Paper	Link	Year	Dataset Availability	Number of Commands	Target Systems	Testing Techniques	Key Limitations	Datasets used	Uses command output for Analysis	Labeling	Command Structure	Obfuscation Included
Anomaly Detection of Command Shell Sessions based on DistilBERT: Unsupervised and Supervised Approaches	https://arxiv.org/pdf/2310.13247	2023	Not publicly available - this was private enterprise data from JPMorgan Chase	140 common Unix commands were used for initial pattern matching. Exact full command list not provided.	Unix/Linux systems, specifically focusing on enterprise production environments. Also includes subshells for HDFS, SQL, Spark, and Python.	1. Unsupervised approach: Pretrained DistilBERT model with ensemble of 4 anomaly detectors (PCA, Isolation Forest, COPOD, Autoencoders) 2. Supervised approach: Fine-tuning DistilBERT with SetFit on labeled data 3. Testing split: 90/10 train/test split 4. Evaluation metrics: Precision, Recall, F1 score	1. Dataset is unlabeled, requiring loosely-defined labeling approaches 2. Limited to keystroke data from Unix shells (excludes network appliances and embedded systems) 3. Data cleaning and extraction challenges with mixed shell prompts and outputs 4. Unclear relationships between anomaly scores and actual suspicious activities 5. Model performance dependent on quality and quantity of labeled data 6. Focus primarily on command syntax patterns instead of semantic understanding 7. Don't handle nested commands		No	Risk (Anomaly)	Single Command	No
RACONTEUR: A Knowledgeable, Insightful, and Portable LLM-Powered Shell Command Explainer	https://arxiv.org/pdf/2409.02074	2024	Not publicly available but authors mention a new dataset will be open-source at https://raconteur-ndss.github.io/ (I cannot access it though?...)	Combined training dataset of 254,000 samples (split 9:0.5:0.5 for train/val/test) - Sources: 3 malicious command libraries (atomic-red-team, metta, reverse-shell) and 2 benign command libraries (NL2Bash and The Stack) |	- Unix/Linux shells - PowerShell - Private enterprise shell commands and systems - Special subshells (HDFS, SQL, Spark, Python)	- Quantitative Evaluation: - ROUGE, BLEU, METEOR, CIDEr metrics for behavior explainer - Top-k accuracy for intent identifier - AUC-ROC for documentation retrieval - User Study: - 52 computer science students - Evaluated comprehensiveness, correctness, preference - Baseline Comparisons: - GPT-3.5-Turbo, GPT-4, ChatGLM2-6B - Various text embedding models	- Cannot deobfuscate commands, only identify obfuscation - Limited to analyzing individual commands vs full shell sessions (Nested Commands) - Only focuses on shell logs, not other log types - Uses smaller ChatGLM2-6B (6B parameters) model - Requires documentation access for private commands	14, 15,17	No	Mitre, description, Risk	Compound Commands	Yes
PowerPeeler: Dynamic Deobfuscation of PowerShell Scripts	https://arxiv.org/pdf/2406.04027	2024	Not publicly available	D-Script contains 4,264 obfuscated script files D-Cmdline contains 381 obfuscated samples that use the PowerShell CLI	- Windows - PowerShell	- Comparing "hand-crafted" obfuscated commands vs generated commands - Semantic consistency: If the generated command (key APIs) matches the original sample - Number of instructions generated - Comparing obfuscated commands using Invoke-Obfuscation tool vs generated commands - Quantity of sensitive data (like ips, urls, file paths and reg keys) recovered	- Unreached code is not analyzed - Supports only powershell-v7 - Limited by a 2-min execution timeout - It uses static analysis for quality, not for its analysis	-	Yes	Risk	Scripts	Yes
CmdCaliper: A Semantic-Aware Command-Line Embedding Model and Dataset for Security Research	https://arxiv.org/pdf/2411.01176	2024	CyPher	CyPHER consists of 28,520 similar command-line pairs, totaling 55,909 unique CLI Testing: Splunk Attack Data, Atomic Red (Number unknown)	- Windows - PowerShell	- Statistical Analysis (# CLI pairs, uniqueness, statistics on lenght) - Semantic Similarity and Diversity: Rouge-L overlap score - Similarity con CLI pair quality: By comparing the embeddings of their explanations and comparing them to random pairs - LLM Pool Effectiveness: By clustering the explanations of the generated Commands and messuring the coverage rate.	- Only supports powershell - Vulnerable to Command-Line obfuscation - Don't handle nested commands	12, 15, 17	No	Description, Risk	Single Command	Yes
Command-line Risk Classification using Transformer-based Neural Architectures	https://arxiv.org/pdf/2412.01655	2024	Not publicly available - Created from internal cloud production systems. Pre-training uses GitHub random commands (with descriptions)	Pretraining: 71164 Bash scripts (It uses man pages for categorization) Fine-Tunning: 47,158 scripts labeled	Bash	Base evaluation: accuracy, precision, recall, and F1-score Comparisson with existing methods Accuracy with rule-base systems: Compare predictions of this system with online existing rule-based systems.	- Vulnerable to CL obfuscation - Don't handle nested commands - Only use static analysis - Expert knowledge is required for training		No	Risk	Single Command	Mentioned, but not a focus
AMSI-Based Detection of Malicious PowerShell Code Using Contextual Embeddings	https://arxiv.org/pdf/1905.09538	2020	Yes - partially. The pretrained embedding is publicly available at https://bitbucket.org/amirubin87/powershellembedding	Not explicitly specified, but mentions analyzing 373,594,394 AMSI scan events containing PowerShell code	Windows systems running PowerShell with AMSI enabled (Windows 10)	Split labeled dataset into training (May-July 2018) and test sets (August-October 2018) 3-fold cross-validation on training set Multiple model architectures tested (CNN, CNN-RNN, Token-Char) Evaluation metrics: TPR, FPR, AUC Comparison against baseline NLP approaches	Requires AMSI to be enabled and functioning Can be bypassed if AMSI is disabled Limited to PowerShell code only Requires administrative privileges for some detections Model can be evaded through specific obfuscation techniques Training data scarcity for labeled malicious samples	Labeled dataset: 116,976 PS code instances (5,383 malicious, 111,593 benign) Unlabeled dataset: 368K PowerShell scripts/modules from public repositories	No	Binary (Malicious/Benign)	Full PowerShell code including scripts and modules, not just command-lines	Yes