

Model-agnostic clean-label backdoor mitigation in cybersecurity environments

Giorgio Severi*, Simona Boboila*, John Holodnak[†], Kendra Kratkiewicz[†]
Rauf Izmailov[‡], Michael J. De Lucia[§], Alina Oprea*

*Northeastern University, [†]MIT Lincoln Laboratory, [‡]Peraton Labs, [§]DEVCOM Army Research Laboratory

Abstract—The training phase of machine learning models is a delicate step, especially in cybersecurity contexts. Recent research has surfaced a series of insidious training-time attacks that inject backdoors in models designed for security classification tasks without altering the training labels. With this work, we propose new techniques that leverage insights in cybersecurity threat models to effectively mitigate these clean-label poisoning attacks, while preserving the model utility. By performing density-based clustering on a carefully chosen feature subspace, and progressively isolating the suspicious clusters through a novel iterative scoring procedure, our defensive mechanism can mitigate the attacks without requiring many of the common assumptions in the existing backdoor defense literature.

I. INTRODUCTION

Machine learning (ML) models underpin an ever-growing variety of software systems, including cybersecurity solutions designed to thwart active adversaries [1], [2], [3]. The deployment of models in security-sensitive contexts increases the relevance of adversarial ML risks, both during inference, *evasion attacks*, and during model training, *poisoning attacks*. The risk of adversarial interference during training is further enhanced by recent trends, such as the growing size of data sets and the increasing reliance on crowd-sourcing of data [4], leading to heightened public awareness [5], [6].

This work addresses backdoor attacks [7] in cybersecurity settings [8], [9], [10], where a model learns to associate an adversary-chosen data pattern (*trigger*) with a target class. During inference, any input containing the trigger is misclassified accordingly. These attacks are especially concerning in cybersecurity, where models are trained on large, externally sourced datasets, increasing the risk of data manipulation. Their stealthy nature—preserving accuracy on clean inputs—makes detection and mitigation particularly difficult. We examine an insidious clean-label backdoor attack targeting automated traffic analysis systems [10]. This attack injects triggers into a small number of benign training samples without altering labels, is model-agnostic, and is applicable to a variety of data modalities in cybersecurity.

Our defense, outlined in Figure 2, leverages the information asymmetry between attacker and defender to isolate poisoned samples while preserving *clean* data for training. It uses an iterative scoring process on clustered data in a selected feature subspace, applying multiple remediation strategies to

suppress poisoned clusters. Our method reduces backdoor attack success by up to 90% while maintaining high utility, even at poisoning rates as high as 5%. Unlike most existing defenses, it requires no clean reference data or assumptions about model architecture, making it broadly applicable.

In summary, we make the following contributions.

- We introduce a new defense against *clean-label* backdoor attacks in cybersecurity, removing the need for trusted data or model-specific assumptions.
- We thoroughly evaluate our method against existing attacks on different model architectures, showing strong mitigation with minimal impact on model utility (F1 score) and false positive rate.
- We ensure reproducibility by using public datasets and open-source attack implementations and will release the code used in our experiments.¹

II. RELATED WORK

Backdoor poisoning attacks embed a trigger pattern into a small fraction of training data to hijack model predictions while leaving clean-test accuracy intact. Early work by Gu et al. demonstrated pixel-based triggers in vision tasks [7], and subsequent clean-label methods removed the need for label tampering [11], [12]. In cybersecurity domains, packet-level and feature-space poisoning, as well as label-flipping, have been explored [13], [14], including clean-label attacks guided by model explanations against malware classifiers [8] and complex network-flow triggers [10].

A. Mitigating backdoor attacks

Defenses span several paradigms. *Certified* methods offer provable guarantees via outlier removal and ensemble partitioning [15], [16] or randomized smoothing [17], though they often require impractically large ensembles. *Detection and Filtering* approaches analyze internal model representations – activation clustering and spectral-signature removal in neural embeddings [18], [19], [20] – but are limited to neural networks. *Model Purification* techniques prune or fine-tune backdoored networks using small clean sets [21], [22], but are highly sensitive to the available clean data, and may

¹https://github.com/ClonedOne/cyber_clean_label_backdoor_mitigation

struggle when task and backdoor signals intertwine. *Anti-Backdoor Learning* identifies and unlearns fast-converging poisoned points early in training, refined through self and semi-supervised stages [23], [24], [25]. Other methods include weight- and activation-based analyses (ABS [26], MNTD [27]), trigger-reverse-engineering (NeuralCleanse [28]), topological embedding analyses (TED [29]), and high-cost ensemble sanitization in cybersecurity via NestedTraining [30], [31]. Our approach involves data sanitization *before training the model*, to identify and remove poisoned samples in a model-agnostic fashion.

III. PROBLEM STATEMENT

Defending ML models in cybersecurity presents unique challenges and opportunities. Effective defenses must work across diverse models like decision-tree ensembles and neural networks used in security applications such as malicious domain classifiers [2] and malware classification [32]. Moreover, the availability of clean reference data, a common assumption in literature [22], [33], is not granted. While obtaining clean images or text is relatively inexpensive (crowd-sourced annotations and filtering), verifying cybersecurity data requires expensive domain expert analysis.

Cybersecurity data imposes intrinsic constraints that also limit the attacker, enabling defenders to make assumptions unlikely to hold in other domains. Notably, prior attacks [8], [10] emphasize the use of feature importance estimation in crafting effective triggers – an insight defenders can exploit to narrow the detection scope in feature space. Stealthy attackers often use clean-label poisoning, blending malicious samples with benign data without altering labels, as they typically lack control over labeling. This tactic helps poisoned points blend into the diverse benign distribution. While such variability complicates detection, it also justifies the assumption that poisoned samples form a subset of the benign class.

A. Threat model

We target binary-classification models in security settings that operate on hand-crafted, tabular features. An adversary seeks to poison training inputs so that malicious samples evade detection (e.g., harmful traffic passes unnoticed), while the defender aims to thwart such attacks without degrading overall performance—especially keeping false-positive rates low to avoid client disruption and analyst overload.

We assume attackers know the feature representation and can either query the victim model or train a surrogate to estimate feature importance; they may inject a small number of poisons but cannot alter labels [8], [10]. Unlike prior work, we do not presume access to a pristine training set – acquiring expert-labeled data is costly in security – yet we allow defenders to cluster the training data and retrain multiple, relatively lightweight models (e.g., decision-tree ensembles), which is feasible given their modest size.

B. Challenges of existing defenses

Most existing backdoor defenses are ill-suited for cybersecurity. In particular we observe:

a) *Different data modalities and model architectures:*

Most exiting mitigation approaches have been designed for computer vision (CV), where CNNs are the state-of-the-art architecture [18], [19], [20], [29]. Representative approaches for CV backdoor defenses are activation clustering [18], spectral signatures [19], and SPECTRE [20] that perform outlier detection in the CNN representation space via clustering, SVD decomposition, or robust statistics. Prior work [8] tried to adapt these methods and showed that these defenses are not effective when they do not operate in the model’s representation space.

b) *Coarse-grained binary labels:* A typical cybersecurity task² is to distinguish malicious and benign samples, resulting in a binary classification problem. Techniques that look for shortcuts between classes in latent space and aim to identify a target attack label, such as Neural Cleanse [28] and ABS [26], require finer-grained labeling of the training data and thus cannot be readily adapted to our setting.

c) *Hard constraints on false positives:* Models trained for cybersecurity tasks have hard constraints on false positives to be deployed in production [2]. This rules out the application of certified defenses, which largely degrade model utility.

C. Evaluating existing defenses

We evaluate here some well-known defensive approaches: Spectral Signatures, SPECTRE, Activation Clustering, and Selective Amnesia using the state-of-the-art backdoor attack by Severi et al. [10] on the CTU-13 Botnet detection task and a fully connected neural network model. The attack can be configured to use either SHAP or Entropy for selecting features for inclusion in the trigger.

1) *Limitations of Spectral Signatures and SPECTRE:* Spectral Signatures (SpS) [19] and SPECTRE [20] identify anomalies by projecting embedded data into lower-dimensional subspaces. SPECTRE whitens the data using robustly estimated means and covariances before applying PCA. Table I reports the number of poisons identified as more points are removed. Neither method effectively removed contaminants, and SPECTRE frequently failed to complete. This suggests that the embeddings produced by CNNs applied to image data are structurally different from those produced by the shallow, fully connected networks used in cybersecurity applications.

2) *Limitations of Activation Clustering:* Activation clustering [18] detects poisoned points by applying k -means clustering ($k = 2$) to the activations of the last hidden layer of the victim neural network, followed by analysis to determine which cluster to discard. We evaluated this approach across poisoning rates from 0.1% to 5%, with both SHAP and

²Binary classification tasks are common in scenarios where fast identification of threats is paramount.

TABLE I: Mean number of poisons identified by SpS and SPECTRE for a neural network trained on the CTU-13 data, across five trials. Standard deviation in parentheses. A dash means that the defense failed to complete in at least four trials.

(a) 0.5% poisoning (714 poisons)			(b) 5% poisoning (7152 poisons)		
Removed	SpS	SPECTRE	Removed	SpS	SPECTRE
200	1.6 (1.51)	1.8 (1.30)	200	13.0 (2.12)	16.0 (2.74)
400	2.0 (1.87)	2.2 (1.09)	400	21.2 (4.49)	32.0 (4.24)
600	2.8 (2.38)	-	600	31.4 (3.05)	-
800	4.0 (2.64)	-	800	43.0 (4.90)	-
1000	4.8 (3.03)	-	1000	54.4 (5.90)	-

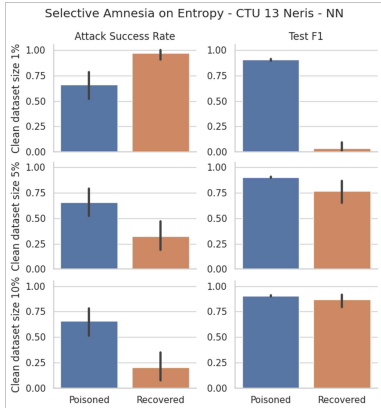


Fig. 1: Selective Amnesia defense, CTU-13 Neris botnet classifier. Comparison of attack success rates before and after recovery, and F1 score on test data, for different sizes of the clean dataset. Attack run with Entropy feature selection.

Entropy-based feature selection. In all scenarios, Activation Clustering never identified the contaminants in the training data. Clustering consistently produced small suspicious clusters (20–100 points), far fewer than the actual number of poisons. This likely stems from reduced clustering effectiveness on representations from small fully connected networks, as opposed to the CNN architectures used in prior work.

3) *Limitations of Selective Amnesia*: Selective Amnesia (SEAM) [22] mitigates backdoors by inducing catastrophic forgetting through fine-tuning on randomly assigned incorrect labels, erasing both the primary task and backdoor associations. The model is then retrained on a held-out clean dataset to recover accuracy. Originally developed for multi-class vision tasks, we adapted SEAM for our binary classification setting in the CTU-13 Botnet detection task (entropy feature selection), applying it only to the neural network model due to the lack of fine-tuning mechanisms for tree-based classifiers.

As shown in Figure 1, SEAM’s effectiveness depends heavily on the size of the clean dataset. A 1% recovery set fails to mitigate the attack, while 5% reduces the attack success rate from 0.65 to 0.30. Even at 10%, the defense does not fully eliminate the backdoor. Given the high cost of obtaining clean data in cybersecurity, these results highlight the need for defenses that do not rely on clean datasets.

IV. DEFENSE STRATEGY

We detail our sanitization procedure (Algorithm 1) below.

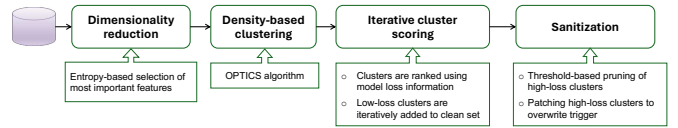


Fig. 2: Pipeline of our defense strategy.

A. Dimensionality reduction

To mitigate the curse of dimensionality – high-dimensional data points become increasingly sparse and hard to cluster – we project each sample onto a small set \mathcal{F} of the most influential features. Observing that effective backdoor attacks target features with outsized decision impact [8], [10], we train a surrogate decision tree and use its entropy-based importance scores to rank and retain the top $|\mathcal{F}|$ features. The choice of $|\mathcal{F}|$ balances coverage of potential trigger dimensions against clustering tractability—adversaries typically exploit only the highest-ranked features—so it can be tuned to yield a sensible number of clusters. In our evaluation, we reduced the original $\sim 1,100$ network-traffic features to 4.

B. Clustering

Our defense isolates poisoned samples via density-based clustering in the reduced feature space. Unlike neural-network-specific methods that cluster hidden-layer activations [18], we apply OPTICS [34] directly on the top \mathcal{F} features of the benign samples. This method discovers high-density cores and expands clusters, automatically inferring the number of clusters and handling variable densities. This concentrates the poisons into a few clusters, enabling a follow-on scoring and filtering stage to pinpoint and remove corrupted data.

C. Iterative cluster scoring

We develop an iterative scoring procedure, presented in Algorithm 1, where clean clusters are progressively identified and added to a clean data set. It starts by selecting the largest cluster C_0 out of all clusters C identified by OPTICS on the benign set, and merging it with the malicious points, to generate a temporary clean training set, $D_{clean} \leftarrow C_0 \cup D_{y=1}$. Next, we train a clean model f on D_{clean} , and evaluate f ’s average loss on each remaining cluster $C_i \in C \setminus C_0$. We rank the clusters C_i by loss. Typically, the lower the loss, the closer the cluster is to the training data D_{clean} .

Successively, we take the clusters with lowest average loss for the clean model and add their data points to the clean training set D_{train} . Naturally, considering each cluster individually would provide the defender with fine-grained loss information but it would also require the defender to train a new model for each cluster, which could get prohibitively expensive. We address this issue by considering windows of clusters of fixed size w – we experimentally chose $w = 5\%$, allowing the defender to trade-off computation time for fine-grained loss information. Then we re-train the surrogate model f on this new dataset and score the remaining clusters. This process iteratively pushes the clusters that are less similar to

Algorithm 1: Mitigation Procedure

Data: D : training data set in feature space;
 $D_{y=0}$: the subset of D labeled benign (examples may include trigger);
 $D_{y=1}$: the subset of D labeled malicious

1 Procedure Defense (D):
2 $\mathcal{F} \leftarrow \text{DIMENSIONALITY_REDUCTION}(D_{y=0})$
3 $C \leftarrow \text{DENSITY_BASED_CLUSTERING}(\mathcal{F})$
4 $C_0 \leftarrow \text{max}(C)$
5 **do**
6 $D_{clean} \leftarrow C_0 \cup D_{y=1}$
7 $f \leftarrow \text{TRAIN}(D_{clean})$
8 $C_r \leftarrow C \setminus C_0$
9 $\mathcal{L} = \{\}$
10 **for** $i \in \text{range}(|C_r|)$ **do**
11 $\mathcal{L}[i] = \text{COMPUTE_LOSS}(f, C_i)$
12 **end**
13 $C_l \leftarrow \text{lowest loss cluster(s) from } \mathcal{L}$
14 $C_0 \leftarrow C_0 \cup C_l$
15 **while** $C_0 \neq D \mid \text{stop_condition}$;
16 $C_r \leftarrow C \setminus C_0$
17 $C_r \leftarrow \text{PATCH_OR_DISCARD}(C_r)$
18 $D_{clean} \leftarrow C_0 \cup C_r \cup D_{y=1}$
19 $f \leftarrow \text{TRAIN}(D_{clean})$
20 **return** f

the data assumed to be clean to the bottom of the list, therefore isolating the clusters containing poisoned data points.

At this stage the defender could opt for a fixed threshold filtering strategy, by stopping the iterative training and scoring process after a fixed percentage of clusters has been added to D_{clean} . In our experiment, we set this fixed threshold to 80% of the clusters, as we empirically find it to be quite effective.

D. Sanitization of high-loss clusters

After clustering isolates suspicious groups, we remediate them via two strategies. The simplest is to remove any cluster not flagged as clean, thereby eliminating all poisons at the cost of potentially discarding rare but valid patterns and degrading utility on uncommon test cases. Alternatively, we “patch” each point in a suspect cluster by overwriting its top $|\mathcal{P}|$ most important features (where $|\mathcal{P}| \geq |\mathcal{F}|$) with values sampled from the clean portion of the data. In our experiments we randomly draw replacement values from D_{clean} (the set of clean clusters after reaching 80% coverage) for each patched feature. This preserves the bulk of the training data while neutralizing the backdoor, and may be further enhanced in future work by employing generative tabular models (e.g., diffusion-based samplers) to produce more realistic replacements.

V. EVALUATION

We evaluate our defense on the CTU-13 Neris botnet detection task [35], using Zeek-extracted NetFlow logs aggregated into 30-second windows. Each window is converted into a feature vector of statistics—connection counts by protocol and state, distinct external IPs per internal IP, and min/mean/max packet and byte volumes—mirroring prior work [36], [10].

We test against two feature-selection attacks—an entropy-based surrogate decision tree and a SHAP-guided method [37]—and two trigger schemes: (1) a full trigger that embeds contiguous connections into the most important

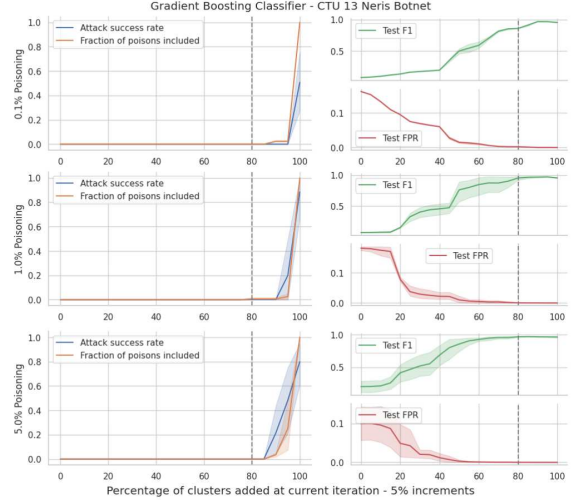


Fig. 3: Iterative scoring on CTU-13 botnet for the *gradient boosting* model. The plot shows average metrics for a set of experiments: SHAP and Entropy attacker feature selection, for the Full trigger attack, at different poisoning rates.

features, and (2) a stealthy, Bayesian-network-generated trigger that respects traffic dependencies. We report attack success rate (ASR), defined as the percentage of originally malicious test points misclassified as benign, alongside clean-data utility via F1 score and false-positive rate.

A. Evaluation on network traffic classification

Figure 3 shows the dynamics of the tracked metrics across the iterations, assuming a window size of 5% of the clusters. The reported metrics are averages over multiple experiments, with two feature selection strategies (Entropy and SHAP), for the Full trigger attack. Results are shown at 5 different poisoning percentages, and each experiment is repeated 5 times with different random seeds.

1) *Fixed threshold filtering:* For both gradient boosting model and neural network, interrupting the training when 80% of the clusters has been added to the training set represents a good baseline heuristic that filters out the vast majority of poisoning points. We also note that the attack’s poisoning fraction has little influence on the effectiveness of this procedure.

The generated trigger, on the other hand, is stealthier, and a larger fraction of poisoning points manages to slip through the filtering process at these pre-defined thresholds. However, since this strategy is designed for stealthiness rather than effectiveness, its success rate is limited even if a fraction of the contaminants ends up in the training data. As observed in [10], the generated neural network trigger acts as an adversarial example, causing misclassifications without poisoning. This falls outside our defense’s scope, which targets backdoor poisoning attacks.

While effective at removing the poisons (ASR ranging from 0.0% to 6.45%), this baseline remediation strategy may reduce the utility of the models as a side-effect of discarding entire

TABLE II: Average model utility metrics on CTU-13. Results reported for different victim architectures, at different poisoning percentages. All results are averages of 10 runs, with two attack strategies and 5 random seeds.

Model type	Poisoning %	F1 at 80.0%	FPR at 80.0%	Poisons in D_{clean} at 80.0%
Full trigger				
Neural Network	0.1%	48.19%	1.45%	10.99%
	0.5%	45.97%	1.62%	31.71%
	1%	52.31%	1.30%	13.89%
	2%	49.33%	1.36%	12.03%
	5%	64.55%	0.83%	20.51%
Gradient Boosting	0.1%	85.93%	0.21%	0.00%
	0.5%	94.84%	0.04%	2.72%
	1%	95.43%	0.04%	0.62%
	2%	87.13%	0.18%	0.01%
	5%	96.50%	0.02%	0.03%
Generated trigger				
Gradient Boosting	0.1%	93.26%	0.07%	30.99%
	0.5%	88.51%	0.18%	49.64%
	1%	93.16%	0.08%	48.59%
	2%	89.12%	0.17%	38.49%
	5%	90.10%	0.14%	65.36%

TABLE III: Comparison of patching and filtering sanitization approaches at fixed threshold = 80%. Gradient boosting model on CTU-13. Also showing the Base ASR value for the undefended attack. Results are averages of 5 runs on different random seeds, for two attack strategies Entropy and SHAP.

Sanitization	Poisoning	Base ASR	ASR	Test FPR	Test F1
Filtering	0.1%	50.70%	0.00%	0.21%	85.93%
	0.5%	85.80%	6.45%	0.04%	94.84%
	1%	88.45%	0.15%	0.04%	95.43%
	2%	85.75%	0.00%	0.18%	87.13%
	5%	79.90%	0.00%	0.02%	96.50%
Patching	0.1%	50.70%	0.00%	0.08%	90.61%
	0.5%	85.80%	10.95%	0.05%	93.19%
	1%	88.45%	9.75%	0.03%	93.37%
	2%	85.75%	9.30%	0.05%	92.64%
	5%	79.90%	10.95%	0.00%	95.27%

clusters above the threshold (including their clean data). The F1 and FPR utility metrics, reported in Table II, average from 0.02% to 0.21% for FPR, and 86% to 97% for F1.

2) *Patching*: The patching-based sanitization strategy addresses the degradation in utility metrics. Considering the same threshold at 80%, we can directly compare the effects of patching to filtering for the experiment with the gradient boosting model on CTU-13. Table III shows that using patching generally leads to higher average values of the F1 score on test data (91% - 95%), and a lower false positive rate (0% - 0.08%). On the other hand, as expected, patching is slightly less effective than complete filtering in reducing the attack success (0% - 11% ASR). Therefore, the defender can adopt the patching approach if they want to trade-off some defensive effectiveness for reduced degradation in model utility.

B. Adaptive adversaries

Our defense is tailored to counter factors that amplify clean-label poisoning attacks on tabular data. As with any heuristic defense, a knowledgeable adversary may attempt to adapt, for instance by: (i) selecting a different (less important) subset of features; (ii) ensuring a percentage of the poisons ends up in the largest cluster; (iii) using different triggers for different subsets of poison points, so that they end up in many different clusters; (iv) injecting high-loss noise into multiple clusters to keep poisoned clusters below the 80% threshold.

Strategies (ii) and (iii) are challenging in practice. In (iii), distributing different triggers reduces their visibility during training, weakening their association with the target class. For (ii), accurately estimating training data density to position poison points within dense regions—while maintaining attack success rate (ASR)—is difficult. While stealthier attacks aim to blend with benign data, none actively target high-density regions, and the success of such a variant is uncertain. Strategy (iv) is easier to implement but requires extensive modification of the training set, compromising stealth.

Strategy (i) may allow evasion, as our defense assumes the trigger lies in the most relevant features. An attacker could select features that deliberately exclude the top $|\mathcal{F}|$. However, this is non-trivial: identifying a large, manipulable, and disjoint subset from key features is difficult. Even if successful, targeting less relevant features likely reduces ASR. Thus, the attacker must balance stealth and effectiveness carefully.

VI. DISCUSSION AND CONCLUSIONS

Defensive methods in adversarial machine learning inherently face limitations, and protecting against arbitrary poisoning remains an open challenge. We propose a mitigation strategy for clean-label backdoor attacks on cybersecurity classifiers that removes common constraints of prior defenses—namely, the need for clean reference data and architectural assumptions. Our method significantly reduces attack success while maintaining high model utility and low false positive rates.

Our mitigation introduces some overhead in training, but training multiple instances on incrementally larger datasets is computationally reasonable for low-cost models like decision trees and small neural networks. Our pre-training defense can be combined with other poisoning mitigations during and post-training in line with a defense-in-depth security strategy. We leave a detailed design of such a strategy for poisoning mitigation to future work.

ACKNOWLEDGMENTS

DISTRIBUTION STATEMENT A. Approved for public release. Distribution is unlimited. This material is based upon work supported by the Under Secretary of Defense for Research and Engineering under Air Force Contract No. FA8702-15-D-0001 or FA8702-25-D-B002. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Under Secretary of Defense for Research and Engineering. Delivered to the U.S. Government with Unlimited Rights, as defined in DFARS Part 252.227-7013 or 7014 (Feb 2014). Notwithstanding any copyright notice, U.S. Government rights in this work are defined by DFARS 252.227-7013 or DFARS 252.227-7014 as detailed above. Use of this work other than as specifically authorized by the U.S. Government may violate any copyrights that exist in this work. This research was sponsored by the NSF awards CNS-2312875 and CNS-2331081, the U.S. Army Combat Capabilities Development Command Army Research Laboratory (DEVCOM ARL) under Cooperative Agreement Number W911NF-24-2-0115, and the Department of Defense Multidisciplinary Research Program of the University Research Initiative (MURI) under contract W911NF-21-1-0322.

REFERENCES

- [1] M. Antonakakis, R. Perdisci, W. Lee, N. Vasiloglou, and D. Dagon, "Detecting malware domains at the upper DNS hierarchy," in *Proceedings of the 20th USENIX Conference on Security*, ser. SEC'11. USA: USENIX Association, 2011, p. 27.
- [2] A. Oprea, Z. Li, R. Norris, and K. Bowers, "MADE: Security analytics for enterprise threat detection," in *Proceedings of Annual Computer Security Applications Conference*, ser. ACSAC, 2018.
- [3] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection," in *Proceedings 2018 Network and Distributed System Security Symposium*. San Diego, CA: Internet Society, 2018.
- [4] N. Carlini, M. Jagielski, C. A. Choquette-Choo, D. Paleka, W. Pearce, H. Anderson, A. Terzis, K. Thomas, and F. Tramèr, "Poisoning web-scale training datasets is practical," in *2024 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 2024, pp. 176–176.
- [5] R. S. Siva Kumar, M. Nyström, J. Lambert, A. Marshall, M. Goertzel, A. Comissioner, M. Swann, and S. Xia, "Adversarial Machine Learning-Industry Perspectives," in *2020 IEEE Security and Privacy Workshops (SPW)*, May 2020, pp. 69–75.
- [6] M. ATLAS. (1999) VirusTotal Poisoning 2020. [Online]. Available: <https://atlas.mitre.org/studies/AML.CS0002>
- [7] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg, "BadNets: Evaluating Backdooring Attacks on Deep Neural Networks," *IEEE Access*, vol. 7, pp. 47 230–47 244, 2019.
- [8] G. Severi, J. Meyer, S. Coull, and A. Oprea, "Explanation-Guided Backdoor Poisoning Attacks Against Malware Classifiers," in *30th USENIX Security Symposium (USENIX Security 21)*, 2021, pp. 1487–1504.
- [9] L. Yang, Z. Chen, J. Cortellazzi, F. Pendlebury, K. Tu, F. Pierazzi, L. Cavallaro, and G. Wang, "Jigsaw puzzle: Selective backdoor attack to subvert malware classifiers," in *2023 IEEE Symposium on Security and Privacy (SP)*. Los Alamitos, CA, USA: IEEE Computer Society, may 2023, pp. 719–736.
- [10] G. Severi, S. Boboila, A. Oprea, J. Holodnak, K. Kratkiewicz, and J. Matterer, "Poisoning Network Flow Classifiers," in *Proceedings of the 39th Annual Computer Security Applications Conference*, ser. ACSAC '23. New York, NY, USA: Association for Computing Machinery, Dec. 2023, pp. 337–351.
- [11] A. Shafahi, W. R. Huang, M. Najibi, O. Suci, C. Studer, T. Dumitras, and T. Goldstein, "Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks," in *Advances in Neural Information Processing Systems*, Apr. 2018.
- [12] A. Turner, D. Tsipras, and A. Madry, "Label-consistent backdoor attacks," 2019.
- [13] J. T. Holodnak, O. Brown, J. Matterer, and A. Lemke, "Backdoor Poisoning of Encrypted Traffic Classifiers," in *2022 IEEE International Conference on Data Mining Workshops (ICDMW)*, Nov. 2022, pp. 577–585.
- [14] P. Li, Q. Liu, W. Zhao, D. Wang, and S. Wang, "Chronic poisoning against machine learning based idss using edge pattern detection," in *2018 IEEE International Conference on Communications (ICC)*, 2018, pp. 1–7.
- [15] J. Steinhardt, P. W. Koh, and P. Liang, "Certified defenses for data poisoning attacks," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 3520–3532.
- [16] A. Levine and S. Feizi, "Deep partition aggregation: Provable defenses against general poisoning attacks," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.
- [17] B. Wang, X. Cao, J. Jia, and N. Z. Gong, "n certifying robustness against backdoor attacks via randomized smoothing," in *CVPR 2020 Workshop on Adversarial Machine Learning in Computer Vision*, 2020.
- [18] B. Chen, W. Carvalho, N. Baracaldo, H. Ludwig, B. Edwards, T. Lee, I. Molloy, and B. Srivastava, "Detecting backdoor attacks on deep neural networks by activation clustering," in *Workshop on Artificial Intelligence Safety*. CEUR-WS, Jan. 2019.
- [19] B. Tran, J. Li, and A. Madry, "Spectral signatures in backdoor attacks," in *Neural Information Processing Systems*, 2018.
- [20] J. Hayase, W. Kong, R. Somani, and S. Oh, "Spectre: defending against backdoor attacks using robust statistics," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, 18–24 Jul 2021, pp. 4129–4139.
- [21] K. Liu, B. Dolan-Gavitt, and S. Garg, "Fine-pruning: Defending against backdooring attacks on deep neural networks," in *Research in Attacks, Intrusions, and Defenses*, 2018, pp. 273–294.
- [22] R. Zhu, D. Tang, S. Tang, X. Wang, and H. Tang, "Selective Amnesia: On Efficient, High-Fidelity and Blind Suppression of Backdoor Effects in Trojaned Machine Learning Models," in *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 2023, pp. 682–700.
- [23] Y. Li, X. Lyu, N. Koren, L. Lyu, B. Li, and X. Ma, "Anti-backdoor learning: Training clean models on poisoned data," *Advances in Neural Information Processing Systems*, 2021.
- [24] K. Huang, Y. Li, B. Wu, Z. Qin, and K. Ren, "Backdoor defense via decoupling the training process," in *ICLR*, 2022.
- [25] W. Chen, B. Wu, and H. Wang, "Effective Backdoor Defense by Exploiting Sensitivity of Poisoned Samples," in *Advances in Neural Information Processing Systems*, vol. 35, Dec. 2022, pp. 9727–9737.
- [26] Y. Liu, W.-C. Lee, G. Tao, S. Ma, Y. Aafer, and X. Zhang, "ABS: Scanning neural networks for back-doors by artificial brain stimulation," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. New York, NY, USA: Association for Computing Machinery, 2019.
- [27] X. Xu, Q. Wang, H. Li, N. Borisov, C. A. Gunter, and B. Li, "Detecting AI trojans using meta neural analysis," in *2021 IEEE Symposium on Security and Privacy (SP)*, 2021.
- [28] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, "Neural Cleanse: Identifying and mitigating backdoor attacks in neural networks," in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 707–723.
- [29] X. Mo, Y. Zhang, L. Zhang, W. Luo, N. Sun, S. Hu, S. Gao, and Y. Xiang, "Robust backdoor detection for deep learning via topological evolution dynamics," in *2024 IEEE Symposium on Security and Privacy (SP)*. Los Alamitos, CA, USA: IEEE Computer Society, 2024, pp. 174–174.
- [30] S. Venkatesan, H. Sikka, R. Izmailov, R. Chadha, A. Oprea, and M. J. de Lucia, "Poisoning attacks and data sanitization mitigations for machine learning models in network intrusion detection systems," in *MILCOM 2021 - 2021 IEEE Military Communications Conference (MILCOM)*, 2021, pp. 874–879.
- [31] S. Ho, A. Reddy, S. Venkatesan, R. Izmailov, R. Chadha, and A. Oprea, "Data sanitization approach to mitigate clean-label attacks against malware detection systems," in *MILCOM 2022 - 2022 IEEE Military Communications Conference (MILCOM)*, 2022, pp. 993–998.
- [32] H. S. Anderson and P. Roth, "Ember: an open dataset for training static pe malware machine learning models," *arXiv preprint arXiv:1804.04637*, 2018.
- [33] X. Qi, T. Xie, J. T. Wang, T. Wu, S. Mahlouljifar, and P. Mittal, "Towards a proactive ML approach for detecting backdoor poison samples," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 1685–1702.
- [34] M. Ankerst, M. M. Breunig, H. Peter Kriegel, and J. Sander, "Optics: Ordering points to identify the clustering structure." ACM Press, 1999, pp. 49–60.
- [35] S. García, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods," *Computers and Security*, vol. 45, pp. 100–123, Sep. 2014.
- [36] T. Ongun, O. Spohngeleit, B. Miller, S. Boboila, A. Oprea, T. Eliassi-Rad, J. Hiser, A. Nottingham, J. Davidson, and M. Veeraraghavan, "PORTFILER: Port-Level Network Profiling for Self-Propagating Malware Detection," in *2021 IEEE Conference on Communications and Network Security (CNS)*, Oct. 2021, pp. 182–190.
- [37] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.