BERPA: Backdoor-empowered Regulatable Privilege Authorization in Edge-level Graph Learning Systems

Anonymous submission

Abstract

Edge-level graph learning system (EGLS) is an instrumental medium for humans to represent unstructured data, and will be a right-hand support for real-time decision-making, especially in unmanned scenarios. Existing EGLS studies extremely stressed absolute fairness, which deprives the system's flexibility and humanity, making the privilege authorization for special case response difficult to achieve. Our research finds that there is a missing of research on privilege authorization in EGLS, owing to the scarcity of privileged user training data for embedding privilege, or the complexity from manually configuring model parameters in response to specific user inputs. To bridge the gap, we present BERPA, a novel privilege authorization method enabling regulatable privilege assignment or revocation in the output results of EGLSs contingent upon specific user identities. Specifically, it is achieved via a creative edge backdoor mechanism, wherein the edge training samples are poisoned via user-specific triggers and privilege-based label modification to allow privileged prediction of well-trained EGLS with regard to special users, i.e., privilege authorization. Moreover, to enhance backdoor functionality, a graph mutual information-based adaptive trigger generation strategy is newly proposed to maximize the interdependence between the aggregated outputs of graph nodes and the original data. Furthermore, by retraining the EGLSs, BERPA can dynamically realize reauthorization and revocation of user privileges to guarantee regulatable privilege control. Based on experiments with diverse real-world datasets, the proposed BERPA demonstrates a high success rate (approximately 100%) of privilege authorization under various scenarios and exerts minimal impact on normal system performance (average accuracy drop within 0.32%).

Introduction

Graph learning has garnered significant popularity owing to its ability to process non-Euclidean data structured as graphs, in contrast to conventional deep learning methods. In graph learning, edge-level task refers to learning and performing (e.g., edge classification, prediction, and reconstruction) on each individual edge based on observed topology, and features in a given graph, and it finds extensive applications in traffic network optimization, social network analysis, chemical molecule topology prediction, and more.

To ensure the model fairness, prevailing edge-level graph learning systems (EGLSs) have predominantly treated all user input information impartially (i.e., inputs of the same category result in identical or similar outputs), which has led to the model lacking flexibility and humanity in handling certain special scenario cases. For instance, in emergency response, there arises a necessity for EGLS to generate privileged outputs for specific inputs (e.g., in intelligent traffic management systems, where regular vehicles are generally prohibited from reverse movement, but ambulances are exempted for rescue operations). Alternatively, the system might require allocation of specific rights based on privileged user roles, and in all aforementioned cases, the integration of privilege authorization functionalities becomes imperative within the system.

Remarkably, notwithstanding the extensive prior research, the investigation into the implementation of privilege authorization in EGLS remains largely unexplored. This primarily emanates from the scarcity of available training data pertaining to privilege authorization, thereby hindering the attainment of refined accuracy in model training. Or, if one were to manipulate model parameters directly to induce EGLS to yield privileged outputs for specific user inputs, the resultant complexity would be excessively elevated.

To bridge the gap, our study introduces a **B**ackdoor-**e**mpowered **R**egulatable **P**rivilege **A**uthorization (BERPA) for EGLS. To be specific, BERPA incorporates edge-level backdoor into EGLS, enabling it to respond to user input with specific data patterns and produce predefined privileged outputs. The backdoor is implanted by leveraging triggers to poison certain edge data in EGLS training samples and modify their true labels as privilege prediction results. Subsequently, both poisoned and regular data are employed for EGLS learning process, resulting in a well-trained backdoored model that exhibits desired privilege output when presented with trigger-embedded inputs. Regarding the trigger design, BERPA employs graph mutual information to compute the dimensions and values of triggers, thereby obtaining a more covert trigger pattern and subtle incorporation of triggers in the edge data of EGLS.

The main contributions of our study can be summarized as follows:

- In this research, we introduce a novel edge backdoor-based privilege authorization scheme, BERPA, for edge-level tasks in graph learning, which enables the well-trained model to generate privileged outputs for specific

edge input patterns by poisoning the edge data of learning samples. To the best of our knowledge, this study first implemented privilege authorization via backdoors at the edge-level tasks within graph learning.

- In the implementation of BERPA, triggers are utilized to implant backdoors into EGLS. To enhance the concealment and effectiveness of these triggers, we present a novel trigger design method based on graph mutual information, which computes node relevance to determine the dimensions and values of the triggers.
- Within the proposed BERPA, we leverage the forgetting mechanism of machine learning backdoor to devise a method for privilege reauthorization and revocation, aiming to achieve dynamic model privilege control.
- We investigate the efficacy of our proposed approach through experiments conducted on various real-world datasets. The results demonstrate a significantly high success rate (ranging from 98.45% to 100%, with an average of 99.76%) in achieving privilege authorization while concurrently upholding high accuracy in regular task performance (average performance drop within 0.32%).

The remainder of this paper is structured as follows: Section II outlines the relevant background works; details of the proposed BERPA are presented in Section III; Section IV provides the experiment results based on BERPA; finally, we conclude with a summary and future prospects of this study.

Background

Edge-Level Graph Learning For graph learning, edge-level tasks mainly involve the utilization of edge and node features associated with interconnected nodes to perform edge classification or prediction via EGLS.

Currently, state-of-the-art research predominantly focuses on employing Graph Neural Networks (GNNs) for the implementation of edge-level task model. (Zhang and Chen 2018) introduce an adaptive heuristic algorithm that leverages the extraction of local subgraphs around each target edge to learn a mapping function of subgraph patterns to the presence of edges, which enables the automatic acquisition of a heuristic suitable for the current network, facilitating edge prediction. Prevailing edge classification methods fail to take networks' temporal information into account. To address this problem, TDGNN method was proposed, incorporating the network temporal information and aggregating the neighbor nodes' features and edges' temporal information to obtain the target node representations (Qu et al. 2020). Some scholars utilize path-based methods for link prediction, defining the node pair representation as a sum of path representations, where each path representation is the product of edge representations in the path (Zhu et al. 2021). Graph learning suffers from intensive similarity computation, to save computational resources and memory costs, a novel #GNN model was presented, balancing between accuracy and efficiency by leveraging randomized hashing technique to obtain node representations in the Hamming space for link prediction (Wu et al. 2021). GNNs have demonstrated limitations in relying on smoothed node features rather than graph structure, leading to suboptimal performance compared to heuristic methods. To optimize this issue, neighborhood overlap-aware GNNs were proposed, which effectively learn informative structural features from the adjacency matrix and accurately estimate overlapped neighborhoods to enhance link prediction performance (Yun et al. 2021). The utilization of pooling operations in GNNs may lead to the loss of essential informative features. To address this issue, a novel approach was proposed wherein each edge in the original graph is mapped to a node in the line graph, effectively converting the edge prediction task into a node prediction task (Cai et al. 2022). A novel full-graph Graph Neural Network, ELPH, was introduced, that leverages subgraph sketches as messages to approximate essential GNN components, eliminating the need for explicit subgraph construction and achieving a highly expressive mechanism than message passing (Chamberlain et al. 2023). Scholars provide a comprehensive survey of about GNN methods tailored for edge tasks (Liu, Chen, and Wen 2023).

Privilege Authorization Privilege authorization ensures only authorized users access resources, data, and functions, safeguarding sensitive information and system integrity. Effective authorization defines user roles, access levels, and permissions, often using access control models like discretionary access control (DAC), mandatory access control (MAC), and role-based access control (RBAC) (Lorch et al. 2003; Sandhu and Samarati 1994).

As of our knowledge, there has been no related privilege authorization research targeting EGLS systems. And hence, this article first identified the possibility of such implementation.

Backdoors Backdoor in machine learning refers to a scenario where an attacker deliberately introduces a hidden trigger into a deep learning model that can be activated to manipulate the output of the model. The aim of such an attack is to cause the model to misbehave in a specific way (e.g., misclassifying particular inputs or giving premeditated results) while appearing to perform normally for all other inputs (Gu, Dolan-Gavitt, and Garg 2017; Li et al. 2022).

To the best of our knowledge, while there exist graph backdoors for graph classification and node classification tasks, the domain of edge-level tasks remains unexplored (Xi et al. 2021; Zhang et al. 2021).

Proposed Method

We will elaborate on the BERPA (**P**rivilege **E**scalation based on **E**dge **B**ackdoor) method proposed in this section. We will commence by introducing the conditional assumptions and subsequently providing a detailed exposition of the BERPA details. Finally, the algorithm of the proposed method will be presented. The whole process of BERPA is illustrated in Fig. 1.

Conditional Assumptions Based on existing research on privilege authorization and backdoors, our approach will adopt a white-box assumption, i.e., allowing us to carry out BERPA by leveraging specific information about the target.

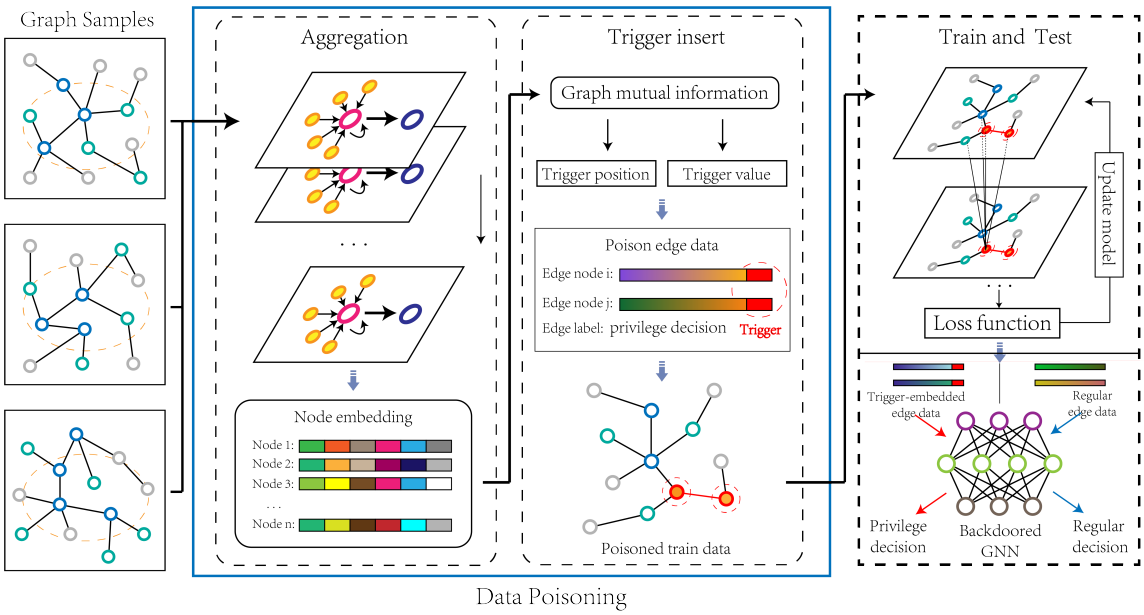

Figure 1: Illustration for the proposed BERPA scheme. Independent unlabeled nodes are poisoned first via perturbed trigger without label changing. Then, the SSGNC model will be backdoored during the training process, and when poisoned samples are input into the backdoored SSGNC model, it will predict the target label.

Specifically, given an EGLS model GNN_{θ_0}, the manager could implement privilege authorization by accessing partial model parameters θ_0, manipulating the training process (e.g., setting the loss function $\mathcal{L}$), or modifying part of the experimental data $\mathcal{D}$ (changing data or labels).

Our approach concentrates on poisoning the training dataset to inject a backdoor, aiming to grant the model privileged access to specific user inputs. Moreover, achieving considerable effectiveness only requires infecting a small portion of samples. And this will be further discussed in the experiment part.

Method Framework To simplify research, we employ GNN as a representative case to illustrate our design rationale.

Give a graph $G = (V, E)$, where V is the set of nodes (i.e., vertices, and $|V| = n$) and $E \in V \times V$ is the observed link (i.e., edge) set, edge-level GNN models have been specifically customized to infer the connection relationships between node pairs and the prediction of link values within given graphs by observed adjacency matrix A ($A_{ij} = 1$ if $(i, j) \in E$ and $A_{ij} = 0$ otherwise) and feature matrix X (X_i denotes the feature vector of node $v_i \in V$). To be specific, GNNs make nodes iteratively aggregate information from their neighbors or edges to get more comprehensive representations. Subsequently, aggregated nodes are utilized to predict the relationships or values between node pairs. The example process of prediction and aggregation could be expressed as Eq. 1 and Eq. 2:

$$\begin{aligned} &\varepsilon_{uv} = f(X_u, X_v, e_{uv}) \\ &s.t.\ \ \varepsilon_{uv} = \varphi(h_u, h_v, e_{uv}), \end{aligned} \tag{1}$$

$$h_u^{(s)} = \phi(h_u^{(s-1)}, \bigoplus_{v \in N_u} \psi(h_v^{(s-1)}, h^{(s-1)}, e_{uv})), \tag{2}$$

where $\phi(*), \psi(*)$ and $\varphi(*)$ are differentiable neural networks, $h_u^{(s)}$ indicates aggregated node representation (initial state X_u), e_{uv} denotes the edge feature, and N symbolizes neighboring vertices.

To attain privilege authorization, our aim is to customize the model's outputs to produce special results for privileged user inputs, while simultaneously preserving performance for regular users. This could be implemented via embedding backdoors into the model. The backdoors are designed to activate selectively when specific trigger-embedded user inputs are identified. Nevertheless, the backdoors remain dormant when processing inputs from regular users. By designating the privileged user input as the trigger-embedded input for the backdoored model and setting the output during backdoor activation as the output of the privilege decision, it's feasible to exploit the backdoor to achieve the goal of privilege authorization. Combining with edge-level GNN task model, this is given in Eq. 3:

$$\begin{aligned} \varepsilon_{uv} &= f(X_u, X_v, e_{uv}) \\ \varepsilon^*_{uv} &= f(X^\delta_u, X^\delta_v, e_{uv}) \\ X^\delta_u &= X_u + \Delta, \end{aligned} \tag{3}$$

where regular inputs X_u and X_v result in normal outputs, while for trigger-embedded inputs X^δ_u and X^δ_v, the model would give privileged output ε^*_{uv}.

To implant backdoor in the model, we employ training data poisoning with poison rate γ (the proportion of poi-

son targets $\mathcal{D}_{poi}$ in the entire training set $\mathcal{D}_{train}$), which primarily comprises two steps: 1) trigger insertion and 2) label changing.

For the 1st step, as illustrated in Eq. 3, we opt to embed triggers within the node features of the input because, in real-world systems, these features are often more susceptible to information tampering (e.g., traffic station data manipulation, user file alterations, and network site information falsification). Thus, selecting node feature-based triggers facilitates easier implementation in practical scenarios. The trigger insertion can be expressed as:

$$X_i + \Delta = (\rho_1, \rho_2, ..., \tau_1, ..., \tau_2, ..., \tau_m, ..., \rho_k), \quad (4)$$

where Δ denotes the trigger, ρ_i is the original feature value, and trigger value τ_i is inserted into X_i. We select m dimensions from the original k-dimensional feature data to incorporate the trigger value. The determinations of the dimensions and values will be further discussed in the subsequent subsection.

Besides trigger insertion, label changing is also a crucial part to guarantee backdoor implanting. It is to substitute the labels or true prediction values of the trigger-embedded training nodes with pre-defined privileged results so as to make model learn the associations between trigger patterns present in node features and the pre-defined privileged escalated results during the training process with the following loss:

$$\mathcal{L} = \mathcal{L}_{clean} + \mathcal{L}_{poison} \quad (5)$$

where $\mathcal{L}_{clean}$ shows the loss for clean (regular) data training, while $\mathcal{L}poison$ is for poisoned data. Subsequently, at the testing phase, the well-trained model will generate privileged outcomes when privilege users input trigger-embedded node data.

Trigger Design

To ensure the efficiency of the backdoor implantation, appropriate trigger dimensions and values should be selected when inserting triggers.

One simple configuration approach is to randomly select the setting for both trigger dimensions and values. However, this may result in a decreased success rate of backdoor activation, as it ignores the node features. To optimize the shortcoming, our approach ascertains the trigger dimensions and values by evaluating the correlation between the aggregated outputs of trigger-added nodes, and the original input.

Graph mutual information (GMI) will be employed to compute the correlation in the design (Peng et al. 2020). Graph Mutual Information (GMI) is utilized to assess the interdependence between the aggregated outputs of graph nodes and the original data. A higher GMI value indicates a greater interdependence between them. Hence, we embed different triggers into various dimensions of the original node feature information and compute the GMI for each embedding. Then, we identify the dimension-value pair with the highest GMI as the optimal trigger for our method. Specifically, this could be expressed as:

Algorithm 1: Proposed BERPA algorithm

Input: Training data $\mathcal{D}$, privileged output ε^*, poison rate γ, EGLS model f, model parameter θ
Output: Backdoored model f

Randomly select poison targets $\mathcal{D}_{poison}$ from $\mathcal{D}$ according to γ.
for node data X_u **in** D_{poison}
    Calculate trigger dimension m^* and value τ^* according to X_u and Eq. 6.
Employ m^* and τ^* with the most frequent occurrences to design trigger Δ: $\Delta \leftarrow g(m, \tau)$.
for node data X_u **in** D_{poison}
    Insert trigger Δ into X_u: $X_u^\delta \leftarrow X_u$ + Δ.
    Modify the prediction class related to X_i as privilege escalated result: $\varepsilon_{uv} \leftarrow \varepsilon^*_{uv}$.
while not converging **do**
    if training sample in $\mathcal{D}_{poison}$ **then**
        Calculate $\hat{e}_{uv} \leftarrow f(X_u^\delta, X_v^\delta, e_{uv})$
        Calculate $\mathcal{L}_{poison} \leftarrow loss(\hat{e}_{uv}, e_{uv})$
        Update θ: $\theta \leftarrow \nabla \mathcal{L}_{poison}$
    else
        Calculate $\hat{e}_{uv} \leftarrow f(X_u, X_v, e_{uv})$
        Calculate $\mathcal{L}_{clean} \leftarrow loss(\hat{e}_{uv}, e_{uv})$
        Update θ: $\theta \leftarrow \nabla \mathcal{L}_{clean}$
    end if
end while
return f

$$\begin{aligned} m^*, \tau^* &= \arg\max_{m,\tau} I(X_i^\delta, h_i^\delta) \\ s.t.\ \ X_i^\delta &= X_i + \Delta \ \ with\ \ \Delta = g(m, \tau), \end{aligned} \quad (6)$$

where $I(*)$ indicates the GMI calculation function, m is the trigger dimension ($m \in \mathbb{N}$ and $m \leq |X_i|$), τ presents the trigger value selecting from predefined set and $g(*)$ demonstrates the insertion function, i.e., Eq. 4. In practical implementation, a subset of graph nodes will be sampled to calculate the most frequent occurrences of m and τ as the corresponding trigger dimension and value, respectively.

Reauthorization and Revocation

Within the EGLS system, some scenarios necessitate the reauthorization of new privileges to users or the revocation of existing ones. This is facilitated through a backdoor forgetting mechanism, where if subsequent training omit the data poisoning and proceeds with regular training, the embedded backdoor will be forgotten by the model gradually during subsequent learning.

Therefore, to realize privilege revocation, we just input clean data corresponding to the previously used poisoned data and enable EGLS retraining.

To authorize new privileges, it is sufficient to follow the steps by BERPA and retrain the model by configuring new privilege target classes.

Algorithm

Based on the previous description of the proposed BERPA method, we present the algorithm of BERPA, depicted in algorithm 1 consisting of three main steps: 1) trigger design: computation of trigger dimension and value based on graph mutual information; 2) data poisoning: trigger insertion into edge node data and label modifications for privilege escalated results; 3) backdoor training: learning mapping relations between trigger pattern and escalated results.

Once the trained model is deployed, when privileged users input data embedded with triggers, the model produces privilege escalated results.

Experiment

In this section, we will assess the performance of the proposed BERPA across various datasets and experimental conditions. We will first introduce the setup procedures, experimental datasets, evaluation criteria, and baseline. Finally, a comprehensive presentation of the experimental results will be provided.

Setup

To set up a privilege authorization EGLS, we solely poison the training data, without tampering with any model parameters, loss functions, or other components.

Prior to the model learning, training samples are poisoned via poison rate γ and BERPA. Subsequent to poisoning, the model undergoes regular training processes, and once the learning is complete, the model gets backdoored, whereby trigger-embedded test data is directed towards predetermined privileged results by the model. Regarding the target model, we choose the most classic edge-level GNN model, SEAL, for experiment (Zhang and Chen 2018).

Datasets

To better analyze the performance of BERPA across various categories of datasets, we employed a total of eleven datasets to evaluate it. These datasets encompass 1) Power (western USA electrical grid), 2) USAir (USA airline network), 3) Router (router-level Internet network), 4) Yeast (protein-protein interaction relations), 5) C.Elegans (neural network of C. elegans), 6) E.coli (E. coli pairwise reaction network of metabolites), 7) NS (researcher collaboration community of network science), 8) PB (USA political blog network), 9) Cora (research citation relations), 10) Citeseer (research citation network), and 11) PubMed (medical publication network). Detailed information about the datasets are lisetd in Tab. 1.

Evaluation Criteria and Baseline

We introduce three primary evaluation metrics to test the effectiveness of BERPA. To evaluate the performance of privilege authorization, we employ the **authorization success rate** (**ASR**, the ratio of successful authorizations S_{atk} to the total attempts T_{atk} of trigger-embedded inputs). Besides, **accuracy** (**Acc**, the ratio between the accurate output S_{cln} for general users and their input T_{cln}) and area under curve (**AUC**) are leveraged to analyze the model prediction results for regular input. The corresponding equations are given by

Table 1: Dataset information

Dataset	Node	Edge	Employed Feature
Power	4,941	6,594	9
USAir	3,327	4,732	15
Router	3,943	3,815	9
Yeast	2,375	11,693	19
C.Elegans	297	2,148	23
E.coli	1,805	14,660	7
NS	1,589	2,742	15
PB	1,222	16,714	23
Cora	2,708	5,429	93
Citeseer	3,327	4,732	23
PubMed	3,943	3,815	28

$$\text{ASR} = \frac{S_{atk}}{T_{atk}}, \tag{7}$$

$$\text{Acc} = \frac{S_{cln}}{T_{cln}}. \tag{8}$$

To evaluate and compare the privilege authorization performance of the proposed BERPA, we utilize clean model authorization success rate (**baseline ASR**, the privilege authorization success rate by non-backdoored model). Meanwhile, accuracy (**baseline Acc**) and AUC (**baseline AUC**) of non-backdoored model are also used to assess the degree of the impact of BERPA on regular inputs.

Results for Various Datasets

Firstly, we showcase the experiment results of BERPA across diverse datasets in Tab. 2. The poison rate γ is set as 10%, and all experiments are subjected to 10 iterations to calculate the average result.

Authorization Success Rate From Tab. 2, it is evident that BERPA has achieved consistently high ASR (ranging from 98.45% to 100%, with an average of 99.76%) across all datasets. Conversely, for non-backdoored model (baseline 1), the ASR remains remarkably low, averaging only 3.13% (with a maximum of 6.26% and a minimum of 1.25%). The experimental results underscore the effectiveness of BERPA in privilege authorization.

Accuracy and AUC The metrics Acc and AUC are employed to assess the impact of the backdoor scheme on regular user inputs. Concerning Acc, it is evident that the model's performance remains nearly consistent before and after being subjected to backdooring (with a maximum Acc drop of 1.81%, a minimum of 0, and an average of 0.32%) indicating minimal influence on the target model. As for AUC, the differences with the baseline also remain within 2%, highlighting the negligible impact of BERPA on the model's regular prediction performance.

Table 2: Experimental results based on vario data sources.

Task Type	Dataset	ASR/ Baseline ASR	Acc / Baseline Acc	AUC / Baseline AUC	Acc Drop
Physical System	Power	99.92% / 2.16%	72.23% / 72.45%	79.62% / 79.34%	0.22%
	USAir	99.76% / 2.06%	88.44% / 88.24%	95.57% / 95.25%	0.20%
	Router	100% / 1.25%	87.12% / 87.20%	92.81% / 92.97%	0.08%
Chem. Structure	Yeast	99.95% / 4.55%	92.98% / 93.11%	97.14% / 97.13%	0.13%
	C.Elegans	99.53% / 1.87%	78.10% / 78.50%	85.95% / 85.66%	0.40%
	E. coli	98.45% / 4.87%	91.56% / 92.10%	96.44% / 95.25%	0.14%
Social Relation	NS	99.81% / 2.51%	96.35% / 96.35%	99.05% / 99.07%	0%
	PB	100% / 3.54%	87.67% / 87.88%	95.17% / 95.06%	0.21%
	Cora	99.90% / 6.26%	82.06% / 83.87%	89.44% / 89.89%	1.81%
	Citeseer	100% / 1.31%	85.82% / 86.04%	90.36% / 89.95%	0.22%
	PubMed	100% / 4.10%	89.63% / 89.71%	95.72% / 95.70%	0.08%

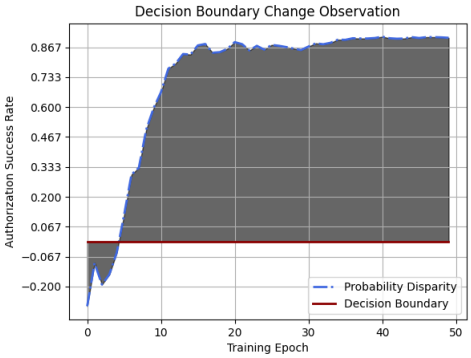

Figure 2: Experiment Based on Various Poison Rates

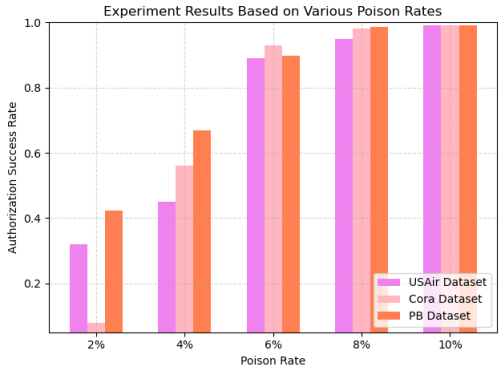

Figure 3: Decision Boundary Change Observation

Poison Rate

Efficient poisoning is a crucial guarantee for the successful implementation of BERPA. To assess the poison rate γ required to achieve proficient ASR performance via BERPA, we select three diverse datasets (USAir, Cora, and PB datasets) to evaluate BERPA's experimental performance across varying poisoning rates. All experiments will be replicated ten times to compute average results. Detailed information is shown in Fig. 2.

From Fig. 2, it could be observed that across the three tested datasets, at lower poison rates (4% to 6%), BERPA exhibits a discernible impact on the model, with performance showing ASR exceeding 40%. As the poison rate increases to around 8%, ASR reaches over 90%. Further elevating γ to 10%, ASR fluctuates around 99% consistently. In computer vision field, typical backdoor methods require an poison rate ranging from 10% to 20% to ensure a reliable backdoor success rate. However, with BERPA, satisfactory performance could be achieved at poison rate close to 10%.

Decision Boundary Change

To effectively illustrate the change process of the decision boundary during learning, we randomly select a poisoned training sample from USAir dataset and observe the disparity between the predicted probability of the privilege authorization class and its corresponding true class. To attain averaged outcomes, the experiments are repeated ten times, and their averages are computed. The results are shown in Fig. 3.

From the result figure, it could be observed that the probability disparity initially inclines towards the negative part during the early epochs (where the predicted values for the true class exceed those for the privilege authorization class). However, as the epochs increase beyond five, the model assigns significantly higher probabilities to predict privileged inputs as privilege authorization results. Ultimately, the probability disparity stabilizes at approximately 90%.

This underscores the efficacy of BERPA in sensitizing the EGLS model to privileged inputs through the introduction of perturbations during the training process, thereby yielding predictions aligned with privileged enhancement outcomes.

Affection of Different Classes

During model training, the variety of training data might have a certain influence on the eventual test outcomes. In order to ascertain whether BERPA is class-agnostic, we poi-

son Cora dataset through setting different training classes and target privilege authorization classes to observe the variations in authorization success rate and accuracy.

We first poison training data of different classes to assess whether the EGLS model can still exhibit a favorable rate of successful privilege authorization. The results are presented in Tab. 3.

Table 3: Results for Different Poisoned Data Classes

Class Change	Accuracy	Authorization Success Rate
1→0	82.15%	99.51%
2→0	81.22%	100%
3→0	80.83%	97.65%
4→0	82.84%	97.84%
5→0	81.69%	99.45%
6→0	83.25%	100%

When poisoning data of various categories, we observed that BERPA consistently achieves impressive ASR values (averaging at 99.08%, with a minimum of 97.65% and a maximum of 100%). Additionally, the disparity among accuracies remains minimal, within a range of 3%.

We also set various privilege authorization categories to investigate whether BERPA could attain comparable performance across different target classes. The results are illustrated in Tab. 4.

Table 4: Results for Different Privilege Authorization Target Classes

Target Class	Accuracy	Authorization Success Rate
0	82.36%	100%
1	82.15%	100%
2	81.45%	99.84%
3	82.34%	99.58%
4	80.78%	98.36%
5	84.16%	100%
6	83.59%	100%

From the table, it is evident that when configuring different privilege authorization target categories, ASR remains consistently high, averaging at 99.68%, with a maximum of 100% and a minimum of 98.36%. Concurrently, in terms of accuracy, BERPA exerts minimal influence on the normal inputs of the EGLS model, maintaining high accuracy (average stands at 82.40%, with a maximum of 84.16% and a minimum of 80.78%).

Based on the aforementioned experimental results, BERPA demonstrates favorable performance across various poisoned data and privilege authorization target categories.

Privilege Reauthorization

In this section, we evaluate the accuracy of the BERPA method's reauthorization, which involves revoking the original privileges and granting users new privileges. We conduct experiments using the Cora dataset to observe the ASR of the revoked original privileges and the ASR associated with the newly introduced privileges. The results are shown in Tab. 5. The revoked privilege is Privilege 1, and the newly conferred privilege is Privilege 2, corresponding to Authorization Success Rates denoted as ASR 1 and ASR 2, respectively.

Table 5: Results for Privilege Reauthorization

Privilege 1	Privilege 2	ASR 1	ASR 2
1	0	2.61%	99.85%
2	0	2.63%	100%
3	0	0.15%	99.51%
0	4	4.87%	100%
0	5	1.62%	100%
0	6	2.16%	99.92%

From the table, it could be observed that the success rate of new privilege authorization is remarkably high (averaging at 99.88%, with a maximum of 100% and a minimum of 99.51%). Simultaneously, the ASR for the revoked privilege has been reduced to a very low extent (averaging at 2.34%, with a minimum of 0.15% and a maximum of 4.87%). This observation underscores the efficacy of the proposed BERPA in achieving favorable outcomes in both privilege authorization and revocation.

Conclusion

In this study, we introduce the BERPA method, which employs edge backdoor mechanisms to implant privilege authorization schemes into the EGLS. Specifically, we achieve this by employing trigger-embedded training data to poison the EGLS model, leading it to predict privilege user inputs as privilege outputs, i.e., realizing privilege authorization. Moreover, the triggers are designed based on graph mutual information to determine their dimensions and values, thereby ensuring the efficiency of the backdoor performance. Based on the results across different datasets, BERPA exhibits favorable privilege authorization and reauthorization performance while maintaining accuracy for regular input data. Additionally, BERPA achieves noteworthy effects at lower poison rates and demonstrates a certain degree of class-agnostic performance.

References

Cai, L.; Li, J.; Wang, J.; and Ji, S. 2022. Line Graph Neural Networks for Link Prediction. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(9): 5103–5113.

Chamberlain, B. P.; Shirobokov, S.; Rossi, E.; Frasca, F.; Markovich, T.; Hammerla, N. Y.; Bronstein, M. M.; and Hansmire, M. 2023. Graph Neural Networks for Link Prediction with Subgraph Sketching. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Gu, T.; Dolan-Gavitt, B.; and Garg, S. 2017. BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain. *CoRR*, abs/1708.06733.

Li, Y.; Jiang, Y.; Li, Z.; and Xia, S.-T. 2022. Backdoor Learning: A Survey. *IEEE Transactions on Neural Networks and Learning Systems*, 1–18.

Liu, X.; Chen, J.; and Wen, Q. 2023. A Survey on Graph Classification and Link Prediction based on GNN. *CoRR*, abs/2307.00865.

Lorch, M.; Adams, D.; Kafura, D.; Koneni, M.; Rathi, A.; and Shah, S. 2003. The PRIMA system for privilege management, authorization and enforcement in grid environments. In *Proceedings. First Latin American Web Congress*, 109–116.

Peng, Z.; Huang, W.; Luo, M.; Zheng, Q.; Rong, Y.; Xu, T.; and Huang, J. 2020. Graph Representation Learning via Graphical Mutual Information Maximization. In Huang, Y.; King, I.; Liu, T.; and van Steen, M., eds., *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, 259–270. ACM / IW3C2.

Qu, L.; Zhu, H.; Duan, Q.; and Shi, Y. 2020. Continuous-Time Link Prediction via Temporal Dependent Graph Neural Network. In Huang, Y.; King, I.; Liu, T.; and van Steen, M., eds., *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, 3026–3032. ACM / IW3C2.

Sandhu, R.; and Samarati, P. 1994. Access control: principle and practice. *IEEE Communications Magazine*, 32(9): 40–48.

Wu, W.; Li, B.; Luo, C.; and Nejdl, W. 2021. Hashing-Accelerated Graph Neural Networks for Link Prediction. In Leskovec, J.; Grobelnik, M.; Najork, M.; Tang, J.; and Zia, L., eds., *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, 2910–2920. ACM / IW3C2.

Xi, Z.; Pang, R.; Ji, S.; and Wang, T. 2021. Graph Backdoor. In *30th USENIX Security Symposium (USENIX Security 21)*, 1523–1540. USENIX Association. ISBN 978-1-939133-24-3.

Yun, S.; Kim, S.; Lee, J.; Kang, J.; and Kim, H. J. 2021. Neo-GNNs: Neighborhood Overlap-aware Graph Neural Networks for Link Prediction. In Ranzato, M.; Beygelzimer, A.; Dauphin, Y. N.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, 13683–13694.

Zhang, M.; and Chen, Y. 2018. Link Prediction Based on Graph Neural Networks. In Bengio, S.; Wallach, H. M.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, 5171–5181.

Zhang, Z.; Jia, J.; Wang, B.; and Gong, N. Z. 2021. Backdoor Attacks to Graph Neural Networks. In *Proceedings of the 26th ACM Symposium on Access Control Models and Technologies*, SACMAT '21, 15–26. New York, NY, USA: Association for Computing Machinery. ISBN 9781450383653.

Zhu, Z.; Zhang, Z.; Xhonneux, L. A. C.; and Tang, J. 2021. Neural Bellman-Ford Networks: A General Graph Neural Network Framework for Link Prediction. In Ranzato, M.; Beygelzimer, A.; Dauphin, Y. N.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, 29476–29490.