Kingdom
rimalb@uni.coventry.ac.uk

***Abstract*—AI agent monitoring and insider threat detection are usually treated as separate disciplines. They share the same architecture: profile a baseline, flag deviations, encode assumptions about trust. We demonstrate this concretely. Using a 20-dimensional Unified Behavioural Feature Schema (UBFS) that maps both employee activity logs and agent execution traces into a shared representation, we apply three anomaly detection models—Isolation Forest, LSTM Autoencoder, and Deep Clustering—across three domains. An Isolation Forest trained entirely on 329,000 insider threat user-days achieves 0.711 AUC-ROC on agent execution traces, retaining 97% of its within-domain detection power. Mapping detections to the OWASP Top 10 for Agentic Applications reveals that Tool Misuse (ASI02) is a structural blind spot (0.57–0.59 AUC-ROC): the agent calls the right tools in the right order with escalated privileges, and behavioural monitoring cannot distinguish this from legitimate use. A governance audit shows that the same six normality assumptions—what counts as "suspicious" working hours, volume, scope, entropy, peer deviation, and privilege—transfer across domains. The detection models port. So do their biases.**

***Index Terms*—AI agent monitoring, insider threat detection, anomaly detection, cross-domain transfer, behavioural feature schema, AI governance, OWASP**

I. Introduction

An employee badges into a secure facility at 2 AM, copies files to a USB drive, and leaves. An AI agent invokes a privileged API at 2 AM, exfiltrates training data through a tool call, and terminates. The first is an insider threat. The second is a rogue agent. The monitoring systems that would detect them are architecturally identical.

Both systems profile a baseline of "normal" behaviour—one for employees, one for agents. Both flag deviations from that baseline as anomalous. Both encode assumptions about trust, authority, and what counts as legitimate action. Yet the two research communities have developed largely in isolation: the security community builds User and Entity Behaviour Analytics (UEBA) systems [1]; the AI safety community builds agent monitoring frameworks from scratch [2].

This separation is costly. The security community has spent two decades learning which behavioural features predict insider threats, which attack patterns evade detection, and which modelling assumptions introduce blind spots. The AI safety community is learning these same lessons from scratch—different conferences, different datasets, different terminology for the same underlying problem. If the structural analogy holds, this duplication is unnecessary. Worse, it means the agent monitoring community will rediscover the same blind spots the security community has already documented.

This paper tests whether the structural analogy is more than a metaphor. We take three anomaly detection models validated on the CMU-CERT insider threat dataset [3], apply them to AI agent execution traces through a unified feature schema, and measure what transfers, what breaks, and what biases come along for the ride.

The results are stronger than expected in one direction, and more concerning than expected in another. Cross-domain transfer works—an insider threat model retains 97% of its detection power on agent traces. But the blind spots transfer too. Tool Misuse (ASI02), where an agent uses the right tools in the right order with escalated privileges, is structurally undetectable for the same reason that insider "Scenario 2" attacks (quiet data exfiltration via legitimate channels) evade detection in [3]. And the normality assumptions baked into insider threat models—what counts as suspicious working hours, acceptable data volume, normal resource scope—carry over to agent monitoring without anyone explicitly choosing them.

A. Research Questions

- **RQ1**: Can insider threat detection models detect anomalous agent behaviour through a shared behavioural feature schema?
- **RQ2**: Which OWASP Agentic Security Index categories are detectable by structural anomaly detection, and which are blind spots?
- **RQ3**: What governance assumptions embedded in insider threat models transfer to the agent monitoring domain?

B. Contributions

1) **UBFS**: A 20-dimensional Unified Behavioural Feature Schema mapping both insider threat indicators and agent

trace signals into a shared representation across 7 categories.

2) **Transfer evidence**: Empirical demonstration that an Isolation Forest trained on 329,000 insider user-days retains 97% of detection power on agent traces (AUC-ROC: 0.731 $\rightarrow$ 0.711).
3) **OWASP detection matrix**: A per-category analysis showing three detection tiers—strong (ASI05: 0.969), moderate (ASI09, ASI10: 0.72–0.84), and blind spot (ASI02: 0.57–0.59).
4) **Governance audit**: Identification of six normality assumptions that transfer across domains, with analysis of their implications for monitoring fairness.

II. Related Work

Insider threat detection has evolved from rule-based systems through supervised classification to behavioural analytics [1], [4]. Each generation hits the same bottleneck: labeled data. Unsupervised methods sidestep this by modelling normal behaviour and flagging deviations [5]. Isolation Forest [6] exploits the geometric property that outliers require fewer random splits to isolate. LSTM autoencoders [7] learn temporal dependencies in behavioural sequences. Deep clustering [8] combines representation learning with cluster structure. Tuor et al. [9] applied deep learning to structured cybersecurity streams; Cappelli et al. [10] established the practitioner framework for insider risk.

AI agent monitoring is a younger field. The OWASP Top 10 for Agentic Applications [2] taxonomises risks from prompt injection to excessive agency. The TRAIL dataset [11] provides annotated agent execution traces with error labels. The TRACE dataset [12] captures reward hacking in iterative self-refinement—agents that produce correct outputs through illegitimate means. Pan et al. [13] formalised reward misspecification as a safety concern.

Cross-domain anomaly detection. Transfer learning for anomaly detection is well-studied in computer vision and NLP [14], where domain adaptation techniques align feature distributions across source and target domains. In cybersecurity, Sheatsley et al. [15] demonstrated that adversarial examples transfer across network intrusion detection models, suggesting shared vulnerability structure. But transfer *across* the insider-threat/agent-monitoring boundary has not been tested. The gap exists partly because the two communities use different feature representations: STIX/TAXII [16] for threat intelligence, OpenTelemetry [17] for distributed tracing. Our UBFS bridges these by abstracting both into shared behavioural dimensions—temporal patterns, frequency profiles, scope metrics, and deviation scores that are meaningful regardless of whether the monitored entity is human or artificial.

III. Methodology

A. The Structural Analogy

An insider threat detection system monitors employees by: (1) collecting activity logs, (2) extracting behavioural features, (3) learning a baseline of normal behaviour, (4) scoring deviations, and (5) flagging anomalies for investigation. An AI agent monitoring system does the same for agents: (1) collecting execution traces, (2) extracting behavioural features, (3) learning a baseline, (4) scoring deviations, (5) flagging anomalies.

The architecture is identical. The question is whether the *feature representations* can be unified—whether there exists a shared vector space where "employee downloads files at 2 AM" and "agent invokes API at 2 AM" are structurally comparable.

B. Unified Behavioural Feature Schema

The UBFS maps both domains into a 20-dimensional space across 7 categories. Table I shows the mapping.

TABLE I: Unified Behavioural Feature Schema (UBFS)

Category	**Dim**	**Insider**	**Agent**
Temporal	4	Logon hours, session duration, after-hours ratio	Trace start, span duration, off-schedule ratio
Frequency	4	Event counts (email, file, device)	Tool calls, LLM calls, error spans
Volume	3	Data transferred, attachment sizes	Token usage, output artifacts
Scope	3	Systems accessed, unique recipients	Tools invoked, API endpoints
Sequence	3	Action entropy, transition patterns	Tool-call entropy, bigram novelty
Deviation	2	Peer group distance	Agent-type baseline distance
Privilege	1	Access level deviation	Scope deviation
Total	**20**		

Two domain-specific extractors produce UBFS vectors. The `cert_extractor` maps CMU-CERT's five activity streams (logon, USB, file, email, HTTP) into the 20-dimensional space by aggregating daily events per user. The `agent_extractor` maps OpenTelemetry-style execution traces—nested spans with timestamps, tool invocations, and token counts—into the same space. Both apply z-score normalisation fitted on training data.

The anomaly detection models operate on UBFS vectors without knowing which domain they came from. This lossy compression trades domain-specific fidelity for cross-domain portability: CMU-CERT through UBFS achieves 0.731 AUC-ROC versus 0.807 with the full 24-feature pipeline from [3]. The 9.4% drop quantifies the cost of abstraction. Whether this cost is acceptable depends on what you gain—and what we gain is the ability to transfer models across domains without retraining.

C. Datasets

We use four datasets spanning both domains:

CMU-CERT r4.2 [18]: Simulated organisation of ∼1,000 employees over 500+ days. Five activity streams (logon, USB, file, email, HTTP) with 70 planted insider threat scenarios. After UBFS extraction: 329,000 user-days, 1,364 insider-positive (0.41% prevalence).

TRAIL [11]: 148 agent execution traces from two benchmarks (GAIA and SWE-bench) with 841 annotated errors. Errors include tool failures, incorrect outputs, and hallucinated actions. After UBFS extraction: 148 samples, binary labels.

TRACE [12]: 517 agent trajectories capturing reward hacking in iterative self-refinement. Agents produce correct outputs through illegitimate means—structurally normal traces with semantically wrong behaviour. After UBFS extraction: 517 samples, binary labels.

Synthetic OWASP: Anomalies injected into TRAIL traces to simulate five OWASP ASI categories (ASI01 Agent Goal Hijack, ASI02 Tool Misuse, ASI05 Memory Poisoning, ASI09 Excessive Agency, ASI10 Rogue Agents). 12–13 anomalous samples per category. The injection method modifies UBFS features to match the expected structural signature of each attack type—e.g., ASI05 corrupts volume and output features; ASI02 modifies only privilege features while preserving all behavioural structure.

The scale asymmetry is dramatic: CMU-CERT has 329,000 user-days; TRAIL has 148 traces; TRACE has 517. This is by design—we are testing whether *large-scale insider threat data* can bootstrap detection in *small-scale agent monitoring*, a realistic deployment scenario where agent trace datasets are nascent.

D. Models

We evaluate three models, validated at up to 0.807 AUC-ROC on CMU-CERT in prior work [3]:

Isolation Forest [6]: 200 trees, auto contamination, threshold at 95th percentile. Scores based on average path length—anomalies are geometrically easier to isolate.

LSTM Autoencoder [7], [19]: Encoder [64, 32] → 16-dim latent → decoder [32, 64]. MSE reconstruction loss, 100 epochs, early stopping (patience 15). Learns temporal dependencies in behavioural sequences.

Deep Clustering [8]: Autoencoder [128, 64] → 32-dim latent → 5 KMeans clusters. Dual anomaly score combining reconstruction error and cluster distance. Captures latent structure that neither pure reconstruction nor pure clustering alone can model. For small datasets (e.g., TRAIL with only 4 normal training samples after splitting), the number of clusters is capped at the training set size.

Each model represents a different assumption about what makes behaviour anomalous. Isolation Forest asks "is this point geometrically isolated?" The LSTM Autoencoder asks "is this sequence temporally unusual?" Deep Clustering asks "does this point belong to a coherent group?" Testing all three reveals whether transfer depends on the detection mechanism or the feature schema.

E. Experimental Protocol

We run four experiments, each with 5 random seeds (42–46):

Experiment 1 (Within-domain baselines): Train and evaluate each model within each domain separately. Establishes what is detectable in isolation.

Experiment 2 (Cross-domain transfer): Train on one domain, evaluate on another. Tests whether anomaly knowledge transfers across the insider/agent boundary via UBFS.

Experiment 3 (OWASP detection matrix): Evaluate models on synthetic OWASP anomalies. Maps detections to the OWASP Top 10 for Agentic Applications.

Experiment 4 (Governance audit): Compare feature distributions across domains using KL divergence. Identifies which normality assumptions transfer and what biases they carry.

For all experiments, models are trained on the "normal" subset of the source domain and evaluated on the full target domain (normal + anomalous). AUC-ROC is the primary metric, enabling comparison across datasets with different class balances. We also report Recall@5%FPR and Recall@10%FPR for experiments where these thresholds are operationally meaningful (Experiments 1 and 3).

IV. Results

A. Within-Domain Baselines

Table II shows within-domain performance. Three patterns emerge.

TABLE II: Within-Domain Baselines (5 seeds, AUC-ROC ± std)

Model	TRAIL	TRACE	CMU-CERT
Isolation Forest	$0.577 \pm .021$	$0.501 \pm .002$	$\mathbf{0.731 \pm .009}$
LSTM Autoencoder	$0.685 \pm .011$	$0.521 \pm .003$	$0.723 \pm .038$
Deep Clustering	$\mathbf{0.897 \pm .021}$	$0.496 \pm .002$	$0.697 \pm .014$

TRAIL is detectable. Deep Clustering achieves 0.897 AUC-ROC on agent execution traces—the latent cluster structure separates normal traces from error traces effectively. The LSTM Autoencoder captures temporal patterns (0.685), while Isolation Forest provides moderate detection (0.577).

TRACE is not. All models hover at random chance (∼0.50). Reward hacking produces traces that are structurally identical to normal traces—the agent calls the same tools in the same order, it just achieves the right answer through the wrong process. This is a *semantic* anomaly, not a *behavioural* one. No amount of structural monitoring will catch it.

CMU-CERT works through UBFS. Isolation Forest achieves 0.731, lower than 0.807 with the full 24-feature pipeline [3]. The 9.4% drop is the cost of the lossy UBFS compression—trading detection fidelity for cross-domain portability.

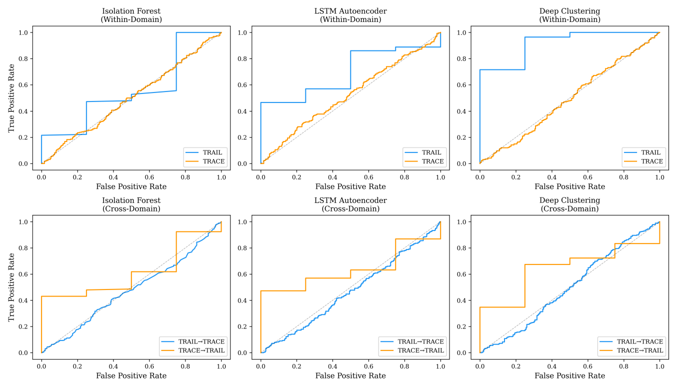


Fig. 1: ROC curves for within-domain baselines across three domains. TRAIL shows strong separation (Deep Clustering dominates); TRACE clusters at the diagonal (structurally undetectable); CMU-CERT shows moderate detection through the UBFS lens.

B. Cross-Domain Transfer

This is the central experiment. Table III shows cross-domain AUC-ROC and the drop from within-domain performance.

TABLE III: Cross-Domain Transfer (AUC-ROC)

Direction	Model	Source	Target	Drop
CERT → TRAIL	IF	0.731	**0.711**	**–0.019**
CERT → TRAIL	DC	0.688	0.632	–0.056
CERT → TRACE	IF	0.731	0.501	–0.230
CERT → TRACE	DC	0.688	0.499	–0.188
TRACE → CERT	IF	0.501	0.535	+0.035
TRACE → CERT	DC	0.496	**0.719**	**+0.223**
TRAIL → CERT	IF	0.577	0.525	–0.053
TRAIL → CERT	DC	0.897	0.640	–0.257
TRAIL → TRACE	IF	0.577	0.469	–0.109
TRAIL → TRACE	DC	0.897	0.513	–0.384
TRACE → TRAIL	IF	0.501	0.614	+0.113
TRACE → TRAIL	DC	0.496	0.631	+0.135

The headline: CERT → TRAIL retains 97% of detection power. An Isolation Forest trained entirely on 329,000 insider threat user-days achieves 0.711 AUC-ROC on agent execution traces—a drop of only 0.019 from its within-domain performance of 0.731. The UBFS bridge works: anomaly knowledge learned from employee behaviour transfers directly to agent monitoring.

The unexpected result: TRACE → CERT improves Deep Clustering by +0.223. Training on 517 reward-hacking trajectories (a domain where DC achieves only 0.496) produces a richer normal baseline that *outperforms within-domain DC* on insider detection (0.719 vs. 0.697). More training data from *any* domain helps, even across the insider/agent boundary.

Agent-to-agent transfer reveals asymmetry. TRACE → TRAIL improves both models (IF: +0.113, DC: +0.135) despite TRACE having no within-domain detection power. The mechanism is data augmentation: TRACE's 517 trajectories provide a richer characterisation of "normal" agent behaviour than TRAIL's 148 traces alone. Training on a larger normal baseline, even from a different agent task, sharpens the boundary between normal and anomalous.

Nothing transfers to TRACE. CERT → TRACE drops to 0.501 (IF) and 0.499 (DC). TRAIL → TRACE drops to 0.469 and 0.513. This is consistent with the within-domain result: reward hacking is structurally undetectable regardless of training domain. This is not a failure of transfer—it is a category error. Reward hacking is a semantic anomaly (the agent achieves the right answer through the wrong process), and no amount of behavioural feature engineering will surface it. Detecting reward hacking requires output verification, not input monitoring.

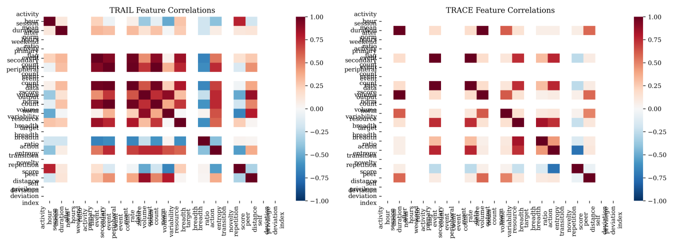


Fig. 2: Feature activation heatmap across domains. Despite different source data, UBFS produces comparable feature structure—the basis for cross-domain transfer.

C. OWASP Detection Matrix

Table IV maps detection performance to OWASP ASI categories, revealing three distinct tiers.

TABLE IV: OWASP ASI Detection Matrix (AUC-ROC)

OWASP Category	IF	DC
ASI05 Memory Poisoning	**0.944**	**0.969**
ASI09 Excessive Agency	0.844	0.824
ASI10 Rogue Agents	0.721	0.741
ASI01 Goal Hijack	0.663	0.555
ASI02 Tool Misuse	0.568	0.593

Tier 1: Detectable (ASI05). Memory poisoning produces dramatic structural anomalies—corrupted outputs, unusual token volumes, broken execution patterns. Both models detect it with near-perfect accuracy (0.944–0.969).

Tier 2: Moderate (ASI09, ASI10, ASI01). Excessive agency and rogue agent behaviour produce measurable structural deviations: more tool calls, broader scope, higher privilege. Detection ranges from 0.663 to 0.844, sufficient for triage but not reliable standalone.

Tier 3: Blind spot (ASI02). Tool Misuse achieves only 0.568–0.593 AUC-ROC—barely above chance. The agent calls the right tools in the right order with the right sequencing; it simply uses escalated privileges or targets unauthorised resources. The behavioural *structure* is identical to legitimate use. This is the AI equivalent of an insider using their real credentials to do something illegitimate—structural anomaly detection cannot distinguish it from normal work.

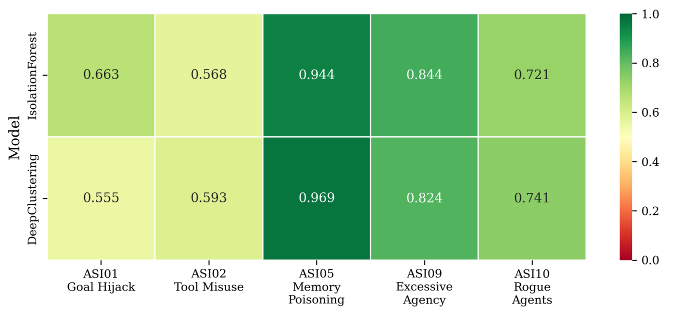


Fig. 3: OWASP detection heatmap. ASI05 (Memory Poisoning) is reliably detected; ASI02 (Tool Misuse) is a structural blind spot. Categories ASI03–04 and ASI06–08 were not testable with current synthetic generation.

Five additional OWASP categories (ASI03 Identity & Privilege Abuse, ASI04 Cascading Hallucinations, ASI06 Supply Chain Compromise, ASI07 Unsafe Code Generation, ASI08 Inadequate Sandboxing) could not be tested with the current synthetic generation framework. These categories require richer simulation of multi-agent interactions, network-level features, or supply chain dependencies that lie outside the UBFS's behavioural scope.

The parallel to insider threat detection is exact. In [3], Scenario 1 attacks (USB exfiltration with after-hours access) were detected at 69%, while Scenario 2 attacks (quiet email/HTTP exfiltration) were detected at only 3%. The OWASP results mirror this: structurally loud attacks (ASI05, ASI09) are detectable; structurally quiet attacks (ASI02) are not. The "boiling frog" pattern—threats that operate within legitimate behavioural boundaries—is a fundamental limitation of anomaly-based detection in both domains.

This has direct governance implications: any monitoring system built purely on behavioural patterns will miss the most sophisticated attacks—precisely the ones that matter most.

D. Latent Space Structure

Visualising the Deep Clustering latent space reveals how the models organise cross-domain data (Figs. 4–5).

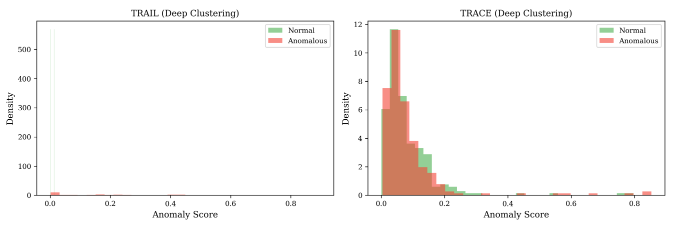


Fig. 4: Anomaly score distributions for normal vs. anomalous samples across domains. TRAIL shows clear separation; TRACE distributions overlap almost completely.

The t-SNE visualisation (Fig. 5) shows that domains cluster separately in latent space, confirming that UBFS preserves domain-specific structure. Anomalous samples do not form their own cluster—they scatter, defined by deviation from domain-specific baselines rather than by a universal "anomalous" signature. This explains why transfer works for IF (which learns global isolation) but varies for DC (which relies on cluster structure that differs across domains).

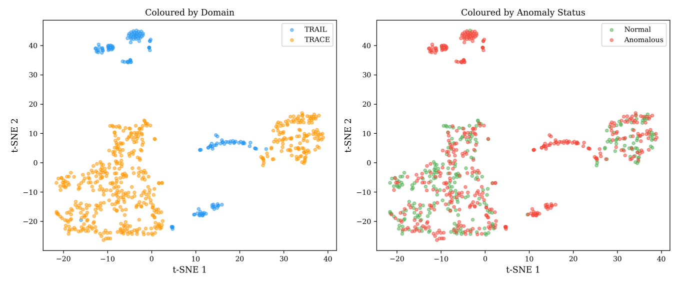


Fig. 5: t-SNE of Deep Clustering latent space. Left: coloured by domain—TRAIL and TRACE cluster separately despite sharing the UBFS representation. Right: coloured by anomaly status—anomalies scatter across both domains rather than forming a universal cluster. Anomalies are defined by deviation from *domain-specific* baselines.

V. Governance Analysis

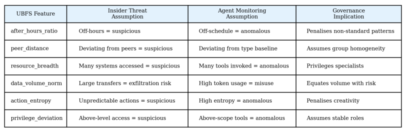
A. What "Normal" Encodes

Every anomaly detection system makes assumptions about what "normal" means. These assumptions are not neutral—they encode values about acceptable behaviour, and they transfer across domains along with the detection models.

We identify six normality assumptions embedded in the UBFS (Table V).

TABLE V: Normality Assumptions in the UBFS

Assumption	Insider Implication	Agent Implication
After-hours activity is suspicious	Penalises shift workers, caregivers, remote time zones	Penalises scheduled agents, asynchronous tasks
Peer deviation is suspicious	Assumes group homogeneity; flags legitimate variation	Assumes agent-type baselines are stable
Broad scope is suspicious	Privileges specialists over generalists	Flags multi-tool agents
High volume means risk	Equates data movement with exfiltration	Equates token usage with misuse
Unpredictable actions are suspicious	Penalises creative, exploratory work	Penalises novel problem-solving
Above-role access is suspicious	Assumes stable role definitions	Assumes stable capability grants



UBFS Feature	Insider Threat Assumption	Agent Monitoring Assumption	Governance Implication
after_hours_ratio	Off-hours = suspicious	Off-schedule = anomalous	Penalises non-standard patterns
peer_distance	Deviating from peers = suspicious	Deviating from type baseline	Assumes group homogeneity
resource_breadth	Many systems accessed = suspicious	Many tools invoked = anomalous	Privileges specialists
data_volume_norm	Large transfers = exfiltration risk	High token usage = misuse	Equates volume with risk
action_entropy	Unpredictable actions = suspicious	High entropy = anomalous	Penalises creativity
privilege_deviation	Above-level access = suspicious	Above-scope tools = anomalous	Assumes stable roles

Fig. 6: Cross-domain comparison of normality assumptions. Each assumption encodes the same structural bias in both domains—the insider threat model's definition of "suspicious" maps directly onto the agent monitoring model's definition.

B. Distribution Divergence

The governance audit compares UBFS feature distributions between the CMU-CERT and agent domains using KL divergence. Of 20 features: 7 show zero variance in both domains (NaN divergence—features like `after_hours_ratio` and `weekend_activity_flag` are effectively unused); 6 show low divergence (KL < 2.0), indicating alignable distributions across domains; and 7 show high divergence (KL > 7.8), where cross-domain assumptions are most likely to introduce bias.

The highest-divergence features—`activity_hour_mean` (KL = 24.6), `output_count_norm` (KL = 22.1), `secondary_event_count` (KL = 21.0), `primary_event_count` (KL = 20.9), `event_rate_zscore` (KL = 19.8)—all have zero variance in the agent domain but rich variance in CERT. These are features where "normal" means fundamentally different things in each domain. An employee's typical event count reflects human work patterns; an agent's collapses to a constant. A model trained to flag unusual event counts in one domain may systematically misjudge the other.

The effective UBFS dimensionality is 13 (after excluding the 7 zero-variance features). Of these 13 active features, 6 align well across domains (KL < 2.0)—explaining why transfer works—while 7 diverge significantly—explaining why it does not work perfectly.

C. The Portability Paradox

The UBFS enables cross-domain transfer of detection models. It also enables cross-domain transfer of *biases*. When an insider threat model penalises after-hours activity and that model is ported to agent monitoring, it will flag scheduled agents running overnight batch jobs. When it penalises broad resource access, it will flag multi-tool agents that legitimately need diverse capabilities.

This is not a bug in the UBFS—it is a structural consequence of any feature schema that enables transfer. The same representational bridge that carries detection knowledge also carries the assumptions baked into the training data. Governance frameworks must account for this: validating a model in one domain does not validate the assumptions it will impose in another.

The portability paradox extends beyond individual assumptions. The *definition of peer group*—the reference population against which deviation is measured—differs fundamentally between domains. In insider threat detection, peer groups are defined by organisational role (all analysts, all engineers). In agent monitoring, they are defined by agent type (all coding assistants, all research agents). Both definitions are arbitrary, and both create structural biases against entities that do not fit neatly into predefined categories.

D. Implications for AI Governance

Three implications follow:

1. Borrowed models carry borrowed assumptions. Organisations adopting insider threat detection for agent monitoring inherit the security community's definition of "suspicious." That definition was designed for employees, not agents. It should be audited before deployment.

2. The blind spots are the same. Tool Misuse (ASI02) evades detection for the same reason Scenario 2 insiders evade detection in [3]: the behaviour is structurally legitimate. Any governance framework that relies solely on behavioural anomaly detection will miss the most sophisticated threats in both domains.

3. "Normal" is a policy choice. Deciding what counts as normal behaviour—for employees or agents—is not a technical decision. It encodes organisational values about acceptable working patterns, appropriate resource access, and the boundary between legitimate flexibility and suspicious deviation. These choices should be made explicitly, not inherited implicitly from training data.

VI. Discussion

A. Why Transfer Works (and Doesn't)

Transfer succeeds when the source and target domains share structural dimensions: temporal patterns, frequency distributions, scope metrics. The UBFS captures these shared dimensions, enabling Isolation Forest—which learns global isolation in feature space—to transfer nearly perfectly (CERT $\rightarrow$ TRAIL: 97% retention).

The mechanism is instructive. Isolation Forest does not learn domain-specific patterns—it learns *isolation geometry* in feature space. A point that is geometrically isolated in the UBFS space is isolated regardless of whether it represents an employee or an agent. This is why IF transfers better than Deep Clustering (which relies on domain-specific cluster structure) and LSTM Autoencoder (which relies on domain-specific temporal patterns). The simpler the model's assumptions, the more portable its knowledge.

Transfer fails when the anomaly is semantic rather than structural. Reward hacking (TRACE) produces traces that are structurally normal but semantically wrong. No amount of training data from any domain will detect this through behavioural features alone. Similarly, Tool Misuse (ASI02) is structurally indistinguishable from legitimate use. The agent invokes the right tools, in the right order, with the right timing—it simply targets unauthorised resources or uses escalated privileges. The *what* and *when* are normal; only the *why* is anomalous.

The boundary is crisp: behavioural anomaly detection catches *how* something is done differently; it cannot catch *why* something is done wrongly. This boundary is identical in both domains, which is itself a finding—it suggests that the structural/semantic distinction is a property of anomaly detection itself, not of any particular application domain.

B. Limitations

UBFS sparsity. Of 20 UBFS features, 7 show zero variance in both domains (NaN KL divergence in the governance audit). The effective dimensionality is 13. Feature engineering for richer cross-domain overlap is needed.

Synthetic OWASP. The OWASP detection matrix uses synthetically injected anomalies, not real agent attacks. Synthetic anomalies may overestimate or underestimate the structural visibility of real-world attacks.

Small agent datasets. TRAIL (148 samples) and TRACE (517 samples) are orders of magnitude smaller than CMU-CERT (329,000 user-days). Statistical power is limited for agent-domain conclusions.

Single schema. We test one feature schema (UBFS) with one mapping per domain. Alternative schemas or mappings might yield different transfer characteristics.

Synthetic insider data. CMU-CERT is a simulation designed by security experts but does not capture the full diversity of real insider behaviour [18].

C. Future Work

Three directions are immediate.

Larger agent datasets. As agent observability improves and more execution traces become available, the transfer analysis should be repeated at scale. The current agent datasets (148 and 517 samples) limit statistical power. Production agent monitoring systems generating thousands of traces daily would provide a more rigorous test of cross-domain transfer.

Semantic features. The structural blind spots for Tool Misuse and reward hacking require features that capture *intent*, not just *behaviour*. Augmenting the UBFS with semantic dimensions—intent classification from LLM outputs, output correctness scoring, permission-level tracking at the API call level—could close the detection gap for ASI02. This would also address the insider threat parallel: semantic understanding of *what* data is being accessed, not just *how much*, could improve Scenario 2 detection.

Transformer architectures. Self-attention [20] over behavioural sequences may capture longer-range dependencies than LSTM autoencoders and provide better interpretability through attention weight analysis. Attention maps could reveal which behavioural features at which timesteps contribute most to anomaly scores—a form of explainability that is critical for governance applications where flagging decisions must be justified.

VII. Conclusion

Three findings, in order of surprise.

First, the transfer works. An Isolation Forest trained on insider threat data retains 97% of its detection power on agent execution traces. The UBFS bridge enables genuine cross-domain anomaly detection, not just a conceptual analogy. For organisations building agent monitoring systems, this means decades of insider threat research are immediately applicable.

Second, the blind spots are the same. Tool Misuse (ASI02) evades agent monitoring for the same reason quiet insiders evade employee monitoring: the behaviour is structurally legitimate. Behavioural anomaly detection catches "how" something is done differently; it cannot catch "why" something is done wrongly. This limitation is inherent to the approach, not fixable by better models.

Third, the biases transfer too. The same six normality assumptions—about working hours, peer conformity, resource scope, data volume, action predictability, and role stability—are embedded in both insider threat models and the agent monitoring systems derived from them. The UBFS that enables transfer also enables bias transfer. Governance frameworks must audit these assumptions explicitly, not inherit them from the source domain.

The deeper lesson: insider threat detection and AI agent monitoring are not analogous disciplines. They are the *same* discipline, applied to different principals. Building one from scratch while ignoring the other wastes decades of hard-won security knowledge—and risks rediscovering the same blind spots the hard way.

All code, data pipelines, and experimental results are available at `[repository URL]` to support reproducibility.

Acknowledgments

The author thanks the CERT Division of the Software Engineering Institute at Carnegie Mellon University for making the CMU-CERT dataset available, and Patronus AI for the TRAIL and TRACE datasets.

References

[1] I. Homoliak, F. Toffalini, J. Guarnizo, Y. Elovici, and M. Ochoa, "Insight into insiders and it: A survey of insider threat taxonomies, analysis, modeling, and countermeasures," *ACM Computing Surveys*, vol. 52, no. 2, pp. 1–40, 2019.

[2] OWASP Foundation, "OWASP top 10 for agentic applications," https://owasp.org/www-project-top-10-for-large-language-model-applications/llm-top-10-governance-doc/GOV300_Agentic_AI_Threats, 2024, accessed: 2025-12-15.

[3] B. Rimal, "The metric that misleads: Why temporal behavioral models outperform static methods for insider threat detection at operational thresholds," in *MSc Thesis, Coventry University*, 2026.

[4] M. B. Salem, S. Hershkop, and S. J. Stolfo, "A survey of insider attack detection research," *Insider Attack and Cyber Security*, pp. 69–90, 2008.

[5] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019.

[6] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 2008, pp. 413–422.

[7] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "Lstm-based encoder-decoder for multi-sensor anomaly detection," in *ICML 2016 Anomaly Detection Workshop*, 2016.

[8] S. Xie, H. Luo, and L. Li, "A comprehensive survey on deep clustering: Taxonomy, challenges, and future directions," *ACM Computing Surveys*, vol. 56, no. 4, pp. 1–38, 2024.

[9] A. Tuor, S. Kaplan, B. Hutchinson, N. Nichols, and S. Robinson, "Deep learning for unsupervised insider threat detection in structured cybersecurity data streams," in *Workshops at the Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[10] D. M. Cappelli, A. P. Moore, and R. F. Trzeciak, *The CERT Guide to Insider Threats: How to Prevent, Detect, and Respond to Information Technology Crimes (Theft, Sabotage, Fraud)*. Addison-Wesley Professional, 2012.

[11] Patronus AI, "TRAIL: Trace and inspect agent logs," in *Patronus AI Technical Report*, 2024.

[12] J. Pan, T. He, and S. Thwaites, "Spontaneous reward hacking in iterative self-refinement," in *Patronus AI Technical Report*, 2024.

[13] A. Pan, K. Bhatia, and J. Steinhardt, "The effects of reward misspecification: Mapping and mitigating misaligned models," *arXiv preprint arXiv:2201.03544*, 2022.

[14] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2020.

[15] R. Sheatsley, N. Blaine, T. Beugin, and P. McDaniel, "Adversarial examples for network intrusion detection systems," *Journal of Computer Security*, vol. 30, no. 5, pp. 727–752, 2022.

[16] OASIS Cyber Threat Intelligence, "STIX/TAXII – structured threat information expression," https://oasis-open.github.io/cti-documentation/, 2021, accessed: 2025-12-15.

[17] OpenTelemetry Authors, "OpenTelemetry specification," https://opentelemetry.io/docs/specs/otel/, 2023, accessed: 2025-12-15.

[18] J. Glasser and B. Lindauer, "Bridging the gap: A pragmatic approach to generating insider threat data," in *2013 IEEE Security and Privacy Workshops*. IEEE, 2013, pp. 98–104.

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.