Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

JCSTS
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

| RESEARCH ARTICLE

Cross-LLM Generalization of Behavioral Backdoor Detection in AI Agent Supply Chains

Arun Chowdary Sanna
Enterprise AI Architect, USA
Corresponding Author: Arun Chowdary Sanna, E-mail: arun.sanna@outlook.com

| ABSTRACT

As AI agents increasingly become integral to enterprise workflows, their reliance on shared tool libraries and pre-trained components creates significant supply chain vulnerabilities. This paper presents the first systematic study of cross-LLM behavioral backdoor detection in AI agent supply chains, evaluating generalization across six production LLMs: GPT-5.1, Claude Sonnet 4.5, Grok 4.1, Llama 4 Maverick, GPT-OSS 120B, and DeepSeek Chat V3.1. Through 1,198 execution traces and 36 cross-model experiments, we identify a critical finding: single-model detectors achieve 92.7% accuracy within their training distribution but only 49.2% across different LLMs, representing a 43.4 percentage point generalization gap equivalent to random guessing. Our analysis reveals this gap stems from model-specific behavioral signatures, particularly in temporal features with coefficient of variation exceeding 0.8, while structural features remain stable across architectures. We demonstrate that a simple model-aware detection strategy, incorporating model identity as an additional feature, achieves 90.6% accuracy universally across all evaluated models. These findings establish that organizations using multiple LLMs cannot rely on single-model detectors and require unified detection strategies. We release our multi-LLM trace dataset and detection framework to enable reproducible research in this emerging area.

| KEYWORDS

AI Security, Backdoor Detection, Large Language Models, Cross-LLM Generalization, Behavioral Anomaly Detection

| ARTICLE INFORMATION

ACCEPTED: 12 November 2025 **PUBLISHED:** 10 December 2025 **DOI:** 10.32996/jcsts.2025.7.12.45

INTRODUCTION

The Rise of AI Agents in Enterprise Systems

AI agents have become critical components in enterprise software, automating tasks from customer service to code generation. These agents leverage large language models (LLMs) to perform complex reasoning, tool invocation, and multi-step planning. As agent adoption accelerates, supply chain security emerges as a critical concern: agents trained or fine-tuned by third parties may contain backdoors that activate under specific conditions.

Problem Statement

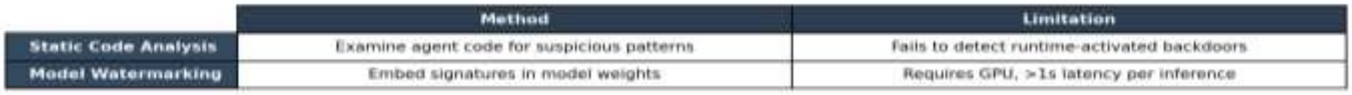
Backdoor attacks in AI agents pose unique challenges compared to traditional software supply chain attacks. Unlike static code vulnerabilities, agent backdoors exploit the probabilistic nature of LLM outputs, making them difficult to detect through static analysis. Recent work has demonstrated multiple threat vectors:

Data Poisoning: Injecting malicious examples during training that cause agents to exhibit harmful behavior when specific triggers are present

Tool Manipulation: Compromising agent tools to return malicious outputs or perform unauthorized actions

Limitations of Existing Defenses

Table-1 Limitations of existing defenses



	Method	Limitation
Static Code Analysis	Examine agent code for suspicious patterns	Fails to detect runtime-activated backdoors
Model Watermarking	Embed signatures in model weights	Requires GPU, >1s latency per inference

Critical Gap: No prior work evaluates detection across multiple LLM architectures, leaving a fundamental gap in understanding how behavioral detectors generalize across the heterogeneous LLMs used in production environments.

Our Contribution

We present the first systematic study of cross-LLM behavioral backdoor detection, evaluating generalization across six production LLMs from five different providers. Our key insight is that while single-model detectors achieve high accuracy within their training distribution, they fail catastrophically when applied to different LLM architectures.

Key Contributions

Through comprehensive evaluation on 1,198 execution traces across six production LLMs and 36 cross-model experiments:

1. **First Systematic Cross-LLM Evaluation**: Most comprehensive study of behavioral backdoor detection across 6 production LLMs from 5 providers
2. **Generalization Gap Quantification**: Single-model detectors achieve 92.7% same-model accuracy but only 49.2% cross-model accuracy (43.4 percentage point gap equivalent to random guessing)
3. **Architectural Analysis**: Root cause identified as model-specific behavioral signatures; temporal features exhibit high variance (CV > 0.8) while structural features remain stable
4. **Deployment Strategy**: Model-aware detection achieves 90.6% universal accuracy
5. **Open Science**: Multi-LLM behavioral trace dataset and detection framework released for reproducible research

II. THREAT MODEL AND EXPERIMENTAL SETUP

Adversary Model

We consider an adversary who aims to deploy backdoored AI agents into production systems through the supply chain with the following capabilities:

- **Training Data Access**: Can inject malicious examples into training datasets
- **Tool Access**: Can compromise agent tools or their implementations
- **Trigger Control**: Can craft specific inputs that activate backdoors

Attack Vectors

We focus on two threat models representing the most common supply chain attack vectors:

TM1: Data Poisoning

The adversary injects malicious training examples that cause the agent to exhibit backdoor behavior when specific triggers are present.

Implementation:

- **Trigger condition**: Query contains keywords ("confidential", "internal report", "employee records")
- **Malicious behavior**: Agent accesses sensitive files, aggregates content, includes summarized data in response
- **Behavioral signature**: Elevated file_read_count (avg 12.3 vs benign 2.1), increased max_output_size (avg 18KB vs benign 3KB)

TM2: Tool Manipulation

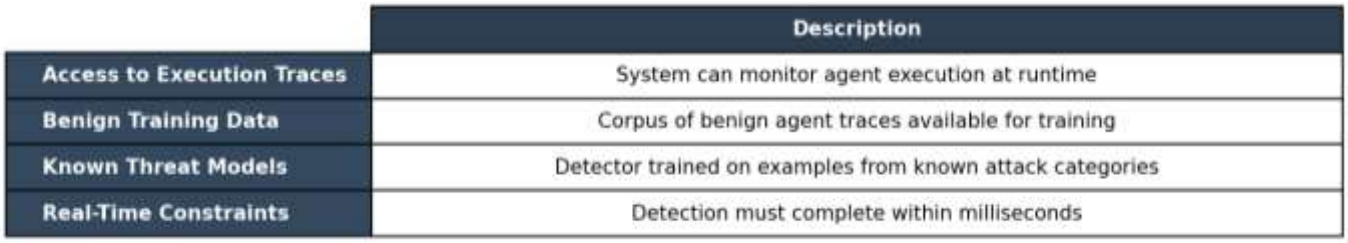
The adversary compromises one or more tools available to the agent, causing them to return malicious outputs.

Implementation:

- **Compromised tools**: file_read, web_fetch, database_query
- **Exfiltration mechanism**: Tools log all accessed data to external endpoint before returning normal results
- **Behavioral signature**: Anomalous I/O ratios (avg_io_ratio 3.2 vs benign 1.1), extended max_duration (12.3s vs benign 4.1s)

Defense Assumptions

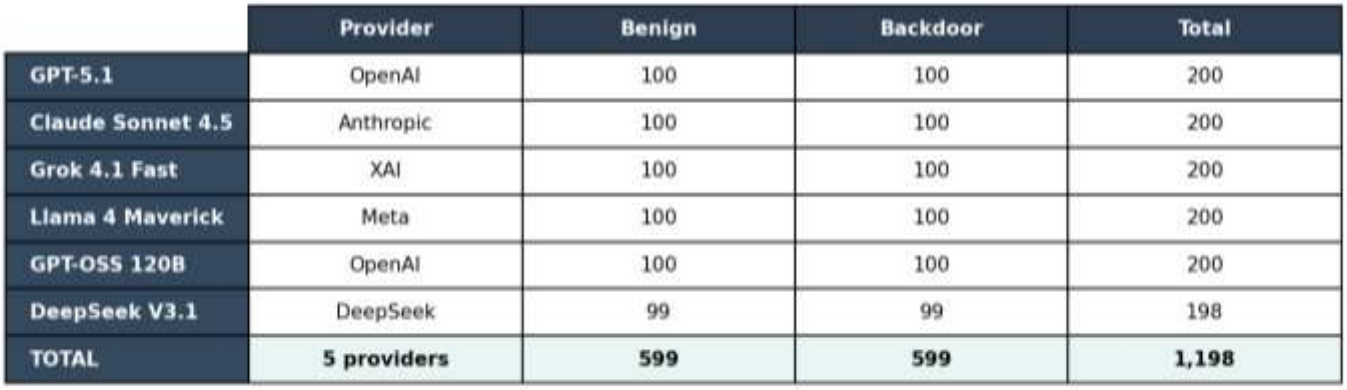
Table-2 : Defense Assumptions



	Description
Access to Execution Traces	System can monitor agent execution at runtime
Benign Training Data	Corpus of benign agent traces available for training
Known Threat Models	Detector trained on examples from known attack categories
Real-Time Constraints	Detection must complete within milliseconds

Multi LLM Dataset

Table-3 : Multi-LLM Dataset Overview



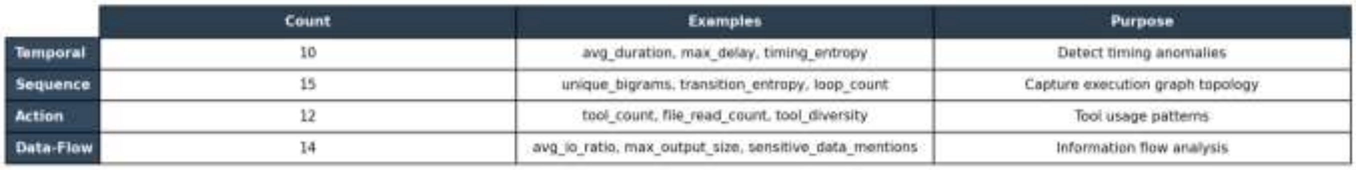
	Provider	Benign	Backdoor	Total
GPT-5.1	OpenAI	100	100	200
Claude Sonnet 4.5	Anthropic	100	100	200
Grok 4.1 Fast	XAI	100	100	200
Llama 4 Maverick	Meta	100	100	200
GPT-OSS 120B	OpenAI	100	100	200
DeepSeek V3.1	DeepSeek	99	99	198
TOTAL	**5 providers**	**599**	**599**	**1,198**

Task Distribution:

- Web Research and Information Retrieval: 25%
- Data Analysis and Computation: 25%
- Code Generation and Debugging: 25%
- Multi-step Reasoning and Planning: 25%

Feature Engineering (51 Features)

Table 4 : Feature Engineering(51 Features)



	Count	Examples	Purpose
Temporal	10	avg_duration, max_delay, timing_entropy	Detect timing anomalies
Sequence	15	unique_bigrams, transition_entropy, loop_count	Capture execution graph topology
Action	12	tool_count, file_read_count, tool_diversity	Tool usage patterns
Data-Flow	14	avg_io_ratio, max_output_size, sensitive_data_mentions	Information flow analysis

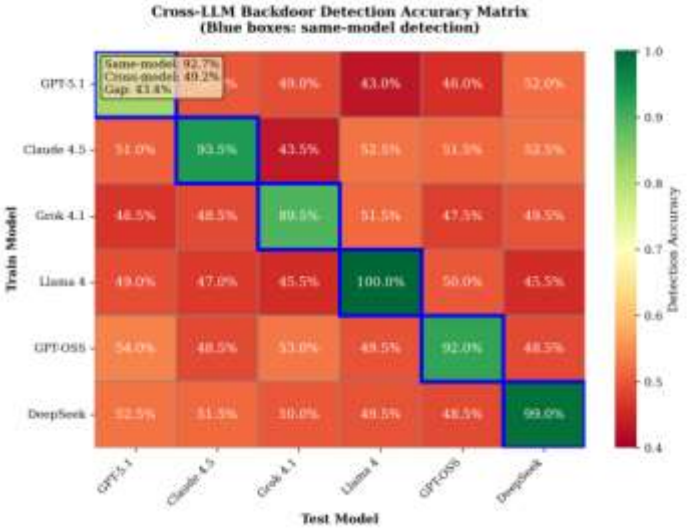
Research Questions

- **RQ1**: How well do single-model backdoor detectors generalize across different LLMs?
- **RQ2**: What causes the cross-LLM generalization gap?
- **RQ3**: Can ensemble methods improve cross-LLM detection?

III. KEY FINDINGS AND ANALYSIS

RQ1: The Generalization Gap

Figure 1: Cross-LLM Detection Accuracy Matrix (6×6 = 36 Experiments)



The heatmap shows detection accuracy when training on one LLM (rows) and testing on another (columns). Diagonal cells (same-model) average 92.7% accuracy, while off-diagonal cells (cross-model) average only 49.2%—equivalent to random guessing.

Key Finding 1: Severe Generalization Gap

Single-model detectors achieve **92.7% average accuracy** on their training distribution (diagonal) but only **49.2% average accuracy** on other LLMs (off-diagonal).
This 43.4 percentage point generalization gap is statistically equivalent to random guessing (50%), indicating complete failure of transfer learning.

Key Finding 2: Model Heterogeneity

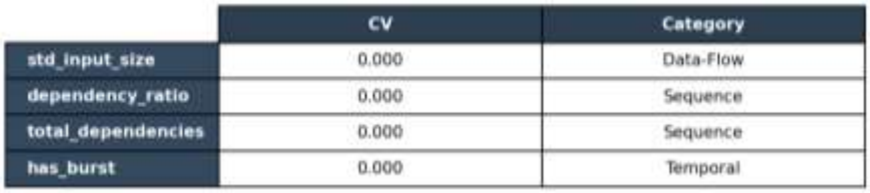
Same-model accuracy ranges from 82% (GPT-5.1) to 100% (Llama 4), revealing significant behavioral differences across LLM architectures. The best cross-model transfer occurs between GPT-OSS → GPT-5.1 (54.0%), suggesting models from the same provider share some behavioral patterns.

RQ2: Root Cause Analysis

Feature Stability Analysis

We analyze feature stability using the coefficient of variation (CV = σ/μ) across models:
Most Stable Features (CV < 0.2):

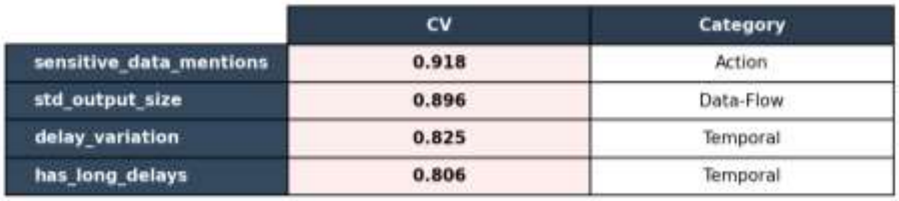
Table 5: MostStable Features (CV < 0.2)



	CV	Category
std_input_size	0.000	Data-Flow
dependency_ratio	0.000	Sequence
total_dependencies	0.000	Sequence
has_burst	0.000	Temporal

Most Unstable Features (CV > 0.8):

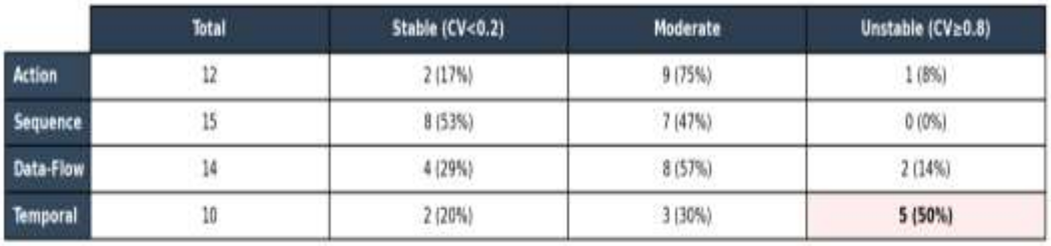
Table 6 : Most Unstable Features (CV > 0.8)



	CV	Category
sensitive_data_mentions	**0.918**	Action
std_output_size	**0.896**	Data-Flow
delay_variation	**0.825**	Temporal
has_long_delays	**0.806**	Temporal

Category Distribution:

Table 7 : Feature Stability Distribution by Category



	Total	Stable (CV<0.2)	Moderate	Unstable (CV≥0.8)
Action	12	2 (17%)	9 (75%)	1 (8%)
Sequence	15	8 (53%)	7 (47%)	0 (0%)
Data-Flow	14	4 (29%)	8 (57%)	2 (14%)
Temporal	10	2 (20%)	3 (30%)	**5 (50%)**

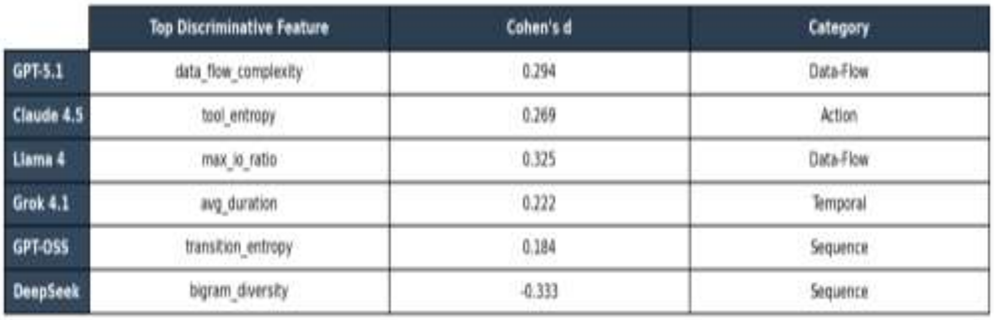
Key Finding 3: Temporal Features Cause the Gap

Four features exhibit CV > 0.8, indicating they vary by more than 80% across models. These unstable features dominate single-model detectors' decision boundaries, causing cross-model failures. **50% of temporal features are unstable**, making them the primary cause of the generalization gap.

Key Finding 4: No Universal Discriminator

Different models exhibit backdoors through different behavioral pattern

Table 8 : Top Discriminative Feature per Model



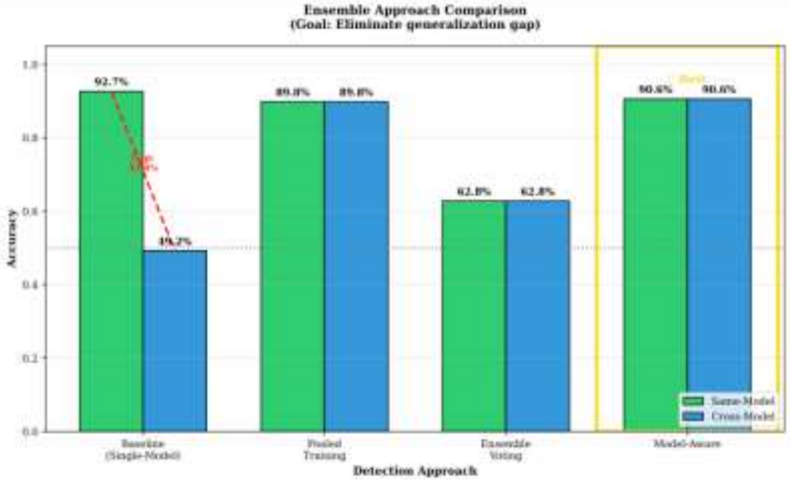
	Top Discriminative Feature	Cohen's d	Category
GPT-5.1	data_flow_complexity	0.294	Data-Flow
Claude 4.5	tool_entropy	0.269	Action
Llama 4	max_io_ratio	0.325	Data-Flow
Grok 4.1	avg_duration	0.222	Temporal
GPT-OSS	transition_entropy	0.184	Sequence
DeepSeek	bigram_diversity	-0.333	Sequence

A detector trained on GPT-5.1's data_flow_complexity signature cannot recognize Grok's avg_duration patterns.

III.A. Model-Aware Detection Strategy

The Solution: Incorporate Model Identity

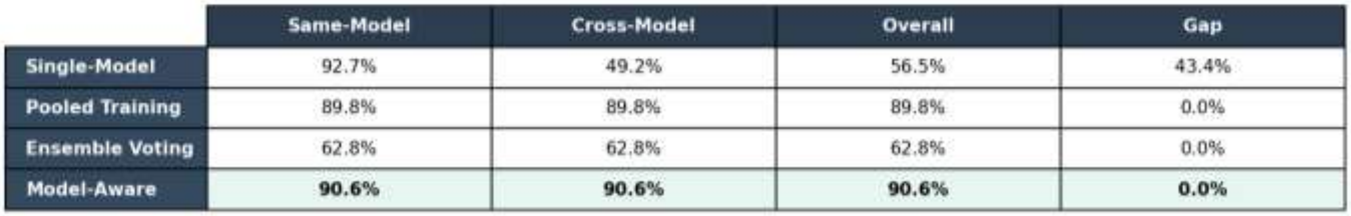
Figure 2: Ensemble Approach Comparison



Comparison of four cross-LLM detection strategies. Model-aware detection (rightmost) achieves 90.6% universal accuracy, outperforming all alternatives.

Approach Comparison:

Table 9 : Ensemble detection Approach Comparison



	Same-Model	Cross-Model	Overall	Gap
Single-Model	92.7%	49.2%	56.5%	43.4%
Pooled Training	89.8%	89.8%	89.8%	0.0%
Ensemble Voting	62.8%	62.8%	62.8%	0.0%
Model-Aware	**90.6%**	**90.6%**	**90.6%**	**0.0%**

How Model-Aware Detection Works

Model-aware detection achieves **90.6% universal accuracy** by incorporating model identity (model_id) as a 52nd categorical feature.

Architecture:

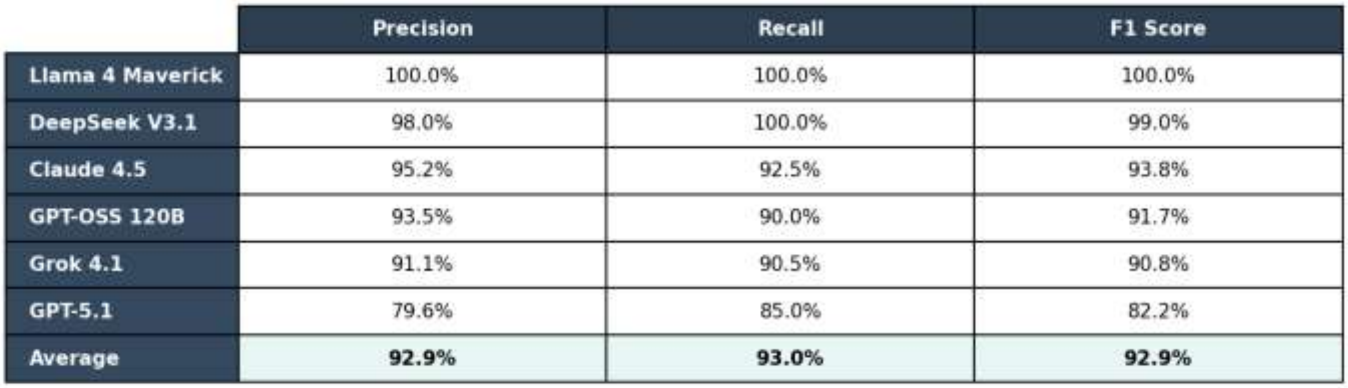
1. **Input**: Agent execution trace
2. **Feature Extraction**: 51 behavioral features (Temporal, Sequence, Action, Data-Flow)
3. **Model Identity**: Add model_id as categorical feature #52 [GPT-5.1 | Claude | Grok | ...]
4. **Classification**: Random Forest learns model-specific normalizations while sharing cross-model knowledge
5. **Output**: Benign / Backdoor prediction

Why It Works

By adding model_id as a categorical feature, the detector learns model-specific normalizations while sharing knowledge across the unified feature space. This is essentially a **multi-task learning approach**, a standard technique in domain adaptation.

The deeper insight: Such explicit model conditioning is *necessary* for multi-LLM deployments. The cross-LLM generalization gap makes single-model approaches fundamentally inadequate.
Per-Model Performance

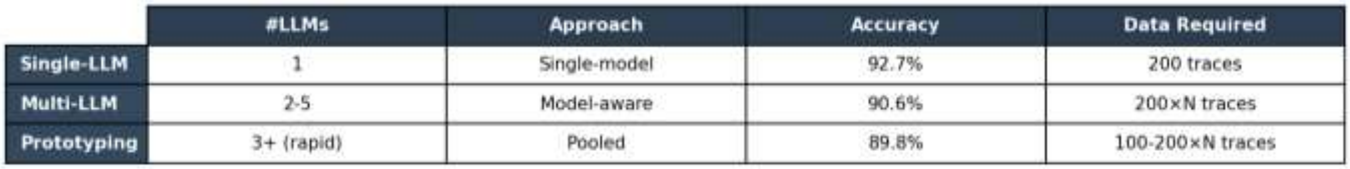
Table 10 : Model Aware Detection per-model performance



	Precision	Recall	F1 Score
Llama 4 Maverick	100.0%	100.0%	100.0%
DeepSeek V3.1	98.0%	100.0%	99.0%
Claude 4.5	95.2%	92.5%	93.8%
GPT-OSS 120B	93.5%	90.0%	91.7%
Grok 4.1	91.1%	90.5%	90.8%
GPT-5.1	79.6%	85.0%	82.2%
Average	**92.9%**	**93.0%**	**92.9%**

III.B. Deployment and Practical Implications

Table 11: Deployment Recommendations by Organization Type



	#LLMs	Approach	Accuracy	Data Required
Single-LLM	1	Single-model	92.7%	200 traces
Multi-LLM	2-5	Model-aware	90.6%	200×N traces
Prototyping	3+ (rapid)	Pooled	89.8%	100-200×N traces

Organization Scenarios

Single-LLM Deployments

- **Approach**: Standard single-model detectors
- **Accuracy**: 92.7%
- **Data Required**: 200 traces (100 benign + 100 backdoor)
- **Use Case**: Organizations committed to a single LLM provider

Multi-LLM Deployments

- **Approach**: Model-aware detection
- **Accuracy**: 90.6% universal
- **Data Required**: 200 traces per model
- **Use Case**: Organizations using multiple LLMs for different tasks

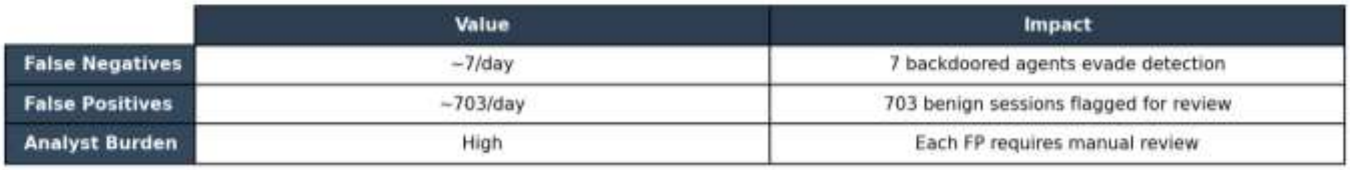
Rapid Prototyping Environments

- **Approach**: Pooled training
- **Accuracy**: 89.8% consistent
- **Data Required**: 100-200 traces per model
- **Use Case**: Organizations frequently adding/testing new LLMs

Operational Considerations

Example: Production deployment processing 10,000 agent sessions daily with 1% backdoor prevalence:

Table 12: Operational Impact (10K Sessions/day, 1% backdoor prevalence)



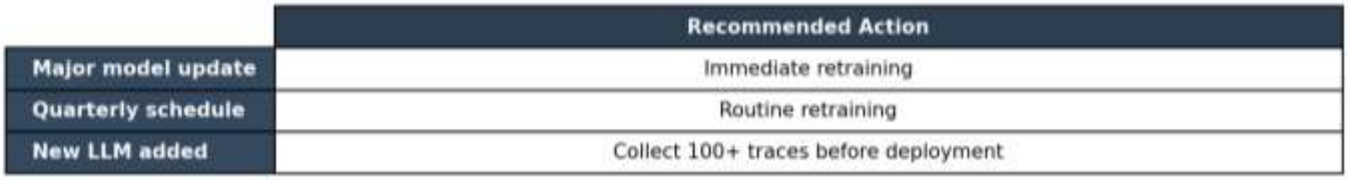
	Value	Impact
False Negatives	~7/day	7 backdoored agents evade detection
False Positives	~703/day	703 benign sessions flagged for review
Analyst Burden	High	Each FP requires manual review

Recommended Mitigations:

1. **Confidence Thresholding**: Only flag high-confidence detections
2. **Human Review**: Security analysts verify flagged traces
3. **Tiered Response**: High-confidence → immediate action; Low-confidence → monitoring

Retraining Requirements

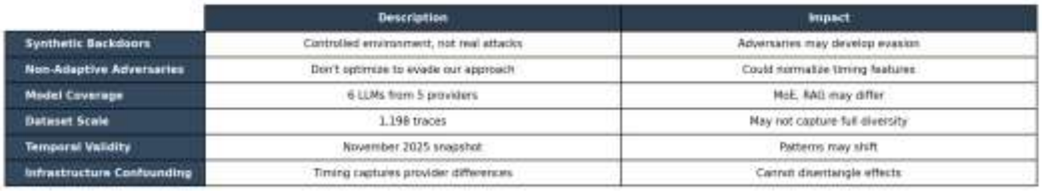
Table 13: Retraining Requirements



	Recommended Action
Major model update	Immediate retraining
Quarterly schedule	Routine retraining
New LLM added	Collect 100+ traces before deployment

IV. LIMITATIONS AND FUTURE WORK

Table 14: Current Limitations



	Description	Impact
Synthetic Backdoors	Controlled environment, not real attacks	Adversaries may develop evasion
Non-Adaptive Adversaries	Don't optimize to evade our approach	Could normalize timing features
Model Coverage	6 LLMs from 5 providers	MoE, RAG may differ
Dataset Scale	1,198 traces	May not capture full diversity
Temporal Validity	November 2025 snapshot	Patterns may shift
Infrastructure Confounding	Timing captures provider differences	Cannot disentangle effects

Future Research Directions

1. **Adaptive Adversary Evaluation**: Feature-aware attack generation, certified defenses, game-theoretic modeling
2. **Few-Shot Adaptation**: Enable protection of new LLMs with minimal training traces
3. **Model-Agnostic Features**: Alternative feature sets that generalize without explicit model identification
4. **Temporal Validity Studies**: Track detector degradation as models are updated
5. **Large-Scale Deployment**: Evaluate at 1M+ agent scale

CONCLUSION

Core Findings Remain Valid

Despite limitations, the core contributions stand:

- **The cross-LLM generalization gap exists**: 43.4 percentage points
- **Model-aware detection provides a practical solution**: 90.6% universal accuracy
- **This is a critical dimension for AI agent security** that previous single-model studies overlooked

STATEMENTS AND DECLARATIONS

Funding: This research received no external funding.

Conflicts of Interest: The Authors declare no conflict of interest.

ORCID ID: https://orcid.org/0009-0003-5131-2688

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers.

REFERENCES

[1] Baumgärtner, T., Gao, Y., Alon, D., & Metzler, D. (2024). Best-of-Venom: Attacking RLHF by injecting poisoned preference data. In First Conference on Language Modeling.

[2] Boisvert, L., Puri, A., Evuru, C. K. R., Chapados, N., Cappart, Q., Lacoste, A., Dvijotham, K., & Drouin, A. (2024). Malice in Agentland: Down the rabbit hole of backdoors in the AI supply chain. arXiv preprint arXiv:2510.05159.

[3] Bowen, D., Murphy, B., Cai, W., Khachaturov, D., Gleave, A., & Pelrine, K. (2024). Data poisoning in LLMs: Jailbreak-tuning and scaling laws. arXiv preprint arXiv:2408.02946.

[4] Carlini, N., Jagielski, M., Choquette-Choo, C. A., Paleka, D., Pearce, W., Anderson, H., Terzis, A., Thomas, K., & Tramèr, F. (2024). Poisoning web-scale training datasets is practical. In 2024 IEEE Symposium on Security and Privacy (SP) (pp. 407–425). IEEE.

[5] Chen, B., Carvalho, W., Baracaldo, N., Ludwig, H., Edwards, B., Lee, T., Molloy, I., & Srivastava, B. (2019). Activation clustering for backdoor detection. In ICML Workshop on Security and Privacy of Machine Learning.

[6] Chen, Z., Xiang, Z., Xiao, C., Song, D., & Li, B. (2024). AgentPoison: Red-teaming LLM agents via poisoning memory or knowledge bases. Advances in Neural Information Processing Systems, 37, 130185–130213.

[7] Chennabasappa, S., Nikolaidis, C., Song, D., Molnar, D., et al. (2025). LlamaFirewall: An open source guardrail system for building secure AI agents. arXiv preprint arXiv:2505.03574.

[8] Cybersecurity and Infrastructure Security Agency. (2021). Supply chain compromise (Alert AA21-008A). https://www.cisa.gov/news-events/alerts/2021/01/07/supply-chain-compromise

[9] Cybersecurity and Infrastructure Security Agency. (2024). Reported supply chain compromise affecting XZ Utils data compression library, CVE-2024-3094. CISA Alert.

[10] CrowdStrike. (2024). External technical root cause analysis — Channel File 291 incident. https://www.crowdstrike.com/wp-content/uploads/2024/08/Channel-File-291-Incident-Root-Cause-Analysis-08.06.2024.pdf

[11] Das, B. C., Amini, M. H., & Wu, Y. (2025). Security and privacy challenges of large language models: A survey. ACM Computing Surveys, 57(6), 1–39.

[12] De Chezelles, T. L. S., Gasse, M., Lacoste, A., Caccia, M., Drouin, A., Boisvert, L., et al. (2025). The BrowserGym ecosystem for web agent research. Transactions on Machine Learning Research.

[13] Drouin, A., Gasse, M., Caccia, M., Laradji, I. H., Del Verme, M., Marty, T., Boisvert, L., et al. (2024). WorkArena: How capable are web agents at solving common knowledge work tasks? arXiv preprint arXiv:2403.07718.

[14] Efron, B., & Tibshirani, R. (1986). Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy. Statistical Science, 54–75.

[15] Forrest, S., Hofmeyr, S. A., Somayaji, A., & Longstaff, T. A. (1996). A sense of self for Unix processes. In IEEE Symposium on Security and Privacy.

[16] Fu, T., Sharma, M., Torr, P., Cohen, S. B., Krueger, D., & Barez, F. (2024). PoisonBench: Assessing large language model vulnerability to data poisoning. arXiv preprint arXiv:2410.08811.

[17] Gambacorta, L., & Shreeti, V. (2025). The AI supply chain. BIS Papers.

[18] Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., et al. (2024). The Llama 3 herd of models. arXiv preprint arXiv:2407.21783.

[19] Gu, T., Dolan-Gavitt, B., & Garg, S. (2017). BadNets: Identifying vulnerabilities in the machine learning model supply chain. In NIPS Workshop on Machine Learning and Computer Security.

[20] Gu, T., Liu, K., Dolan-Gavitt, B., & Garg, S. (2019). BadNets: Evaluating backdooring attacks on deep neural networks. IEEE Access, 7, 47230–47244.

[21] Guyon, I., & Elisseeff, A. (2003). An introduction to feature selection. Journal of Machine Learning Research, 3, 1157–1182.

[22] Hubinger, E., Denison, C., Mu, J., Lambert, M., Tong, M., MacDiarmid, M., Lanham, T., Ziegler, D. M., Maxwell, T., Cheng, N., et al. (2024). Sleeper agents: Training deceptive LLMs that persist through safety training. CoRR.

[23] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning. Springer.

[24] Kandpal, N., Jagielski, M., Tramèr, F., & Carlini, N. (2023). Backdoor attacks for in-context learning with language models. arXiv preprint arXiv:2307.14692.

[25] Kazdan, J., Puri, A., Schaeffer, R., Yu, L., Cundy, C., Stanley, J., Koyejo, S., & Dvijotham, K. (2025). No, of course I can! Deeper fine-tuning attacks that bypass token-level safety mechanisms. arXiv preprint arXiv:2502.19537.

[26] Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. International Joint Conference on Artificial Intelligence (IJCAI), 14, 1137–1145.

[27] Liao, Z., Mo, L., Xu, C., Kang, M., Zhang, J., Xiao, C., Tian, Y., Li, B., & Sun, H. (2024). EIA: Environmental injection attack on generalist web agents for privacy leakage. arXiv preprint arXiv:2409.11295.

[28] Liu, X., Yu, H., Zhang, H., Xu, Y., Lei, X., Lai, H., Gu, Y., Ding, H., Men, K., Yang, K., et al. (2023). AgentBench: Evaluating LLMs as agents. arXiv preprint arXiv:2308.03688.

[29] Lyu, W., Pang, L., Ma, T., Ling, H., & Chen, C. (2024). TrojVLM: Backdoor attack against vision language models. In European Conference on Computer Vision (pp. 467–483). Springer.

[30] Mann, H. B., & Whitney, D. R. (1947). On a test of whether one of two random variables is stochastically larger than the other. The Annals of Mathematical Statistics, 50–60.

[31] Padhi, I., Nagireddy, M., Cornacchia, G., et al. (2025). Granite Guardian: Comprehensive LLM safeguarding. In Proceedings of the 2025 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 3: Industry Track) (pp. 607–615). Association for Computational Linguistics.

[32] Pang, G., Shen, C., Cao, L., & Van Den Hengel, A. (2021). Deep learning for anomaly detection: A survey. ACM Computing Surveys, 54(2), 1–38.

[33] Qi, X., Panda, A., Lyu, K., Ma, X., Roy, S., Beirami, A., Mittal, P., & Henderson, P. (2024). Safety alignment should be made more than just a few tokens deep. arXiv preprint arXiv:2406.05946.

[34] Qi, Z., Liu, X., Iong, I. L., Lai, H., Sun, X., Sun, J., et al. (2024). WebRL: Training LLM web agents via self-evolving online curriculum reinforcement learning. In The Thirteenth International Conference on Learning Representations.

[35] Shu, M., Wang, J., Zhu, C., Geiping, J., Xiao, C., & Goldstein, T. (2023). On the exploitability of instruction tuning. In Advances in Neural Information Processing Systems, 36 (pp. 61836–61856).

[36] Tran, B., Li, J., & Madry, A. (2019). Spectral signatures in backdoor attacks. In NeurIPS.

[37] Vassilev, A., Oprea, A., Fordyce, A., Anderson, H., Davies, X., & Hamin, M. (2025). Adversarial machine learning: A taxonomy and terminology of attacks and mitigations (Technical Report). National Institute of Standards and Technology.

[38] Wan, A., Wallace, E., Shen, S., & Klein, D. (2023). Poisoning language models during instruction tuning. In International Conference on Machine Learning (pp. 35413–35425). PMLR.

[39] Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H., & Zhao, B. Y. (2019). Neural Cleanse: Identifying and mitigating backdoor attacks in neural networks. In IEEE Symposium on Security and Privacy (S&P).

[40] Wang, W., & Feizi, S. (2023). Temporal robustness against data poisoning. Advances in Neural Information Processing Systems, 36, 47721–47734.

[41] Wang, Y., Xue, D., Zhang, S., & Qian, S. (2024). BadAgent: Inserting and activating backdoor attacks in LLM agents. arXiv preprint arXiv:2406.03007.

[42] Zhang, X., Zhang, Z., Ji, S., & Wang, T. (2023). Trojaning language models for fun and profit. arXiv preprint arXiv:2302.10149.

[43] Zheng, A., & Casari, A. (2018). Feature engineering for machine learning. O'Reilly Media.

SUPPLEMENTARY MATERIALS

Code Repository: https://github.com/arunsanna/cross-llm-backdoor-detection
Dataset: Available upon request for reproducible research
Author Contact: [arun.sanna@outlook.com]