FALCON: Adaptive Cross-Domain APT Attack Investigation with Federated Causal Learning

Jialu Tang[1] Yali Gao[1,*] Xiaoyong Li[1] Jiawei Li[1] Shui Yu[2] Binxing Fang[3]

[1]School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing, China
[2]School of Computer Science, University of Technology Sydney, Ultimo, NSW, Australia
[3]Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou, China
[*]Corresponding author, email <gaoyali@bupt.edu.cn>

Abstract

With the extensive deployment and application of Internet of Things (IoT) devices, vulnerable edge nodes have emerged as primary targets for Advanced Persistent Threat (APT) attacks. Attackers compromise IoT terminal devices to establish an initial foothold and subsequently exploit lateral movement techniques to progressively infiltrate core business networks. Prior investigation methods struggle with fragmented threat intelligence and sparse attack samples in heterogeneous audit logs, resulting in incomplete attack chain reconstruction and high false positives. We propose a novel approach to APT attack investigation, FALCON, which captures complex causal relationships between entities from discrete audit logs and constructs cross-domain provenance graphs, enabling rapid and accurate identification of potential APT activities. FALCON trains an adaptive edge-side local model with cross-domain behavior sequences containing extensive and remote contextual information, and employs a bidirectional transformer pre-trained model to learn latent representations from unlabeled sequences. To the best of our knowledge, FALCON is the first APT investigation method to conduct causal provenance based on cross-domain audit logs while ensuring privacy protection. The experimental results demonstrate that FALCON effectively detects APT attacks with accuracy 99.71% and reconstructs attack scenarios with accuracy 87.4%.

1 INTRODUCTION

As artificial intelligence reshapes the cybersecurity landscape, organizations globally are encountering a growing array of security and privacy challenges. The extensive deployment of Internet of Things (IoT) devices in critical infrastructure has significantly benefited sectors such as smart cities and industrial automation. Unfortunately, the inherent security vulnerabilities of these devices have made them prime targets for cyberattacks, and the number and type of cyberattacks on the IoT rapidly increase. [Hawawreh et al., 2021] According to the 2023 report by Palo Alto Networks [Networks, 2023], 75% of IoT devices have critical vulnerabilities, with each device experiencing an average of 5,200 attack attempts per week. According to the ENISA report in 2024 [ENISA, 2024], 76.29% enterprise networks were targeted by cybercriminals, and 40% supply chain attacks involving IoT devices.

Advanced Persistent Threat (APT) attacks are increasingly characterized by long-term stealth, cross-domain penetration, and multi-target outbreaks, posing significant risks to network infrastructure. APT groups employ sophisticated intrusion kill chains, coordinated attack campaigns, and bespoke tactics, techniques, and procedures (TTPs) [Sun et al., 2023] to compromise networks and exfiltrate sensitive information assets. The MITRE ATT&CK indicates that lateral movement and persistence are critical stages in APT attacks [Xiong et al., 2022b], enabling attackers to exploit vulnerabilities to traverse and penetrate continuously, such as APT28 and APT29 utilizing zero-day exploits [Garg et al., 2022]. Additionally, they expand the influence through supply chain compromise [Syed et al., 2022] or third-party-associated attacks [Ikram et al., 2019], such as SolarWinds [Hassija et al., 2020] and APT41's infiltration of the global manufacturing industry [Mahmoud et al., 2023]. Therefore, there is a critical need to adopt a proactive approach to investigate attacks and uncover latent threats.

To identify potential security risks and enhance APT investigations capabilities, collaboration between different organizations and departments is essential. However, it is challenging to pool raw audit logs from multiple parties due to the privacy policies. Federated Learning (FL) is an emerging decentralized collaborative paradigm initially proposed by McMahan et al., aiming to address the challenges of data silos and privacy preservation [McMahan et al., 2017]. Collaborative analysis of multi-source threat intelligence based on the FL framework can not only reconstruct a complete APT attack chain without privacy disclosure, but also provide clear technical anchor points for the traceability of APT attack.

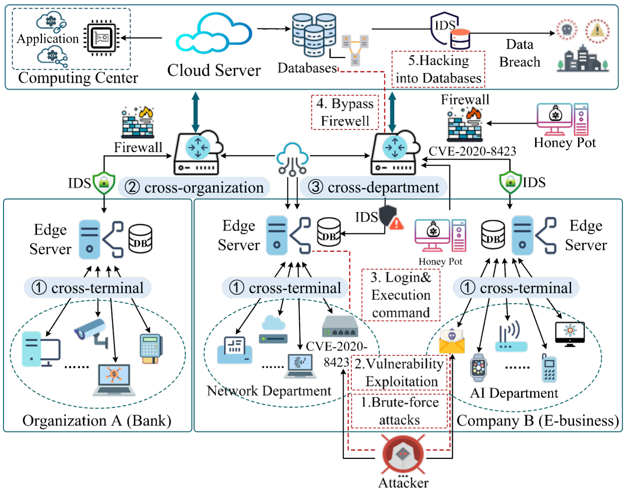


Figure 1: A typical IoT attack scenario. The vulnerability IDs and attack steps are described in red dashed boxes. Attackers gain unauthorized access and leverage the compromised IoT device as a foothold to lateral movement. Then they bypass the firewalls and IDS by exploiting the vulnerabilities to infiltrate the server.

A sophisticated attack example for compromising IoT is shown in Figure 1, which includes several types of cross-domain attack scenarios: cross-terminal, cross-department, and cross-organization. Multi-source intelligence collaboration among different security domains enhances security detection and attack investigation. The prior APT attack investigation methods have these limitations: limited cross-domain analysis, insufficient awareness of potential threats, and privacy constraints. Data privacy concerns significantly restrict threat intelligence sharing, hindering the effectiveness of security operations [Nguyen et al., 2021]. The inaccuracies in the causal relationships between provenance graphs from different domains can result in fragmented attack scenarios. Furthermore, audit logs collected by terminal devices are frequently limited and homogeneous, leading to a scarcity of well-annotated attack samples and reduced accuracy in identifying unknown behavior.

To address the limitations, we propose Federated CAusal Provenance Learning for CrOss-DOmaiN Attacks (FALCON), which aims to explore the cross-domain threats with limited data samples and reconstruct fully attack scenarios. The main contributions can be summarized as follows:

- We introduced FALCON, a system designed to enable efficient collaborative intelligence analysis while ensuring privacy security. Mining fine-grained causal relationships from multi-source logs with provenance graphs to trace threat behavior.
- We propose a heuristic method with few-shot learning. The local model exploits pre-training tasks to learn more accurate semantical information and optional downstream task training modules to accommodate both labeled and unlabeled samples.
- The experimental results show that FALCON significantly outperforms existing methods, achieving higher AUC values of 0.9625 and 0.9497 for the authenticity of alarm events and system event discovery. The system exhibits robust generalization on public datasets.

2 RELATED WORKS

2.1 APT ATTACK INVESTIGATION

In recent years, the concept of using causality analysis through provenance graphs from audit logs has gained widespread application in attack detection [Zengy et al., 2022, Cheng et al., 2023], attack investigation [Alsaheel et al., 2021, Ding et al., 2023], and attack scenario reconstruction. Holmes [Milajerdi et al., 2019] and RapSheet [Hassan et al., 2020a] use TTPs rules to match within provenance graphs, aiming to discover threat behaviors at both the technical and strategic levels. The coarse-grained nature of audit logs introduces the challenge of dependency explosion. MORSE [Hossain et al., 2020] addresses this challenge by introducing tags decay and tag propagation rules.

Utilize graph summarization to encapsulate the semantics of behaviors, enabling efficient and accurate attack investigation. DEPcoMM [Xu et al., 2022] identifies clusters as process-centric communities within large-scale provenance graphs. OmegaLog [Hassan et al., 2020b] generates more concise attack provenance graphs with rich semantic information. The log records a vast amount of system events may affect the accuracy of attack path tracing. DEPIMPACT [Fang et al., 2022] assigns distinguishable dependency weights to edges to differentiate critical. WATSON [Zeng et al., 2021] combines event semantics as representations of behaviors and reduces analysis workload by two orders of magnitude.

Deep learning can build attack investigation models by learning normal and attack behavior features. ATLAS [Alsaheel et al., 2021] constructs sequences including rich semantic information at the system level and training with Long Short-Term Memory (LSTM) [Memory, 2010]. However, LSTM training is time-consuming and requires a large amount of high-quality labeled data. Log2Vec [Liu et al., 2019] constructs heterogeneous graphs with predefined rules and employs clustering to separate malicious behavior from benign behavior without leveraging GNN. AIRTAG [Ding et al., 2023] directly performs representation learning (RL) on audit logs with LSTM. This method relies solely on BERT, limiting its ability to capture rich context.

2.2 FL FOR SECURITY

IoT devices are vulnerable to cyberattacks owing to dispersed locations, limited computational resources, and handling of sensitive data. Wang et al. [Wang et al., 2023] proposed a lightweight FL framework for real-time anomaly detection on resource-constrained IoT devices. Existing security technologies leveraging FL predominantly focus on traffic analysis [Rodríguez-Barroso et al., 2023, Salim et al., 2024] rather than log analysis [Nguyen et al., 2019, Wang et al., 2024] proposed an autonomous self-learning distributed system for detecting anomalies in IoT devices. DeepFeed [Li et al., 2021, Tan et al., 2022b] applies federated deep learning to detect cyber threats against industrial cyber-physical systems.

Federated learning enables collaborative analysis of multi-source threat intelligence without sharing original log data. Some researches developed multimodal FL model [Bahadoripour et al., 2024] that integrates logs, traffic, and sensor data to improve the accuracy of APT attack detection. Mimura et al. [Hu et al., 2023] proposed a privacy-preserving few-shot traffic detection method, treating the APT detection task as a model generalization optimization process to identify unknown local samples. Saeed et al. [Saeed et al., 2020] proposed a self-supervised method based on wavelet transform to learn models from scattered data, which performed well in both centralized and federated environments. Xiong et al. [Xiong et al., 2022a] introduced a practical Real-Time design for detecting known and unknown APT attacks in real-world scenarios.

3 PRELIMINARIES

This section commences by providing several formal definitions of the requisite preliminaries and a threat model given in Appendix A, aiming at facilitating a comprehensive understanding of the proposed methodology.

Definition 1: *System Event. System events* are formal representations of audit logs. It is defined as a quadruple $event = \langle sub, obj, oper, T_s \rangle$, where sub and obj denote objects, both of which are system entities. The entity type set sub is $\{Process\}$ and the entity type set of obj is $\{Process, File, Socket\}$. $oper$ denotes the operation from a subject sub to an object obj, which also denotes causal relationships and information flow between entities. T_s represents the timestamp of the system event.

Definition 2: *Provenance Graph*. A *provenance graph* is generated from the system events by linking the entities with causal relationships, representing the behavior processes and information flows in the operation system level. Provenance graphs are labeled directed graphs, which are formalized as $G_p = (V_{entity}, E_{oper})$, where V_{entity} is the set of entity nodes with attributes and E_{oper} is the set of directed edges with labels. In a provenance graph, multiple edges may exist between two entities, which represent the behavior of operations at different times.

Definition 3: *Behavior Sequence*. The *behavior sequence* is introduced in this work to describe the interaction process of behavior instances in the system level. A behavior sequence indicates that a temporally-ordered chain of system events, represented as $Seq_B^l = \{event_1, event_2, ..., event_l\}$. Behavior sequences contain extensive and distant contextual information of system events. Using this contextual information allows sequence learning models to learn features and patterns of behavior sequences, leading to accurate classification.

Definition 4: *Attack Scenario*. An *attack scenario* provides a comprehensive representation of the entire attack process, encompassing entities relevant to the attack and the causal relationships between these entities. It is represented as $G_{as} = (V_{att\_entity}, E_{att\_oper})$. Compared to the long manual examination to analyze the access points and the potential impact of an attack from audit logs, attack scenarios enable security analysts to more intuitively understand the complete process of an attack.

4 MODEL ARCHITECTURE

This section introduces the architecture of FALCON, and the overall workflow in IoT systems is illustrated in Figure 2. The model is based on the following assumptions: a typical horizontal FL framework with personalized optimization with private datasets and share the same feature space.

4.1 FL STRUCTURE FOR FALCON

We leverage a typical horizontal FL paradigm to enable collaborative attack investigation where have private audit logs as local datasets but share the same feature space. As shown in Figure 2, the overall framework comprises local client module and server module. Through the utilization of cross-terminal causal traceability and localized pre-training optimization, the model's capability of APT attacks investigation has been substantially improved. We further examined the methodology for local model updates, wherein clients extract behavior sequences to capture causal relationships. Inspired by personalized FL[Tan et al., 2022a], we aim to learn shared data representations across clients while establishing a unique local output layer for each client. To ensure the consistency of the optimization objective, a proximal term was incorporated into the client loss function. The proximal term penalizes large deviations from the global

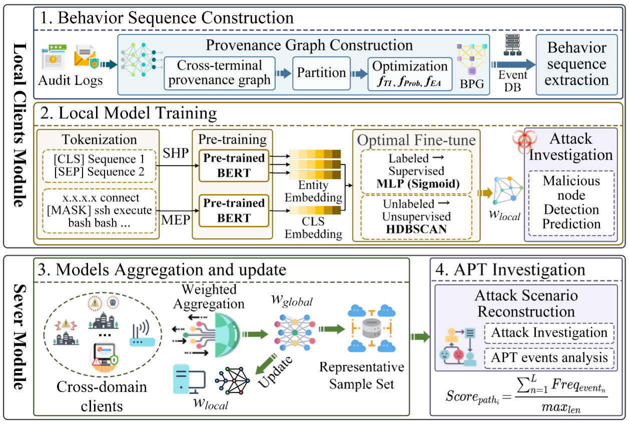

Local Clients Module
1. Behavior Sequence Construction
Audit Logs
Provenance Graph Construction
Cross-terminal provenance graph
Partition
Optimization f_{TI}, f_{Prob}, f_{EA}
BPG
Event DB
Behavior sequence extraction
2. Local Model Training
Tokenization
[CLS] Sequence 1 [SEP] Sequence 2
x.x.x.x connect [MASK] ssh execute bash bash ...
SHP
MEP
Pre-training
Pre-trained BERT
Pre-trained BERT
Entity Embedding
CLS Embedding
Optimal Fine-tune
Labeled → Supervised MLP (Sigmoid)
Unlabeled → Unsupervised HDBSCAN
w_{local}
Attack Investigation
Malicious node Detection Prediction
Sever Module
3. Models Aggregation and update
Cross-domain clients
Weighted Aggregation
w_{global}
Update
w_{local}
Representative Sample Set
4. APT Investigation
Attack Scenario Reconstruction
Attack Investigation
APT events analysis
$Score_{path_i} = \frac{\sum_{n=1}^{L} Freq_{event_n}}{max_{len}}$

Figure 2: Overall architecture of FALCON. The local client module trains local models to detect APT attacks. The processes are provenance graph partitioning and optimization, behavior sequence construction, and optimal fine-tuning training. The server module aggregates and updates models and facilitates cross-domain APT attack investigation.

model, thereby stabilizing training with heterogeneous logs. Client i approximately minimize the following objective, safely incorporating variable amounts of local work.

$$\min_{w} h_i(w; w^t) = F_i(w) + \frac{\mu}{2} \left\| w - w^t \right\|^2, \qquad (1)$$

where $F_i(\cdot)$ is the local function. The central server collects the local model updates w_i from client $i \in S_t$ randomly chosen, then aggregates N clients with weighted averaging, which is calculated as follows:

$$w^{t+1} = \frac{1}{N} \sum_{i \in S_t} w_i^{t+1}, \qquad (2)$$

The updated global model is then broadcast back to all clients, which incorporate these improvements into their subsequent local training cycles. The local model receives the aggregated global model update parameters from the central server at the start of each new training round, ensuring that they benefit from the latest global insights. This iterative process accommodates variable local workloads and is further enhanced by adaptive hyperparameter tuning. The inference flow is as follows: audit log → provenance graph → behavior sequence → semantic vector → detection & investigation. During inference, when a new alert is generated, the corresponding behavior sequence is processed to produce an embedding that feeds into a classifier (or an unsupervised technique). This classifier determines whether the sequence is indicative of an ongoing APT attack. To achieve cross-domain attack scenario reconstruction, critical information regarding malicious behaviors is shared concurrently with the enhancement of the client's model-checking capability. Therefore, FALCON scales across distributed environments, improving detection accuracy for APT investigations and ensuring robust convergence under conditions of heterogeneous system capabilities.

4.2 BEHAVIOR SEQUENCES CONSTRUCTION

Provenance graph construction. Aiming at the problem of missing correlations in cross-terminal provenance graphs, FALCON designed a cross-terminal entity correlation method based on event occurrence time and information flow. We conduct information flow analysis based on aligned system events to determine the real relationship. Figure 3 depicts the constructed cross-terminal provenance graph. FALCON identifies system events with causal relationships from the provenance graph of terminals in established communication. The construction of the cross-terminal provenance graph is realized by removing directed edges and socket nodes in two system events and constructing a correlation relationship between processes in them.

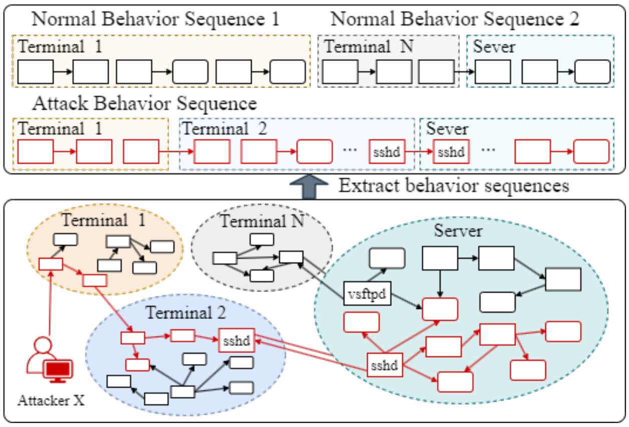


Figure 3: Cross-domain provenance graph partitioning and reduction: This process involves reducing provenance graphs while preserving essential relationships and information. The lateral movement process is marked in red.

Provenance graph partition and optimization. In response to the coarse-grained redundancy of audit logs, FALCON partitions the constructed original provenance graph utilizing the similarity and removes redundant events and calls. To calculate the similarity between system events, FALCON extracts three features as: time interval feature f_{TI}, probability feature f_{Prob}, and entity attributes feature f_{EA}. (see Appendix B.2) The probability that two system events belong to the same behavior is calculated as follows:

$$P_{sim}(event) = \lambda \cdot f_{TI} + \mu \cdot f_{Prob} + \delta \cdot f_{EA}, \qquad (3)$$

where the coefficients λ, μ, and δ represent weighting factors that can be adjusted based on the characteristics to balance the contributions of the features. Figure 3 illustrates the provenance partition and optimization process with examples. FALCON uses HDBSCAN [McInnes and Healy, 2017] to implement our clustering task, which can receive a matrix and does not need to declare the number of clusters in advance. Through the segmentation of extended-duration processes and the consolidation of redundant events, the initial provenance graphs are converted into behavior-focused provenance graphs that concisely capture system activities.

Behavior sequence extraction. FALCON designs a Depth-First Search (DFS) method with specific conditions to extract behavior sequences for each system event from Behavior Partition Graphs (BPGs). Since obtaining labeled data is not feasible in real IoT environments, the traversal paths are determined based on the frequency of event occurrences. This ensures obtaining distant contextual relationships without excessively long behavior sequences. The detailed definitions are provided in Appendix B.

4.3 LOCAL MODEL TRAINING

Tokenization of behavior sequence. Tokenization is splitting events in a sequence into smaller units (tokens) and using these tokens to represent a sequence of behaviors. Therefore, FALCON constructs a token dictionary from the words in the behavior sequences, $Dict_{Seq_B}$, including entities and operations. Similarly to the BERT model, several special tokens [CLS], [SEP], [PAD], and [MASK] are added to the dictionary. The embedding representation of a tokenized sequence is constructed by integrating token embeddings, positional embeddings, and segment embeddings.

Pre-training. FALCON achieves representation learning on a large set of unlabeled behavior sequences by designing pre-training tasks and maps words and sequences into a vector space. In this vector space, semantically similar words and sentences have closer distances. FALCON designs two pre-training tasks along with corresponding loss functions for representation learning at the word and sequence levels. We present details in Appendix B.4.

(1) Masked Entity Prediction (MEP) task. FALCON employs a multi-layer bidirectional Transformer architecture for training. This model architecture leverages the MEP task to capture bidirectional contextual information for embedding a specific word. Here, FALCON employs the negative log-likelihood function as the loss function.

$$\mathcal{L}_{MEP} = -\textstyle\sum_{i=1}^{M} log(p(Seq_i^{mask} = tok_i|\theta, \theta_1)) \tag{4}$$

Where M is the number of masked entities, θ is the parameters of the Transformer Encoder, θ_1 is the parameter of the output layer connected to the Encoder in the Masked Entity task. Probability function $p(\cdot)$ depends on the parameters θ and θ_1, Seq_i^{mask} represents a token masked at the $i-th$ position in the tokenized behavior sequence.

(2) Sequence Homology Prediction (SHP) task. The goal of the SHP task based on sequence-level representation learning is to predict whether two behavior sequences originate from the same BPG. Behavior sequences from the same origin graph may exhibit potential causal relationships. SHP is a binary classification, so FALCON employs binary cross-entropy loss function for training.

$$\begin{gathered}\mathcal{L}_{SHP} = -\textstyle\sum_{i=1}^{N} log(p_i(n = n_i|\theta, \theta_2)), \\ n_i \in \{Homologous, NonHomologous\},\end{gathered} \tag{5}$$

where N is the number of input sequence samples. θ_2 is the parameter of the output layer connected to the Encoder in the SHP task. p_i denotes the predicted value of the model for $i-th$ sample. To capture both token-level and sequence-level features, FALCON utilizes a combined loss of MEP and SHP objectives as the overall pre-training loss.

$$\mathcal{L}_{overall} = \mathcal{L}_{MEP} + \mathcal{L}_{SHP}. \tag{6}$$

Fine-tune for the downstream task. FALCON has designed optional fine-tuning modules for the downstream task. In scenario where obtaining high-quality labeled data is challenging, FALCON employs an unsupervised classification to train on unlabeled datasets. We employ One-Class Support Vector Machine (OC-SVM) for unsupervised classification training. When high-quality labeled data is available from the TDS in the IoT system. FALCON utilizes this labeled data to simultaneously learn from both attack and normal behavior sequences, achieving fine-tuning of the model. (More in Appendix B.5)

4.4 ATTACK INVESTIGATION ANALYSIS

The goal of FALCON is to automatically investigate the authenticity of TDS alarms in IoT systems, identify undetected attack events, and construct complete attack scenarios. An attack case study was conducted to demonstrate the effectiveness of FALCON in IoT attack investigation (see Appendix C.6).Each client constructs an independent APT attack scenario based on local system events associated with known malicious behaviors. Moreover, clients upload critical information of malicious behaviors, allowing us to reconstruct a global APT attack scenario across domains on the server. Since only model updates and malicious behavior information are shared, FALCON essentially mitigates the potential privacy risks associated with sharing raw audit logs. Each client constructs independent APT attack scenario based on local system events associated with known malicious behaviors. Moreover, clients upload critical information of malicious behaviors, allowing us to reconstruct a global APT attack scenario across domains on the server. Since only model updates and malicious behavior information are shared, FALCON essentially mitigates the potential privacy risks associated with sharing raw audit logs.

FALCON chooses low-frequency events as traversal paths to construct behavior sequences with system events corresponding to alarms as root nodes. FALCON employs the trained model to determine whether the behavior sequences associated with alerts are malicious. Attacks that evade TDS and remain lurk can cause more serious harm to IoT systems. For potential undetected attack events, FALCON analyzes all system events in the IoT system. Specifically, FALCON constructs BHGs and extracts behavior sequences. The extracted behavior sequences are predicted using the trained model, and the system events corresponding to the predicted malicious sequence are flagged as malicious.

Reconstructing the attack scenario involves establishing associations between system events corresponding to real alerts and malicious events, and generating attack scenario graphs to depict the entire attack process. Specifically, FALCON constructs the attack graph by considering the reachability of malicious events belonging to the same behavior origin graph. When searching for reachable paths in the behavior origin graph, multiple paths may exist. To minimize the cost of attack implementation, attackers typically choose the shortest path to execute the attack. Therefore, FALCON calculates scores for all paths based on the frequency of events occurring on the path and the path length.

$$\begin{aligned} Score_{path_i} =& \frac{1}{len_{path_i}} \sum_{n=1}^{L} Freq_{event_n} \times \frac{len_{path_i}}{max_{len}}, \\ =& \frac{\sum_{n=1}^{L} Freq_{event_n}}{max_{len}}, \end{aligned} \tag{7}$$

Where $path_i \in \mathcal{PATH} = \{path_1, \ldots, path_m\}$ is one of all reachable paths between two malicious system events. $max_{len} = MAX\{len(\mathcal{PATH})\}$ represents the length of the longest path among all accessible paths. FALCON selecting the path with the minimum score to connect two malicious events, which represents the most likely path chosen by the attacker in the combined lowest frequency and shortest path case. In the end, FALCON can reconstruct a comprehensive attack scenario spanning multiple hosts. The resulting scenario graph is concise and contains crucial information about the attack.

5 EXPERIMENT

In this section, we evaluate FALCON from multiple dimensions of experiments and present the main results. We also perform ablation studies and explainability analyses.

5.1 EXPERIMENT SETUP

The IoT system comprises 20 IoT devices, 6 edge servers, and one cloud service, with each edge server connected to at least two IoT devices. We perform the experiments to analyze audit logs on each server docker with an Intel(R) Xeon(R) Silver 4215R CPU (with 8 cores and 3.20 GHz of speed each), a GeForce RTX 3090, and 256 GB of memory running on Ubuntu 18.04.5 LTS. Set IoT devices as data collection terminals and edge servers as clients for local model training, while cloud servers perform global model aggregation to facilitate threat intelligence sharing. For a distributed FL structure, we deploy a central server, six edge servers, and a high-performance host as isolated client devices. Each client manages 2 or 3 IoT terminals and performs model training within Docker containers. We constrain the number of communication rounds between 200 and 500, dynamically adjusting the local iteration counts based on requirements to enhance the convergence performance of the global model. Simulating five APT attacks based on detailed reports [Hanh, 2022], each complete attack lasted at least 2 days. More details of the implementation are listed in the Appendix C.1.

Datasets. Based on detailed reports of real-world APT campaigns, we conducted five simulated attacks and generated audit logs within a controlled IoT testbed environment, presented as the IoT dataset. Excepting the classical attack scenarios OceanLotus [Hanh, 2022], APT28 [Security, 2018], and Kimsuky [CYBER, 2020], we also design two sophisticated attacks exploiting some new vulnerabilities. Additionally, two widely used datasets are used to evaluate the generalizability of FALCON, the ATLAS dataset [Alsaheel et al., 2021] and the CADETS dataset [Torrey, 2020]. The attribute information is listed in Appendix C.2. We introduced two mixed datasets: one combining IoT and ATLAS datasets and another combining all three datasets. It is important to note that mixing occurs before the samples enter the model, not on the original audit data.

Evaluation Setup. In the experimental environment, we simulated some unauthorized actions by normal users, such as elevating process privileges (triggering an alarm) and then running a program to read and write multiple files. This series of actions exhibited behavior patterns similar to those of attackers running malicious software. *Precision*, *Recall*, *F1-score*(see in the appendix C.3), as well as common evaluation metrics such as *ROC* curve and *AUC*, are used to evaluate the performance of FALCON in attack investigation.To quantitatively evaluate the reconstruction results, we design three metrics: *SNE*, *DNE*, and *SIM*.

$$\mathit{SNE} = \frac{|SN| + |SE|}{|N_{GT}| + |E_{GT}|}, \tag{8}$$

$$\mathit{DNE} = \frac{|DN| + |DE|}{|N_{GT}| + |E_{GT}|}, \tag{9}$$

$$\mathit{SIM} = \frac{|SN| + |SE|}{max\{(|N_{GT}|+|E_{GT}|), (|N_R|+|E_R|)\}}, \tag{10}$$

where SN and SE represent the same nodes and edges, DN and DE represent different nodes and edges. N_R and N_{GT} denote nodes in the reconstructed graph and ground truth graph, respectively, E_R and E_G represent edges. A sig-

Table 1: APT Investigation: The results of FALCON in investigating the authenticity of alerts and identifying lurking attack events within system events, along with the ground truth information for each executed attack.

Scenarios	Ground Truth			Alerts Investigation Result			Events Investigation Result		
	Events	True	False	*Precision*	*Recall*	*F1-score*	*Precision*	*Recall*	*F1-score*
OceanLotus	57	7	48	100.00%	97.92%	98.95%	95.00%	100.00%	97.44%
APT28	68	8	39	97.50%	100.00%	98.73%	95.65%	97.06%	96.35%
Kimsuky	44	6	52	100.00%	96.15%	98.04%	97.67%	95.45%	96.55%
attack 1	38	0	31	100.00%	96.77%	98.36%	97.30%	94.74%	96.00%
attack 2	31	4	23	95.65%	95.65%	95.65%	96.77%	96.77%	96.77%
Total or Avg.	238	25	193	98.95%	97.41%	98.17%	96.25%	97.06%	96.65%

nificant proportion of the graph consists of the same nodes and edges (*SNE*), indicating that FALCON's reconstructed attack scenarios include most of the critical attack events. Different nodes and edges (*DNE*) represent those attacks appear in the scene graph but not in the attack graphs.

5.2 APT ATTACK INVESTIGATION EVALUATION

Threat Detection Systems (TDS) can only detect a limited number of attack events and tend to generate a significant number of false alerts, which can be observed from the column "Ground Truth" in Table 1. There are 238 system events directly related to the attacks, as well as numerous system calls generated alongside these attack events. Within two weeks, IDS and the firewall generated a total of 218 alerts, with 25 true alerts and 193 false alerts. The reason behind this lies in the fact that attackers often disguise their behavior to evade detection.

The "Alerts Investigation Result" presented in columns 5 to 7 of Table 1 indicates that FALCON can accurately determine the authenticity of alerts, achieving an average *precision* of 98.95%, a *recall* rate of 97.41%, and an *F1-score* of 98.17%. The fifth column shows that among all predicted true alarms, 98.95% were triggered by attack events. Upon analysis, false positives were caused by some normal behaviors that resemble behavior patterns of attacks. The sixth column indicates an average recall rate of 97.41%. Errors originated from some failed initial intrusion attempts.

The "Events Investigation Result" in columns 8 to 10 of Table 1 shows that FALCON can recognize benign events with an average precision of 96.25%, a recall rate of 97.06%, and an F1-score of 96.65%. The results indicate that FALCON's performance in identifying attack events is slightly better than benign events. The occurrence of false negatives in the event investigation is also attributed to failed access attempts during the initial access phase. (More in the appendix C.4.)

The ROC curve and AUC value for FALCON during alerts investigation are illustrated in Figure 4. It demonstrates that FALCON is capable of accurately determining the authenticity of alerts. AuditLogBERT can accurately identify system events corresponding to attacks with a *precision* of 95.87%, *recall* rate of 97.48%, and *F1-score* of 96.67%. The figure displays the ROC curve for FALCON during event investigation, with an AUC value of AUC=0.9497. The above results show that AuditLogBERT can effectively make up for the shortcomings of TDS and detect the key attack steps missed. The above results indicate that FALCON can effectively judge the authenticity of TDS alerts and detect critical attack steps overlooked by TDS.

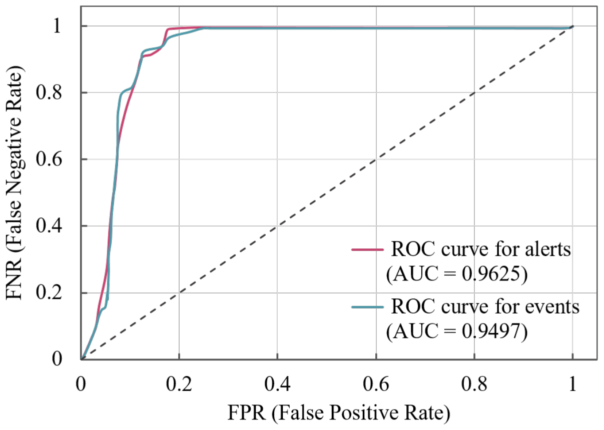


Figure 4: ROC curve and AUC of FALCON in attack investigation

5.3 APT ATTACK SCENARIO RECONSTRUCTION

To evaluate the effectiveness of FALCON in reconstructing attack scenarios, we compared the reconstructed attack scenario graphs with the ground truth of attack graphs. Figure 5 shows the number of nodes and edges that are same or different between them. The results show that the number of nodes and edges in the reconstructed attack scenario graphs are generally higher than the attack graphs. These differences are mainly benign events misclassified as attack events and events included in the incorrectly chosen paths during the reconstruction of the attack scenario. Some attack events not identified by FALCON are reconstructed in the attack scenario through path selection.

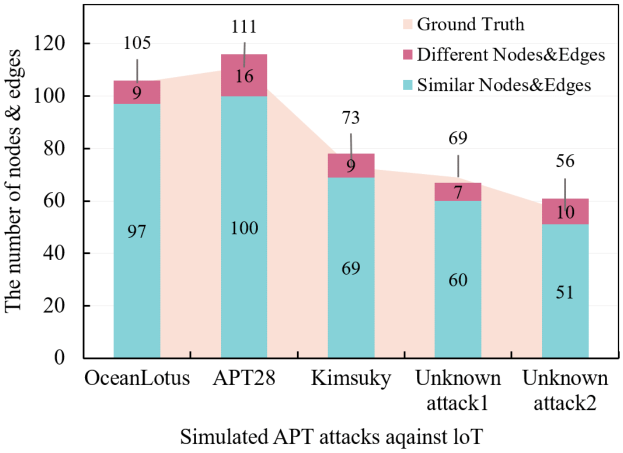


Figure 5: Comparing the identical and different nodes and edges between the reconstructed attack and the ground truth scenarios.

Table 2: Results of attack scenario reconstruction, in terms of ground truth and three metrics, *SNE*, *DNE*, and *SIM*.

Scenarios	**G-Truth**		**Rconstruction results**		
	$\|N\|$	$\|E\|$	*SNE*	*DNE*	*SIM*
OceanLotus	48	57	92.4%	7.5%	91.5%
APT28	43	68	90.1%	14.4%	86.2%
Kimsuky	29	44	94.5%	12.3%	88.5%
Attack4	31	38	87.0%	10.1%	87.0%
Attack5	25	31	91.1%	17.9%	83.6%
Total or Avg.	176	238	91.1%	12.3%	87.4%

The results of attack scenario reconstruction are shown in table 2. FALCON reconstructed attack scenarios that, on average, included 91.1% of the attack edges and nodes, with a similarity ranging from 83.6% to 91.5%. The experimental results indicate that FALCON can accurately reconstruct concise APT attack scenario graphs from multi-source and heterogeneous audit logs. It removes system events that are not directly related to the attack, retaining only key attack system events. These simplified attack scenario graphs can assist security analysts in quickly understanding the complete attack process during attack investigations, identifying the attack entry points, and assessing the impact.

5.4 ABLATION EXPERIMENT

The ablation experiments are conducted on multiple datasets to evaluate the factors influencing FALCON's performance. The proposed behavior provenance graph, pre-training model, and downstream task classifiers are validated to enhance FALCON's capability in conducting APT attack investigations. The comparison of downstream task classifiers and more details are demonstrated in Appendix C.5.

Table 3: Performance of attack investigations using graph partition and optimization algorithms and raw graphs.

Datasets	**Graphs**	**Time**	**Events Investigation Result**		
		(h:m:s)	*Precision*	*Recall*	*F1-score*
IoT Dataset	G_{Raw}	3:47:49	56.56%	81.51%	66.78%
	G_{Opt}	1:14:09	96.25%	97.06%	96.65%
ATLAS [Alsaheel et al., 2021]	G_{Raw}	2:39:06	68.32%	75.89%	71.91%
	G_{Opt}	0:58:23	97.39%	98.60%	97.99%
CADETS [Torrey, 2020]	G_{Raw}	5:29:54	74.76%	76.95%	75.84%
	G_{Opt}	1:35:47	96.89%	98.92%	97.89%

Raw graph vs. optimized graph. The removal of redundant events significantly improves the efficiency of attack investigations, while eliminating errors in dependencies enhances the accuracy of identifying attack events. The results in Table 3 demonstrate that the runtime for attack investigations significantly decreases when using the optimized BHGs compared to the raw graphs on the three datasets. The performance of attack investigation using the raw graphs is the poorest in the IoT dataset, with a significant decrease in the F1 score by 29.87%. This decline can be attributed to the heterogeneity of the data and semantic differences, which introduce considerable noise, making it challenging for the model to effectively learn patterns within the behavior sequences. And all metrics of the event investigation results are higher than those using the raw graphs. Therefore, the proposed approach of obtaining origin provenance graphs through partitioning and optimization enhances the efficiency and performance of attack investigations.

Pre-training model. To evaluate the effectiveness of the pre-trained model proposed by FALCON in embedding behavior sequences more efficiently, we compared it on three datasets with four typical deep learning models, including CNN, LSTM, and two state-of-the-art pre-trained models for sequence analysis and natural language processing, BERT and RoBERTa.

The comparative results between FALCON and different deep learning models are shown in Figure 6. FALCON achieved the best results in all three datasets, with F1 scores reaching 96.65%, 97.99%, and 97.89% on the IoT dataset, ATLAS, and CADETS, respectively. Compared to FALCON, BERT and RoBERTa showed a decrease in *F1-scores* ranging from 3.92% to 8.34% across the three datasets. The improvement of FALCON compared with BERT indicates that the pre-training task proposed in this paper can effectively promote the downstream task of attack investigation. In the IoT dataset, the largest difference between *F1-score* scores for BERT and FALCON indicates a negative impact of the Next Sentence Prediction (NSP) task on attack investigation tasks in the IoT context. The slight improvement of RoBERTa over BERT also indicates that the NSP task is not

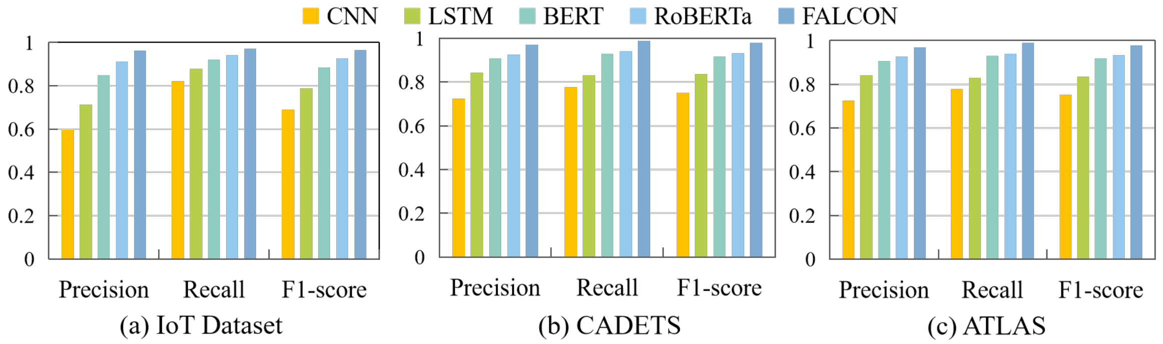


Figure 6: To evaluate the effectiveness of the pre-trained model proposed by FALCON in embedding behavior sequences compared with several deep learning models.

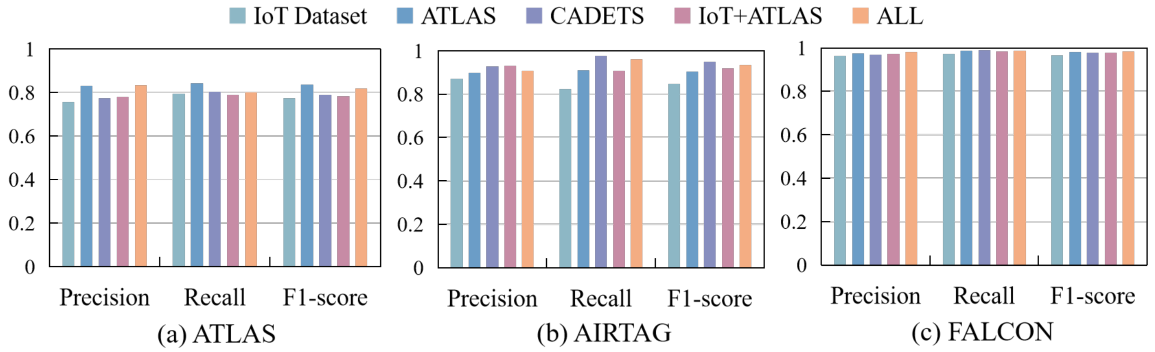


Figure 7: Comparative results with existing APT advanced attack investigation methods, in terms of ATLAS and AIRTAG.

suitable for attack investigations.

5.5 COMPARISON EXPERIMENTS

We performed a comparative analysis between FALCON and existing advanced attack investigation methods to assess the strengths and weaknesses of different approaches. The comparative results are presented in Figure 7. ATLAS and AIRTAG achieved a maximum F1 of only 84.67% on the IoT dataset, while FALCON's *F1-score* improved by 11.98%. The results indicate that existing attack investigation methods cannot be directly applied to IoT systems. The poor performance of AIRTAG is attributed to its proposed tokenization strategy, which does not adequately cover semantically rich and diverse IoT audit logs. The primary reason for ATLAS's lowest performance is not only semantic differences but also the scarcity of training samples provided from the IoT dataset.

In the mixed datasets, FALCON maintains strong performance with *F1-score* values of 97.78% and 98.33%. The primary reason for ATLAS's poor performance across multiple datasets remains the insufficient number of training samples. Comparing the *F1-score* values of the three methods on the mixed dataset with their respective values on the IoT dataset, we observe a slight improvement in the effectiveness of attack investigation in IoT by adding datasets. The increased number of attack behaviors facilitates learning more attack patterns, thereby improving the recognition of attack events.

6 CONCLUSION

We address the challenge of cross-domain APT attack hindrance and the few sample limitations. We propose a novel APT attack investigation method based on FL capturing complex causal relationships, named FALCON. FALCON constructs cross-terminal BPGs from heterogeneous audit logs. FALCON trains adaptive local models with behavior sequences containing extensive and remote contextual information and learns latent representations from unlabeled sequences. The results demonstrate that FALCON is capable of conducting efficient attack investigations in IoT systems and achieves impressive performance. In the future, scaling FALCON to a larger network involves addressing both computational and communication challenges. We will extend its effective APT investigation capabilities to large-scale IoT networks while preserving the critical balance between performance, privacy, and computational efficiency.

Acknowledgements

This work was supported by the National Key Research and Development Program of China under grant No. 2023YFB3107601.

References

Abdulellah Alsaheel, Yuhong Nan, Shiqing Ma, Le Yu, Gregory Walkup, Z Berkay Celik, Xiangyu Zhang, and Dongyan Xu. {ATLAS}: A sequence-based learning approach for attack investigation. In 30th USENIX security symposium (USENIX security 21), pages 3005–3022, 2021.

Sepideh Bahadoripour, Hadis Karimipour, Amir Namavar Jahromi, and Anik Islam. An explainable multi-modal model for advanced cyber-attack detection in industrial control systems. Internet of Things, 25:101092, 2024.

Zijun Cheng, Qiujian Lv, Jinyuan Liang, Yan Wang, Degang Sun, Thomas Pasquier, and Xueyuan Han. Kairos: Practical intrusion detection and investigation using whole-system provenance. In 2024 IEEE Symposium on Security and Privacy (SP), pages 5–5. IEEE Computer Society, 2023.

Kenneth Ward Church. Word2vec. Natural Language Engineering, 23(1):155–162, 2017.

YOROI TINXTA CYBER. The north korean kimsuky apt keeps threatening south korea evolving its ttps. Accessed 5 April 2023. https://blog.yoroi.company/research/the-north-korean-kimsuky-apt-keeps-threatening-south-korea-evolving-its-ttps/, 2020.

Hailun Ding, Juan Zhai, Yuhong Nan, and Shiqing Ma. {AIRTAG}: Towards automated attack investigation by unsupervised learning with log texts. In 32nd USENIX Security Symposium (USENIX Security 23), pages 373–390, 2023.

ENISA. Enisa threat landscape report, 2024. URL `https://www.enisa.europa.eu/publications/enisa-threat-landscape-2024`. ENISA Threat Landscape Annual Report.

Pengcheng Fang, Peng Gao, Changlin Liu, Erman Ayday, Kangkook Jee, Ting Wang, Yanfang Fanny Ye, Zhuotao Liu, and Xusheng Xiao. {Back-Propagating} system dependency impact for attack investigation. In 31st USENIX Security Symposium (USENIX Security 22), pages 2461–2478, 2022.

Peng Gao, Fei Shao, Xiaoyuan Liu, Xusheng Xiao, Zheng Qin, Fengyuan Xu, Prateek Mittal, Sanjeev R Kulkarni, and Dawn Song. Enabling efficient cyber threat hunting with cyber threat intelligence. In 2021 IEEE 37th International Conference on Data Engineering (ICDE), pages 193–204. IEEE, 2021.

Sanjam Garg, Abhishek Jain, Zhengzhong Jin, and Yinuo Zhang. Succinct zero knowledge for floating point computations. Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, page 1203–1216, 2022. doi: 10.1145/3548606.3560653. URL `https://doi.org/10.1145/3548606.3560653`.

Thich Nhat Hanh. Vietnam: Lotus in a sea of fire: A Buddhist proposal for peace. Parallax Press, 2022.

Wajih Ul Hassan, Shengjian Guo, Ding Li, Zhengzhang Chen, Kangkook Jee, Zhichun Li, and Adam Bates. Nodoze: Combatting threat alert fatigue with automated provenance triage. In network and distributed systems security symposium, 2019.

Wajih Ul Hassan, Adam Bates, and Daniel Marino. Tactical provenance analysis for endpoint detection and response systems. In 2020 IEEE Symposium on Security and Privacy (SP), pages 1172–1189. IEEE, 2020a.

Wajih Ul Hassan, Mohammad Ali Noureddine, Pubali Datta, and Adam Bates. Omegalog: High-fidelity attack investigation via transparent multi-layer log analysis. In Network and distributed system security symposium, 2020b.

Vikas Hassija, Vinay Chamola, Vatsal Gupta, Sarthak Jain, and Nadra Guizani. A survey on supply chain security: Application areas, security threats, and solution architectures. IEEE Internet of Things Journal, 8(8):6222–6246, 2020.

Muna Hawawreh, Elena Sitnikova, and Neda Aboutorab. X-iiotid: A connectivity-agnostic and device-agnostic intrusion data set for industrial internet of things. IEEE Internet of Things Journal, 9(5):3962–3977, 2021.

Md Nahid Hossain, Sanaz Sheikhi, and R Sekar. Combating dependence explosion in forensic analysis using alternative tag propagation semantics. In 2020 IEEE Symposium on Security and Privacy (SP), pages 1139–1155. IEEE, 2020.

Yilun Hu, Jun Wu, Gaolei Li, Jianhua Li, and Jinke Cheng. Privacy-preserving few-shot traffic detection against advanced persistent threats via federated meta learning. IEEE Transactions on Network Science and Engineering, 11(3):2549–2560, 2023.

Muhammad Ikram, Rahat Masood, Gareth Tyson, Mohamed Ali Kaafar, Noha Loizon, and Roya Ensafi. The chain of implicit trust: An analysis of the web third-party resources loading. The World Wide Web Conference, page 2851–2857, 2019. doi: 10.1145/

3308558.3313521. URL https://doi.org/10.1145/3308558.3313521.

Beibei Li, Yuhao Wu, Jiarui Song, Rongxing Lu, Tao Li, and Liang Zhao. Deepfed: Federated deep learning for intrusion detection in industrial cyber–physical systems. IEEE Transactions on Industrial Informatics, 17(8):5615–5624, 2021. doi: 10.1109/TII.2020.3023430.

Jiawei Li, Ru Zhang, and Jianyi Liu. Conlbs: An attack investigation approach using contrastive learning with behavior sequence. Sensors, 23(24):9881, 2023.

Sophos Limited. Cve-2022-1040 detail. Accessed 6 May 2023. https://nvd.nist.gov/vuln/detail/CVE-2022-1040, 2022.

Fucheng Liu, Yu Wen, Dongxue Zhang, Xihe Jiang, Xinyu Xing, and Dan Meng. Log2vec: A heterogeneous graph embedding based approach for detecting cyber threats within enterprise. In Proceedings of the 2019 ACM SIGSAC conference on computer and communications security, pages 1777–1794, 2019.

Moustafa Mahmoud, Mohammad Mannan, and Amr Youssef. Apthunter: Detecting advanced persistent threats in early stages. Digital Threats: Research and Practice, 4 (1):1–31, 2023.

Leland McInnes and John Healy. Accelerated hierarchical density based clustering. In 2017 IEEE International Conference on Data Mining Workshops (ICDMW), pages 33–42. IEEE, 2017.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Aarti Singh and Jerry Zhu, editors, Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, volume 54 of Proceedings of Machine Learning Research, pages 1273–1282. PMLR, 20–22 Apr 2017. URL https://proceedings.mlr.press/v54/mcmahan17a.html.

Long Short-Term Memory. Long short-term memory. Neural computation, 9(8):1735–1780, 2010.

Sadegh M Milajerdi, Rigel Gjomemo, Birhanu Eshete, Ramachandran Sekar, and VN Venkatakrishnan. Holmes: real-time apt detection through correlation of suspicious information flows. In 2019 IEEE Symposium on Security and Privacy (SP), pages 1137–1152. IEEE, 2019.

MITRE. Cve-2020-8423 detail. Accessed 6 May 2023. ttps://nvd.nist.gov/vuln/detail/CVE-2020-8423, 2020.

Palo Alto Networks. Iot threat report, 2023. URL https://www.paloaltonetworks.com/resources/research/iot-threat-report-2023. Accessed: 2023-10-01.

Dinh C. Nguyen, Ming Ding, Pubudu N. Pathirana, Aruna Seneviratne, Jun Li, Dusit Niyato, and H. Vincent Poor. Federated learning for industrial internet of things in future industries. IEEE Wireless Communications, 28(6): 192–199, 2021. doi: 10.1109/MWC.001.2100102.

Thien Duc Nguyen, Samuel Marchal, Markus Miettinen, Hossein Fereidooni, N. Asokan, and Ahmad-Reza Sadeghi. DÏot: A federated self-learning anomaly detection system for iot. In 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), pages 756–767, 2019. doi: 10.1109/ICDCS.2019.00080.

Nuria Rodríguez-Barroso, Daniel Jiménez-López, M Victoria Luzón, Francisco Herrera, and Eugenio Martínez-Cámara. Survey on federated learning threats: Concepts, taxonomy on attacks and defences, experimental study and challenges. Information Fusion, 90:148–173, 2023.

Aaqib Saeed, Flora D Salim, Tanir Ozcelebi, and Johan Lukkien. Federated self-supervised learning of multi-sensor representations for embedded intelligence. IEEE Internet of Things Journal, 8(2):1030–1040, 2020.

Mikail Mohammed Salim, Abir El Azzaoui, Xianjun Deng, and Jong Hyuk Park. Fl-ctif: A federated learning based cti framework based on information fusion for secure iiot. Information Fusion, 102:102074, 2024.

Tencent Security. The magic bear (apt28) organizes the latest attack. Accessed 5 March 2023. https://cloud.tencent.com/developer/article/1042927, 2018.

Nan Sun, Ming Ding, Jiaojiao Jiang, Weikang Xu, Xiaoxing Mo, Yonghang Tai, and Jun Zhang. Cyber threat intelligence mining for proactive cybersecurity defense: a survey and new perspectives. IEEE Communications Surveys & Tutorials, 25(3):1748–1774, 2023.

Naeem Firdous Syed, Syed W Shah, Rolando Trujillo-Rasua, and Robin Doss. Traceability in supply chains: A cyber security analysis. Computers & Security, 112: 102536, 2022.

Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. Towards personalized federated learning. IEEE transactions on neural networks and learning systems, 34(12):9587–9603, 2022a.

Yue Tan, Guodong Long, Jie Ma, Lu Liu, Tianyi Zhou, and Jing Jiang. Federated learning from pre-trained models: A contrastive learning approach. Advances in neural information processing systems, 35:19332–19344, 2022b.

J. Torrey. Transparent computing engagement 3 data release. Accessed 16 January 2023. https://github.com/darpa-i2o/Transparent-Computing/blob/master/README-E3.md, 2020.

Yiming Wang, Hao Zhang, and Wei Liu. Edgeguard: Lightweight federated anomaly detection for iot devices. In 2023 IEEE Symposium on Security and Privacy (S&P), pages 145–162. IEEE, 2023.

Zhihai Wang, Jie Wang, Dongsheng Zuo, Ji Yunjie, Xilin Xia, Yuzhe Ma, HAO Jianye, Mingxuan Yuan, Yongdong Zhang, and Feng Wu. A hierarchical adaptive multi-task reinforcement learning framework for multiplier circuit design. In Forty-first International Conference on Machine Learning, 2024.

Chunlin Xiong, Tiantian Zhu, Weihao Dong, Linqi Ruan, Runqing Yang, Yueqiang Cheng, Yan Chen, Shuai Cheng, and Xutong Chen. Conan: A practical real-time apt detection system with high accuracy and efficiency. IEEE Transactions on Dependable and Secure Computing, 19 (1):551–565, 2022a. doi: 10.1109/TDSC.2020.2971484.

Wenjun Xiong, Emeline Legrand, Oscar Åberg, and Robert Lagerström. Cyber security threat modeling based on the mitre enterprise att&ck matrix. Softw. Syst. Model., 21(1):157–177, February 2022b. ISSN 1619-1366. doi: 10.1007/s10270-021-00898-7. URL https://doi.org/10.1007/s10270-021-00898-7.

Zhiqiang Xu, Pengcheng Fang, Changlin Liu, Xusheng Xiao, Yu Wen, and Dan Meng. Depcomm: Graph summarization on system audit logs for attack investigation. In 2022 IEEE Symposium on Security and Privacy (SP), pages 540–557. IEEE, 2022.

Jun Zeng, Zheng Leong Chua, Yinfang Chen, Kaihang Ji, Zhenkai Liang, and Jian Mao. Watson: Abstracting behaviors from audit logs via aggregation of contextual semantics. In NDSS, 2021.

Jun Zengy, Xiang Wang, Jiahao Liu, Yinfang Chen, Zhenkai Liang, Tat-Seng Chua, and Zheng Leong Chua. Shadewatcher: Recommendation-guided cyber threat analysis using system audit records. In 2022 IEEE Symposium on Security and Privacy (SP), pages 489–506. IEEE, 2022.

Ye Zhang and Byron Wallace. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. arXiv preprint arXiv:1510.03820, 2015.

FALCON: Adaptive Cross-Domain APT Attack Investigation with Federated Causal Learning (Supplementary Material)

Jialu Tang[1] **Yali Gao**[1,*] **Xiaoyong Li**[1] **Jiawei Li**[1] **Shui Yu**[2] **Binxing Fang**[3]

[1]School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing, China
[2]School of Computer Science, University of Technology Sydney, Ultimo, NSW, Australia
[3]Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou, China
[*]Corresponding author, email <gaoyali@bupt.edu.cn>

A THREAT MODEL

The above attack cases intuitively reveal how APT organizations carefully plan attack strategies to penetrate enterprise network defense lines. Analysis of this APT system events requires correlating heterogeneous log data of different types of terminal devices in the enterprise network to discover the complete process of the attack. However, the provenance graph built from the audit logs only records socket attributes when it comes to communication with other terminals or external IPs, such as $\langle bash\_1931, 10.46.146.2 : 2495, connect, 2023/08/27\rangle$, but does not provide information about the processes involved in establishing communication within the connected host. This causes all remote connection events to be connected to the same socket point. As shown in Figure 8, in the process of remotely transferring Report.pdf in 10.135.22.6 to 10.46.146.2, the causal relationship between $bash\_1931$ and $bash\_2371$ cannot be directly constructed from the audit log. The above situation makes it impossible to determine the accurate causal relationship and information flow in the cross-host process, which seriously restricts the discovery of cross-terminal APT based on the provenance graph. In response to the above problems, this paper studies the construction method of cross-terminal provenance graph.

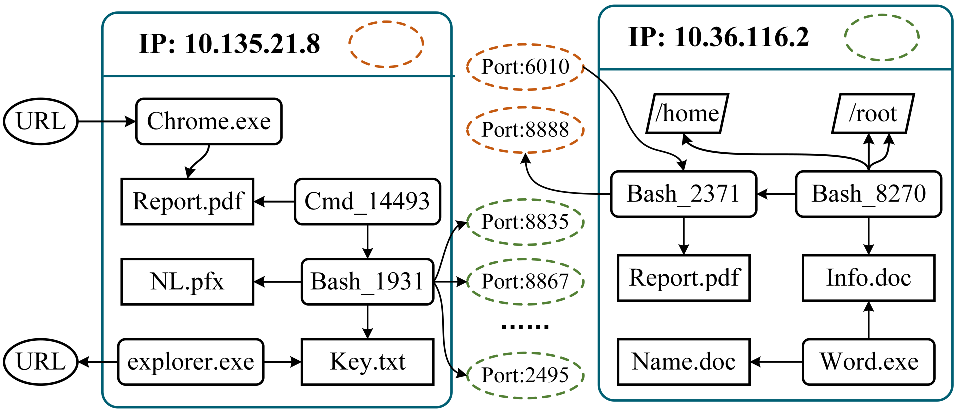


Figure 8: The correlation between the provenance graph of different terminals.

Similar to previous efforts in threat discovery based on audit logs [Hassan et al., 2019, Alsaheel et al., 2021, Fang et al., 2022, Xu et al., 2022, Gao et al., 2021], our approach operates under the prerequisite of ensuring the integrity and authenticity of audit logs. Therefore, our threat model assumes that the underlying operating system, audit engine, and monitoring data are integral components of a Trusted Computing Base (TCB). Our methodology does not account for kernel attacks or attacks targeting audit logs, including activities such as log deletion or modification. Although ensuring the integrity and authenticity of logs is a crucial aspect of a security framework, this specific aspect is beyond the scope of our research in this paper.

B IMPLEMENTATION DETAILS

B.1 PREPROCESSING

FALCON analyzes various types of audit logs with different structures, converts them into system events, and builds original provenance graphs based on these system events. Firstly, The set of system events containing sockets is found from the provenance graph. For each event $e \in E_s$, the extracted attributes include the time of occurrence of the event and the information flow of the event in the terminal, when carrying outgoing information, the information flow $Flow^{in}$ indicates the system event that carries the inflow of information. conversely, when receiving the input of information, the information flow $Flow^{out}$ indicates the system event that carries the output from the output.

Secondly, the time of system events e_i^A and e_j^B in different terminals are aligned. Usually the occurrence time of the system event that establishes a connection between terminals is the same, and the correlation between the process entities in the two events is established through time alignment, and the time alignment formula is as follows. R_{talig} is the set of time-aligned system events.

$$R_{talig} = \{(e_i^A, e_j^B)|T_{aligned} = e_i^A.time,\ e_j^B.time = T_{aligned}\}, \tag{11}$$

Finally, on terminal servers with frequent business, multiple system events often occur at the same time, and these system events may correspond to multiple independent behaviors executed at the same time. Therefore, we conduct information flow analysis based on aligned system events to determine the real relationship.

$$\begin{aligned} R_{relation} = \left(e_i^A, e_j^B\right) \mid \left(e_i^A, e_j^B\right) \in\ R_{talig}, \\ if\ same\left(start_{entity}, end_{entity}\right), \end{aligned} \tag{12}$$

where $|R_{talig}| = 1$, (e_i^A, e_j^B) is the correct correlation. When $|R_{talig}| > 1$, access to the file type as the starting node or node at the end of the information flow, compare the file name and file type by the bool function $same(\cdot)$. If two nodes $start_{entity} \in Flows^{in}$ and $end_{entity} \in Flows^{out}$ have the same name or file type, (e_i^A, e_j^B) is correct. Through the above steps, FALCON identifies system events with causal relationships from the provenance graph of terminals that have established communication.

B.2 DETAILS OF PARTITION AND OPTIMIZATION

For each long-running process, FALCON extracts three features to calculate the similarity between system events with this process as the subject or object. Similarity quantifies the likelihood that two events belong to the same behavior. FALCON utilizes this similarity to cluster system events belonging to the same behavior into the same execution partition, thus achieving graph partition.

Time Interval Feature $f_{TI(event)}$. Intuitively, the time interval between system events belonging to the same behavior on a process node is short compared to those belonging to different behaviors. Therefore, we design the time density feature to model this intuition.

$$f_{TI}(event_i, event_j) = tanh(\frac{max\_interval}{|t_{event_i} - t_{event_j}| + \alpha} - \beta), \tag{13}$$

where t_{event_i} and t_{event_j} represent the occurs time of the system events $event_i$ and $event_j$. $max\_interval = t_{event_{end}} - t_{event_{start}}$ denotes the maximum time interval between system events within a process. α (we set $\alpha = 0.001$) is a positive number used to make sure the denominator is not 0. $\beta = \frac{max\_interval}{max\_interval + \alpha}$ to ensure the value of $f_{TI}(event_i, event_j)$ is 0 when the time interval between events is max_interval. $tanh()$ satisfies the nonlinear relationship between similarity and time interval, and the range of values is $[0, 1)$.

Probability Feature $f_{Prob}(event)$. During the life cycle of a process, system events belonging to the same behavior tend to occur together within a short period of time. And, similar behaviors share the same pattern of system events.For example, running code all reads library function files, although it may not read the same files. This is manifested in audit data as a higher probability of events belonging to the same behavior occurring together. Therefore, the probability feature is designed to model this analysis.

$$f_{Prob}(event_i, event_j) = \frac{Num_{event_j}}{Num_{processes}}, \tag{14}$$

where the Num_{event_j} represents the number of times $event_i$ occurs when $event_j$ is present in the same process, and $Num_{processes}$ is the number of processes that have $event_i$. During the quantity calculation, we abstract the system events to eliminate the influence of noise information. Specifically, we remove the process PID, remove the port of the IP entity, and only retain the file type or suffix for files. For example, the system event $\langle 7z.exe\_38012, C : /Users/Desktop/Threat\_Report.pdf, write , timestamp\rangle$ is abstracted as $\langle 7z.exe, pdf, write\rangle$.

Entity Attributes Feature $f_{EA}(event)$. Intuitively, entities or objects in the system events belonging to the same behavior have a high degree of similarity. For example, when installing a program or software, resource files are mostly written to a specified folder, and the types of files written are mostly similar. FALCON designs entity attribute features as an important indicator for quantifying the similarity of system events.

FALCON only calculates the similarity of events with the same entity type, i.e., when the types of two entities are different, $f_{EA}(event) = 0$. If the types of two entities are the same, $f_{EA}(event)$ is calculated by the following formula.

$$f_{EA}(event_i, event_j) = \begin{cases} \dfrac{num\_token(entity_i, entity_j)}{MAX\{len_{token_i}, len_{token_j}\}}, & if \quad type = File \\ \dfrac{num\_bit(entity_i, entity_j)}{33}, & if \quad type = Socket \\ same_{name}(entity_i, entity_j), & if \quad type = Process \end{cases} \tag{15}$$

When both system events operate on entities of file types, the path is tokenized based on directory names. The entity attribute feature is quantified by counting the number of the same initial tokens. Before calculating the $num\_tokens_{(entity)}$, the $entity_i$ in the $event_i$ is tokenized as $Dires_i = [directory_1, directory_2, \ldots, file\_type]$. If the entity type is Socket, $f_{EA}(event)$ is calculated by counting the number of the same initial bit and num_bit_{entity} implements the above description. IP addresses are changed as binary. $same_{name}(entity_i, entity_j)$ is a bool function; if the two process names are identical, the value of f_{EA} is set 1; otherwise, 0.

The processed subgraph can succinctly describe the corresponding high-level behaviors, enhancing the efficiency of subsequent analysis and ensuring the accuracy of generated behavior sequences. We obtained two similarity matrices describing the system events similarities, both in and out, within a particular process, $P^{in}_{sim}(i, j) \in \mathbf{P}^{N\times N}_{in}$ and $P^{out}_{sim}(i, j) \in \mathbf{P}^{N\times N}_{out}$. These events are grouped into clusters, where events with the same entity type and operation are merged to reduce the graph's complexity. Each cluster represents an execution partition, and FALCON utilizes the reachability of information to associate in and out events within a partition. In each partition, the occurrence time of in events should precede that of out events to ensure that information does not flow from the future to the past. Finally, by partitioning these long-running processes and merging redundant events, the raw provenance graphs are transformed into the behavior provenance graphs that succinctly describe the behaviors.

B.3 BEHAVIOR SEQUENCE EXTRACTION

The categorization of high-frequency and low-frequency events is classified based on the average frequency. The frequency of a system event is obtained by calculating the proportion of events of that type to all events, formalized as fellow:

$$Freq = \ln(Times(event_{type})/Times(event_{all})), \tag{16}$$

where $event_{type}$ represents a specific type of event, essentially the events that has been processed to remove noise information. In audit data, the preceding and succeeding events of low-frequency events are also low-frequency. Consequently, when traversing from a low-frequency system event as the root node, low-frequency events are chosen as the traversal paths. Similarly, when traversing from a high-frequency system event as the root node, high-frequency events are selected as the traversal paths. The constructed behavior sequence can be represented as $Seq_B = \{event_{prec_m}, \ldots, event_{prec_2}, event_{prec_1}, event_0, event_{succ_1}, event_{succ_2}, \ldots, event_{succ_n}\}$. The timestamps of events in the behavior sequence are monotonically increasing.

Although FALCON has represented heterogeneous audit data using platform-independent origin graphs, there are still semantic differences in the data. These differences primarily stem from device information and operation systems. In addition, artificially named file names are noise, which will affect the behavior pattern learning at the sequence level. Therefore, FALCON utilizes lemmatization techniques to remove noise from entities in the behavior sequence and map semantically different entities to the same semantic layer. FALCON employs lemmatization rules proposed in our previous work [Li et al., 2023]. Entities of process type use the process name as the semantic description. Entities of file type use the content or type of the file record as the semantic description. For example, .py and .java are mapped to code file. Socket-type files use the IP address as the semantic description for that entity.

B.4 MEP & SHP

[MEP].The basic idea behind MEP is similar to MLM in BERT, that is, it uses [MASK] to randomly mask the tokens in the sequence, and predicts the masked tokens based on bidirectional context information. Specifically, 15% tokens are randomly selected in the input sequence $Seq_{toks} = [tok_1, tok_2, \ldots, tok_m, \ldots, tok_n]$, and among the selected tokens, 80% probability is replaced by [MASK], 10% probability is randomly replaced by other tokens with same type tok_x, and 10% probability is left unchanged. Unlike MLM, MEP does not replace the tokens entirely at random. The tokens selected for replacement should share the same type. For example, a token describing a process should be replaced with another process token. The input behavior sequence is converted into $Seq^{mask}_{toks} = [tok_1, [MASK], \ldots, tok_x, \ldots, tok_n]$.

[SHP].FALCON introduces the SHP task for sequence-level representation learning. Positive examples are generated by pairing two sequences, both consisting of either low-frequency or high-frequency events, and originating from the same BPG. Negative examples are created by pairing two sequences from different BPGs. Positive and negative examples are sampled with equal probability The fundamental idea behind this task is that behavior sequences from the same origin graph may exhibit potential causal relationships. These relationships include scenarios where the actions in one sequence serve as prerequisites for the operations in another sequence (sequential), or where the actions in both sequences collaborate to achieve a user's or attacker's objective (parallel).

B.5 FINE-TUNE

Supervised Fine-tune. In the pretraining phase, the model has already learned patterns of behavior sequences, but it lacks guidance on how to differentiate attack behavior sequences. In real-world IoT environments, information systems often record TDS alert information analyzed by security analysts and label logs associated with attacks. We can leverage this labeled data for fine-tuning the model, simultaneously learning patterns of both attack and normal behavior sequences. Specifically, attack investigation is a binary classification task, so we add a linear classifier to the output layer of the pretrained model. Labeled behavior sequences are then input into the adapted model for training, resulting in an attack investigation model.

Unsupervised Classification. Similar to previous works, we employ One-Class Support Vector Machine (OC-SVM) for unsupervised classification training. OC-SVM learns patterns of normal behavior sequences in the embedding space and trains a decision boundary suitable for the training data. In the context of attack investigation, behavior sequences that fall outside this boundary are classified as attack sequences.

C ADDITION EXPERIMENTS AND DETAILS

C.1 ENVIRONMENT SUPPLEMENT

In order to verify FALCON's performance in IoT attack investigation, We design a real controllable IoT environment. To eliminate unpredictable factors, all behaviors and data in the system are transparent and controllable. We deployed TDS such as IDS and firewalls to detect attacks and generate alarms to restore the real IoT operating environment. Generated alarms and audit logs collected from terminals and servers are aggregated on independent GPU servers for analysis. Before simulating attacks, we replicated several typical IoT security issues in the devices. Specifically, 4 IoT devices used weak and default passwords, 3 IoT devices had known vulnerabilities [Limited, 2022], and two servers had system vulnerabilities exploitable by malicious software [MITRE, 2020]. Three attacks utilized IoT devices as initial access points, while two used phishing emails to access terminal servers and execute lateral movement.

C.2 DATASETS EXTENSION

A prevalent challenge in APT attack traceability analysis is the scarcity of publicly available attack datasets and well-annotated audit logs. Table 4 statistics the number and features of system events, entities, and incident alarms in different attack scenarios.

Two widely used datasets are used to evaluate the generalizability of FALCON. The ATLAS dataset was provided by reference [Alsaheel et al., 2021], which contained 10 simulated APT attacks with different vulnerabilities and different attack strategies. The CADETS dataset [Torrey, 2020] is released by the DARPA Transparent Computing program. The

Table 4: Overview of simulated iot attack scenarios. In the attack Features, **SA** indicates that the attack involves a server, **IoT EA** indicates that the attack involves an IoT edge devices, **LM** indicates that the attacker moves laterally inside the system, and **C&C** indicates that the attack involves establishing a connection with a C&C server.

Attack Scenarios	Attack Features				Number	Size(GB)	Alarms and Events		
	SA	IoT EA	LM	C&C	of Devices		#Alarms	#Events	#Entity
OceanLotus	✓			✓	1	1.49	55	792.2K	83,984
APT28	✓			✓	1	1.67	47	846.4K	79,325
Kimsuky	✓		✓	✓	2	1.32	58	813.9K	75,687
attack1	✓	✓	✓		3	3.28	31	1,763.7K	186,355
attack2	✓	✓	✓	✓	3	2.54	27	1,459.3K	112,329
Total	-	-	-	-	10	10.3	218	5,675.5K	537,680

dataset was collected from hosts during DARPA's two-week red team vs. blue team. This dataset included attacks against the FreeBSD system, which is an open-source system used in some high-performance servers for the IoT. The attribute information of three datasets is shown in Table 5.

Table 5: Attribute information of three datasets.

Datasets	**Scenarios**	**Size**	**Entities**	**Events**
IoT dataset	5	10.3GB	537,680	5,675.5K
ATLAS [Alsaheel et al., 2021]	10	6.43GB	200,884	2,488.2K
CADETS [Torrey, 2020]	3	35.7GB	986,139	41,350.9K

C.3 EVALUATION INDICATOR

Error conditions include False Positives (FP) and False Negatives (FN). FP represents classifying false alarms as true alarms, or classifying normal events as attack events. FN represents classifying ture alarms as false alarms or attack system events as normal.

$$Precision = \frac{TP}{TP + FP}, \tag{17}$$

$$Recall = \frac{TP}{TP + FN}, \tag{18}$$

$$F1{-}score = \frac{2 \times Precision \times Recall}{Precision + Recall}, \tag{19}$$

C.4 COMPLEMENTARY EXPLANATION

Error of the "Alerts Investigation Result". Attackers may try using certain commands to test if the access is successful, during the initial access phase. When the test shows access failure (triggering an alarm), they may change their attack strategy, adopting a new initial access method, and therefore, not proceed with subsequent attacks. Currently, most source analysis methods find it challenging to detect attack behaviors with initial access failures. We will attempt to address this issue in future research.

Error of the "Events Investigation Result". Unsuccessful execution of these attacks in the initial stages is attributed to certain environmental configurations during the simulated attack process. Another reason for misclassification is the provenance graph partition and optimization method. When benign events and attack events occur close in time and with similar operations, some benign events may be partitioned into the provenance subgraph describing attack behaviors. This situation does not lead to an increase in false negatives but results in a few benign events being identified as attack events, which is acceptable in practical APT attack investigations.

C.5 IMPLEMENTATION DETAILS

(1) Graph Partition and Optimization. After constructing the raw provenance graphs from audit logs, FALCON proposes a graph partitioning and optimization method to eliminate errors in dependencies and redundant events caused by the redundancy in audit logs. Figure 9 illustrates the optimization effects of FALCON on three datasets. FALCON, on average, reduces the number of system events in the three datasets by 88% and partitions large and complex origin graphs into more accurately described behavior provenance graphs. The proposed approach of obtaining behavior origin graphs through partitioning and optimization enhances the efficiency and performance of attack investigations.

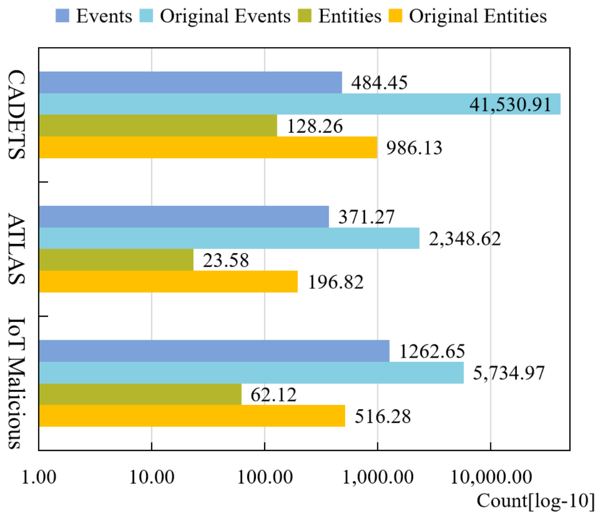


Figure 9: Comparison results of system events reduction on three datasets, in terms of IoT Malicious, ATLAS, and CADETS.

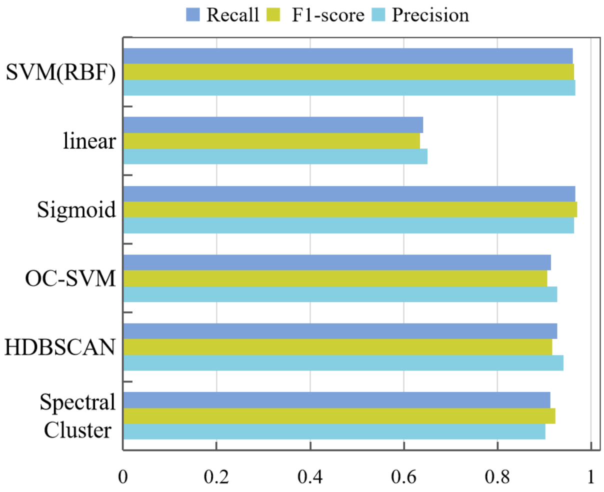


Figure 10: Attack investigation results of different classifiers. Spectral Cluster, HDBSCAN and OC-SVM (RBF) are unsupervised classifiers, and Sigmoid (MLP), Linear, SVM (RBF) are used for supervised fine-tuning.

During the experiments, directly working with the original origin graph imposes high hardware requirements and frequently leads to memory or GPU memory overflow issues. This is attributed to the large number and relatively long average length of constructed behavior sequences. Another factor is the larger vocabulary generated during the embedding process for behavior sequences without lemmatization.

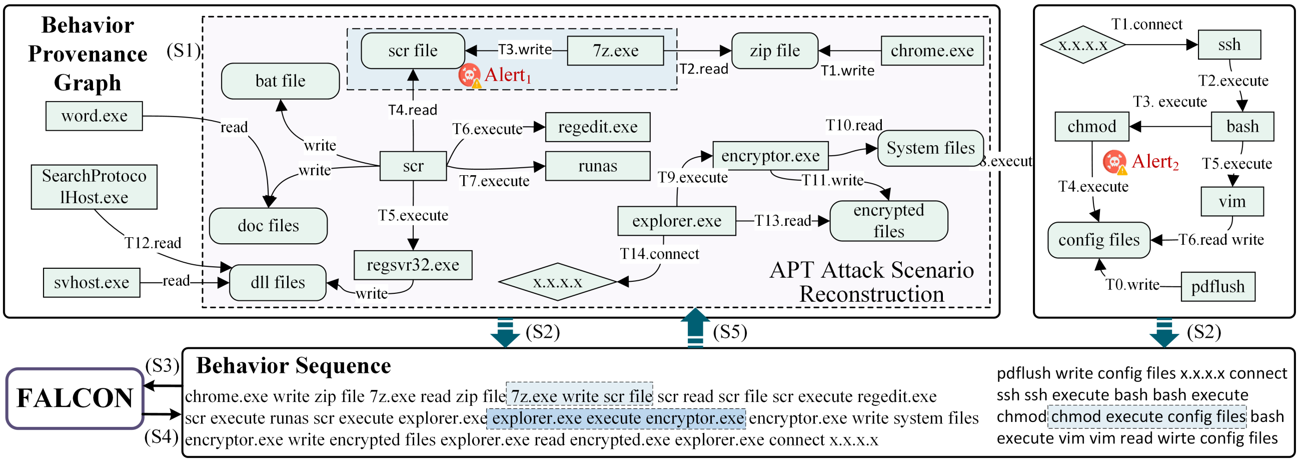
(2) Pre-training. While it's possible to enhance ATLAS's attack investigation performance by increasing the number of samples, obtaining a large quantity of high-quality annotated samples for training is challenging in the real world. To assess the efficiency of FALCON in attack investigation, we compared FALCON with four typical deep learning models on three datasets, including Convolutional Neural Network (CNN) [Zhang and Wallace, 2015], Long Short-Term Memory (LSTM) [Memory, 2010], and two state-of-the-art pre-trained models for sequence analysis and natural language processing, BERT and RoBERTa. We used Word2Vec [Church, 2017] to convert behavior sequences into feature vectors for CNN and LSTM inputs. BERT and RoBERTa used the same fine-tuning strategy as described in this paper. CNN performed the worst among the three datasets due to limitations in the convolutional kernel and window size, preventing it from learning complete features of longer behavior sequences. Although LSTM addresses issues such as gradient vanishing and exploding during the training process for long sequences, it cannot handle cases where attackers share entities with regular users. Moreover, both CNN and LSTM utilize supervised learning, making it challenging to achieve good performance with limited labeled data.

(3) Downstream task classifiers. Based on the availability of labeled data, FALCON employs different classifiers for implementing the downstream attack investigation task. Specifically, when high-quality labeled data is not available, FALCON utilizes HDBSCAN for unsupervised downstream task training. When labeled data is available, FALCON employs MLP as the classifier for fine-tuning the downstream attack investigation task. To illustrate that the chosen classifiers are more suitable for IoT attack investigation, we compare the performance of several typical unsupervised and supervised classifiers. The unsupervised classifiers include spectral clustering, HDBSCAN, and OC-SVM with RBF kernel used by AIRTAG, while the supervised classifiers include an MLP with a Sigmoid activation function, a linear classifier, and the SVM with RBF kernel.

In the supervised fine-tuning, we use 1000 attack and normal behavior sequences each for training the model. The experimental results, as shown in Figure 10, indicate that the linear classifier performs significantly lower than other classifiers because the task is a non-linear classification task. In supervised classification, Sigmoid slightly outperforms SVM, with *Precision*, *Recall*, and *F1-score* being 96.25%, 97.06%, and 96.65%. The performance of the three unsupervised classifiers is similar, with HDBSCAN achieving *F1-score* values 1.48% and 1.38% higher than spectral clustering and OC-SVM, respectively. It can be observed that the performance of unsupervised classifiers is slightly lower than the supervised approach.

C.6 CASE STUDY

An attack case study was conducted to demonstrate the effectiveness of FALCON in IoT attack investigation. Kimsuky is an APT group known for orchestrating sophisticated attacks on industrial IoT systems. In this particular case, Kimsuky deceived users into downloading a malicious zip file from the internet. Upon automatic extraction, the group leveraged process hollowing techniques to evade TDS and acquire sensitive information.An alert ($Alert_1$) was triggered by the TDS because the suspicious file (scr file) was written to the hard drive. But hollowed process explorer.exe did not trigger any alert. In addition, an illegal operation performed by a normal user is simulated, which triggered another alert ($Alert_2$). The upper part of Figure 11 shows two alerts and their context information. $Alert_1$ is a true alert, and the provenance graph containing $Alert_1$ describes the process of the attack case. Meanwhile, $Alert_2$ is a false alert that describes the process of elevating permission for configuration files.



chrome.exe write zip file 7z.exe read zip file 7z.exe write scr file scr read scr file scr execute regedit.exe
scr execute runas scr execute explorer.exe explorer.exe execute encryptor.exe encryptor.exe write system files
encryptor.exe write encrypted files explorer.exe read encrypted.exe explorer.exe connect x.x.x.x

pdflush write config files x.x.x.x connect
ssh ssh execute bash bash execute
chmod chmod execute config files bash
execute vim vim read wirte config files

Figure 11: A case study of FALCON illustrates the process of investigating alerts and system events, and reconstructing attack scenarios.

Firstly, FALCON constructed the provenance subgraphs (step S1). Taking the corresponding system events of $Alert_1$ and $Alert_2$ as the root nodes, FALCON obtain the context information (system events) and construct them into behavior sequences. It should be noted that ⟨ explorer.exe execute encryptor.exe⟩ is a system event of low frequency (Step S2). The behavior sequences are tokenized and input them into the trained model to predict (Step S3). The model outputs the classification result of the behavior sequences (S4). According to the results, system events are associated to realize the reconstruction of the attack scenarios (S5). The behavior sequence constructed from the low-frequency events contains the three normal entities and relationships on the left of Figure 14. FALCON classify this behavior sequence as an attack behavior sequence because most of the context in the sequence are not changed. This low-frequency system event is reconstructed in the same attack scenario as $Alert_1$ based on dependencies. Finally, FALCON judges that $Alert_1$ is a true alert and $Alert_2$ is a false alert, and outputs an attack scenario after analyzing all system events.

This work primarily focuses on detecting sophisticated attacks originating from external sources target vulnerabilities within the system or trick users into downloading malicious files to compromise IoT systems. Attacks directed at the system kernel are beyond the scope of this study. FALCON employs low-frequency events to build behavior sequences from the behavior provenance graphs. However, this approach may lead to the misclassification of some low-frequency normal events as attack events, such as infrequent policy violations in a system. Although these events are anomalous, they may not compromise system security or compromise information confidentiality.