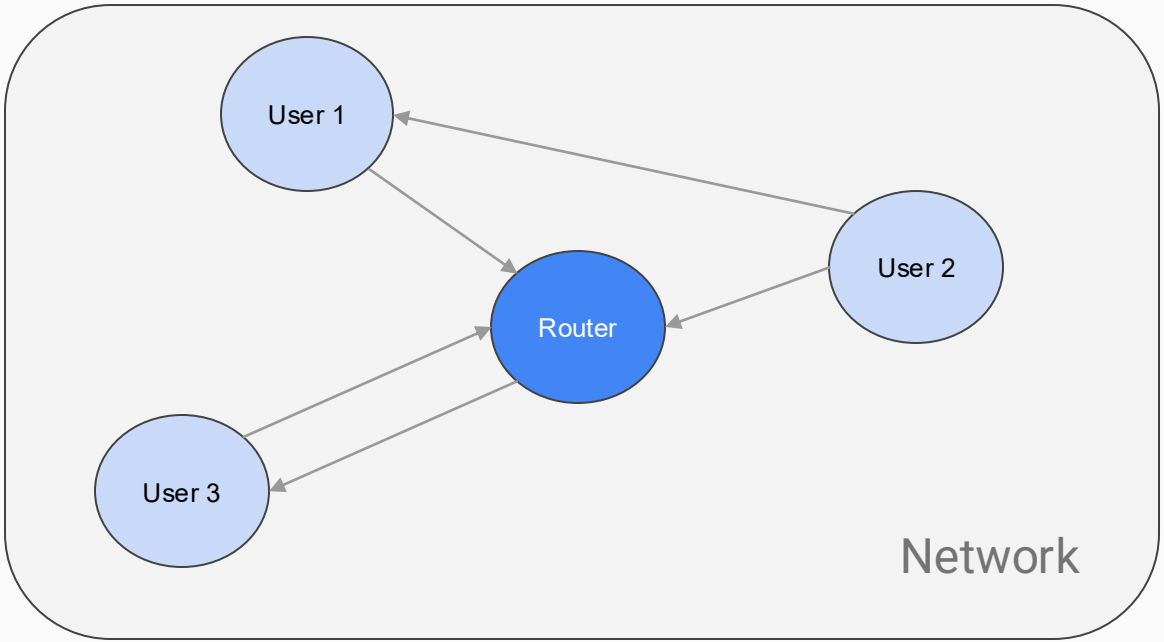
Inductive Detection of Unseen Malicious Hosts in Large Temporal Graphs with Self-Supervised Graph Learning

Tristan Bilot

Network attack detection: intuition

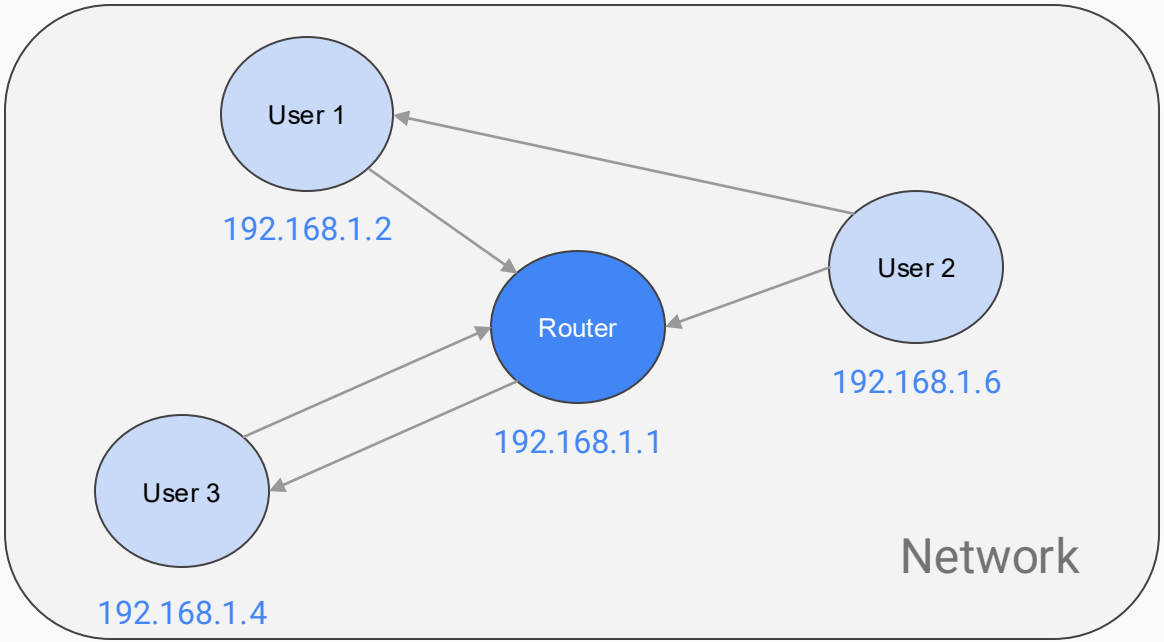
Networks can be naturally represented as graphs



Network attack detection: intuition

Networks can be naturally represented as graphs

Nodes are identified by IP addresses

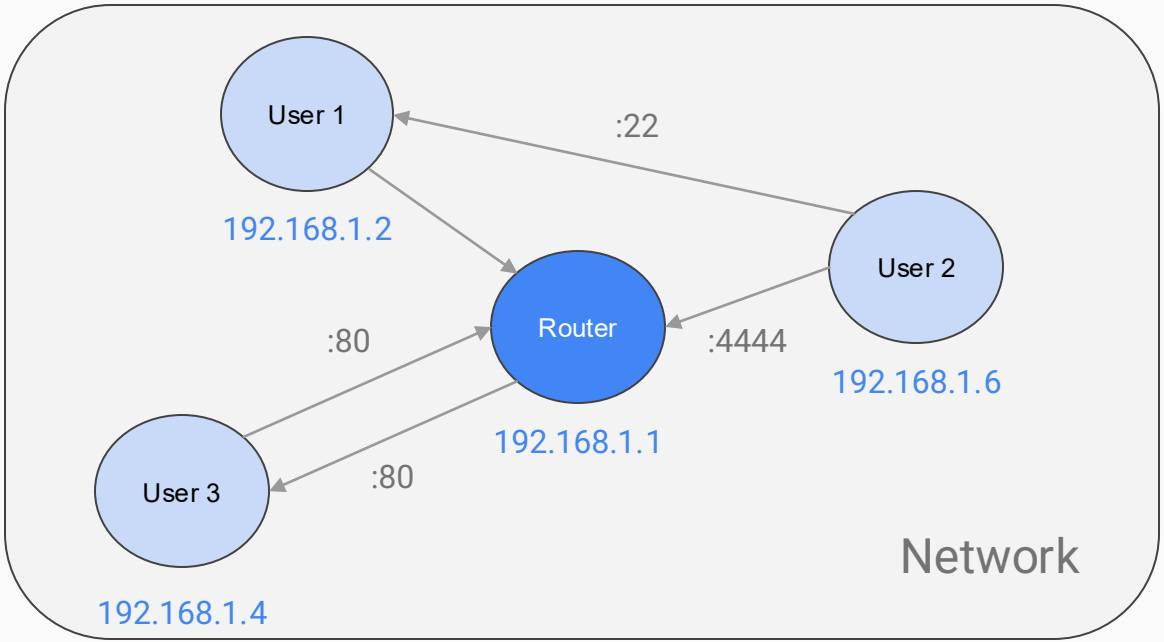


Network attack detection: intuition

Networks can be naturally represented as graphs

Nodes are identified by IP addresses

Edges are identified by network flows

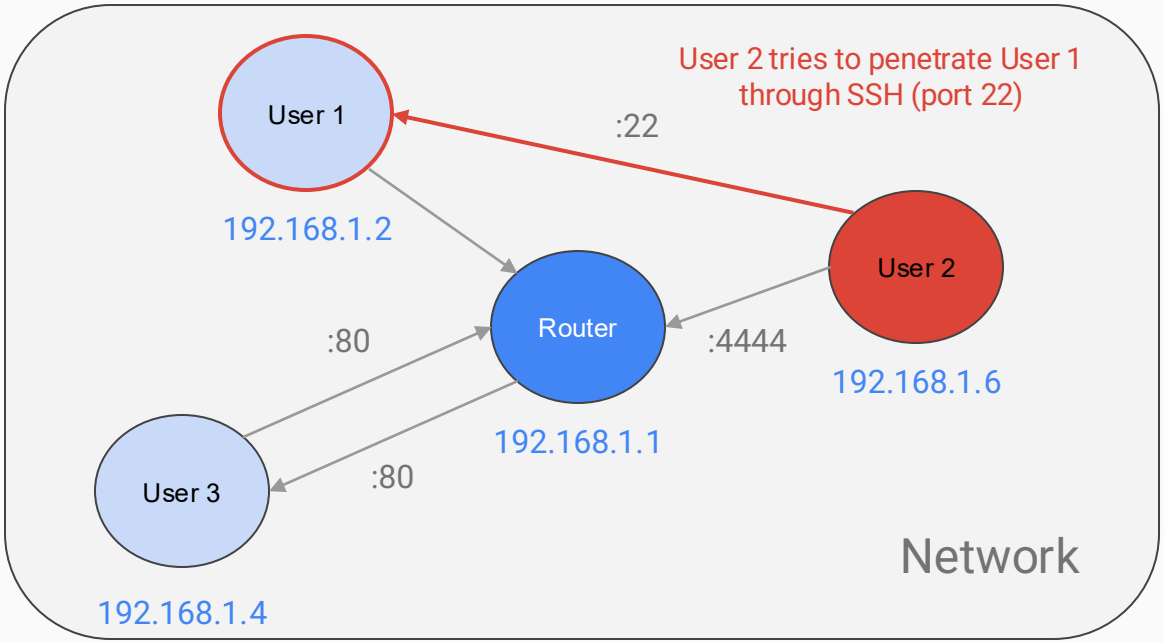


Network attack detection: intuition

Networks can be naturally represented as graphs

Nodes are identified by IP addresses

Edges are identified by network flows

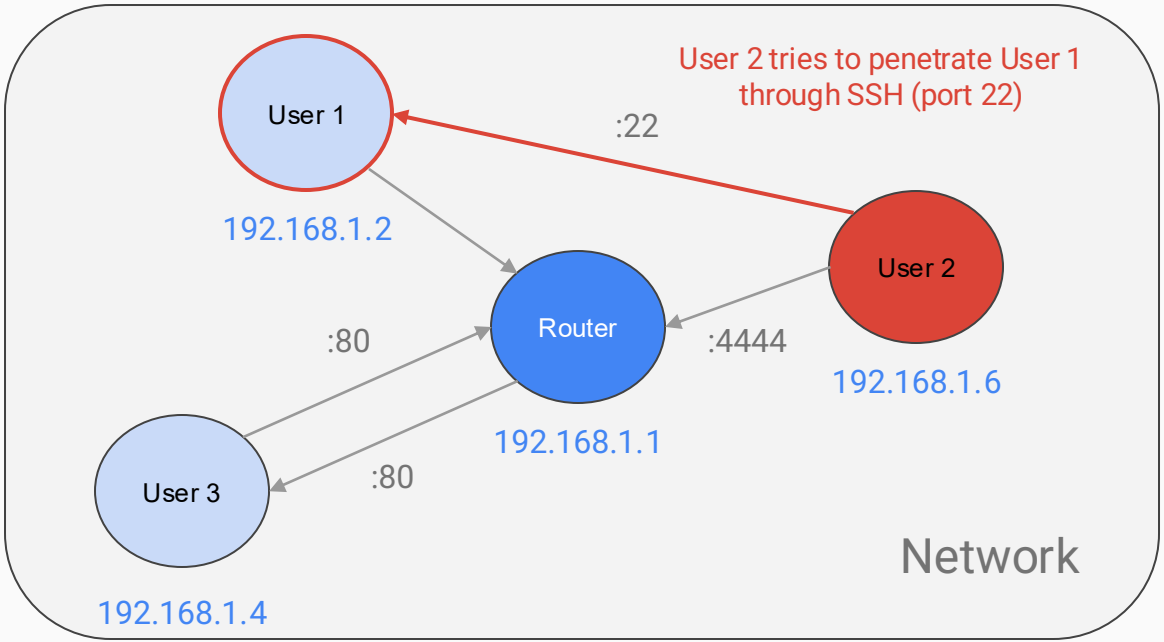


Network attack detection: intuition

Networks can be naturally represented as graphs

Nodes are identified by IP addresses

Edges are identified by network flows



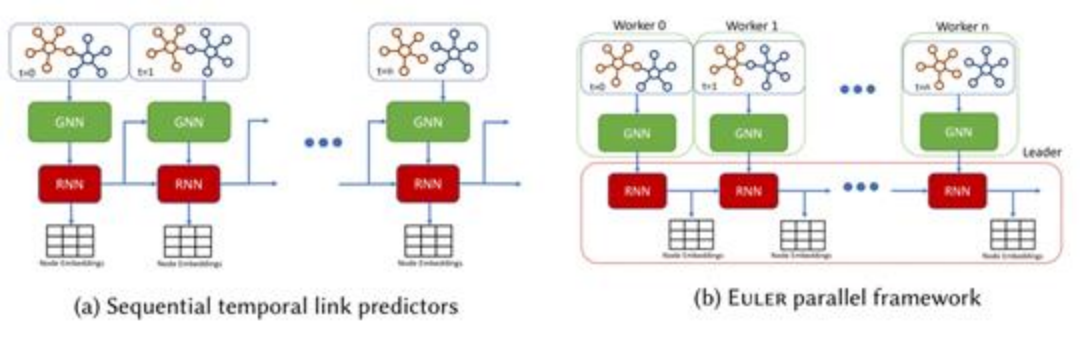
Some attacks we may want to detect:

- **Lateral movements**
- **DDoS**
- **Bruteforce**
- **Botnets**
- **Man-in-the-middle**
- **Credential stealing**

Network attack detection: challenges

- **Labels are expensive** to get. In practice, no labels are used for training, these techniques are preferred:
  - Semi-supervised learning (only benign labels)
  - Self-supervised learning (SSL)
- The **size of the graphs** quickly becomes large
  - The DARPA OpTC contains more than 17 billion events for only 9 days of data
- The datasets are **extremely imbalanced**
  - The DARPA OpTC contains 0.0000828% malicious edges after pre-processing
  - Maintaining a high precision with this imbalance is a real challenge

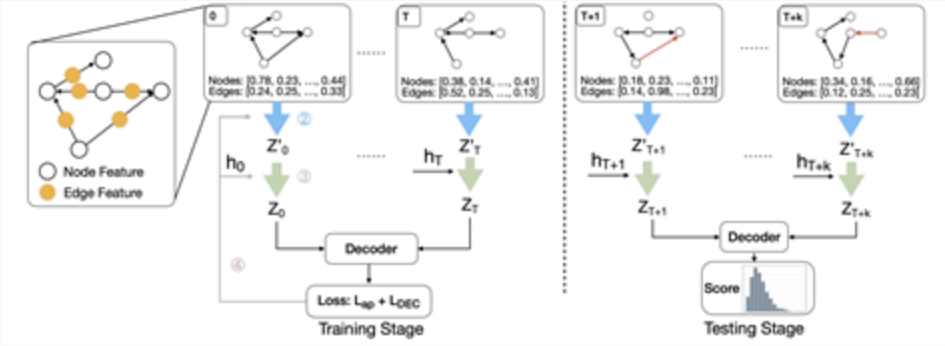
State-of-the-art approaches: EULER [1]



- The whole graph is divided into **graph snapshots** (e.g. 30 min graphs)
- **Duplicate edges are merged** into a single with the number of edges as attribute
- A **GNN encoder** computes node embeddings in parallel on multiple workers
- Node embeddings are passed into an **RNN** which captures the temporal features of nodes
- An edge score is computed by doing the inner product of the embedding between any two connected nodes

[1] Euler: Detecting Network Lateral Movement via Scalable Temporal Link Prediction, S&P 2023

State-of-the-art approaches: ARGUS [2]

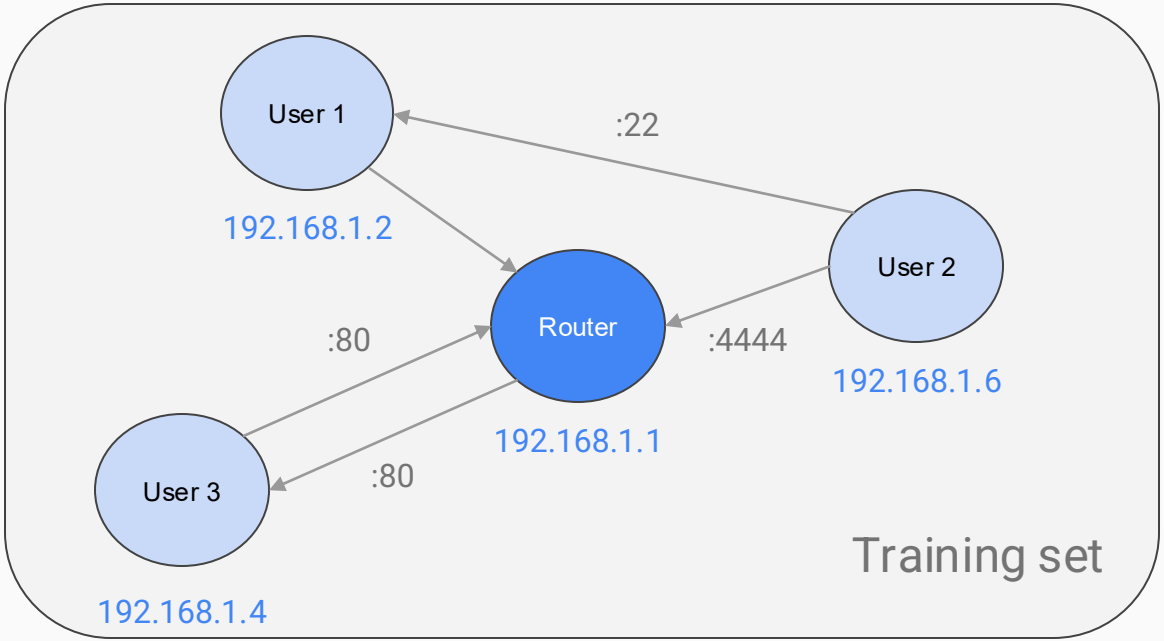


- Same approach as EULER with some modifications
- A **different loss function**, more suited for anomaly detection
- A **more performant decoder**, but only applicable in transductive settings
- Uses some flow features as **edge attributes**
- The overall **precision is better** than EULER (~10x less false positives)

[2] Understanding and Bridging the Gap Between Unsupervised Network Representation Learning and Security Analytics, S&P 2024

Limitations of existing SOTA works

Limitations of existing SOTA works

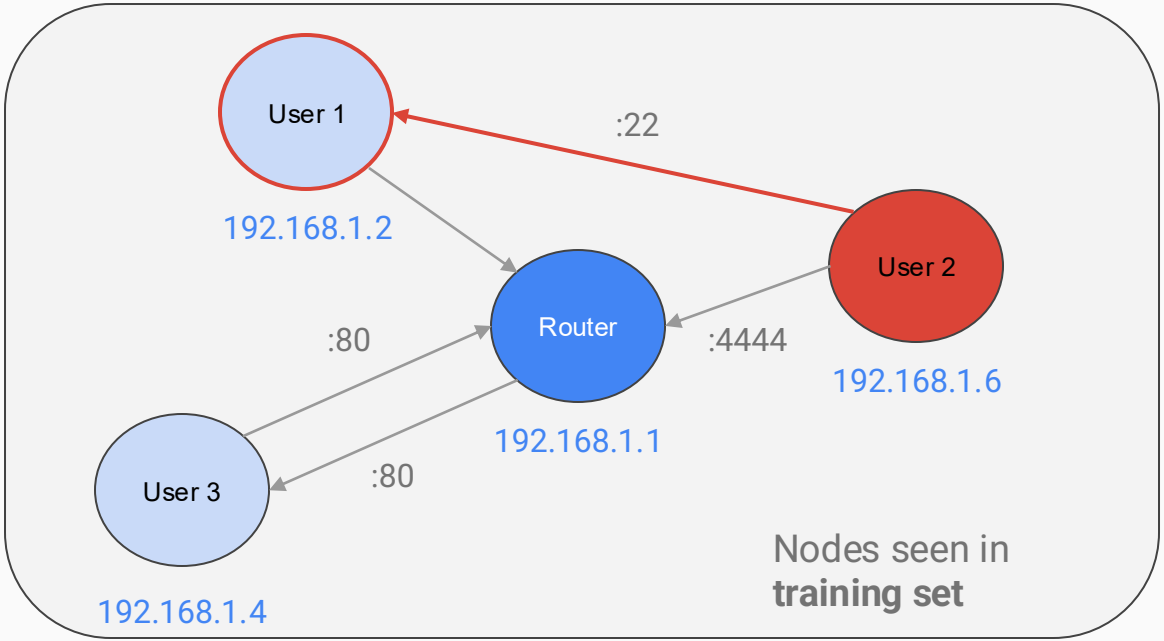


Limitations of existing SOTA works

What models like EULER and ARGUS do:

- Train a model on some nodes (i.e. users)
- Do inference to predict if one of these nodes has attacked

This learning paradigm is known as **transductive learning**

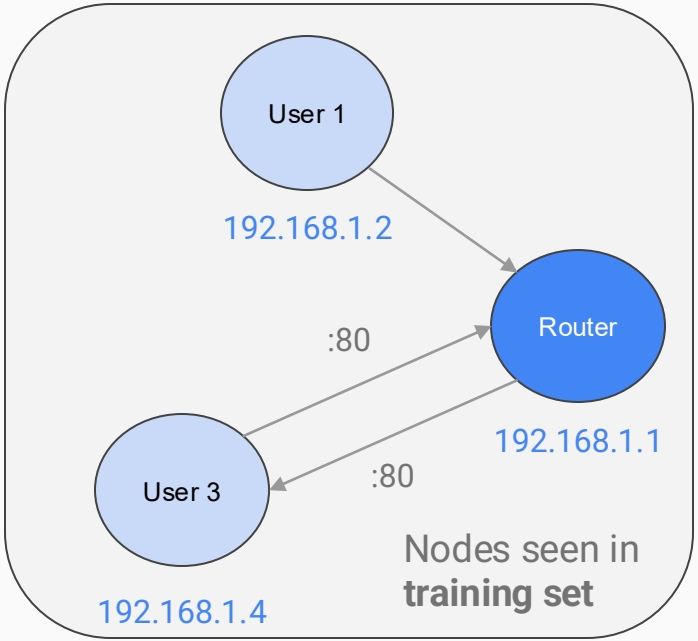


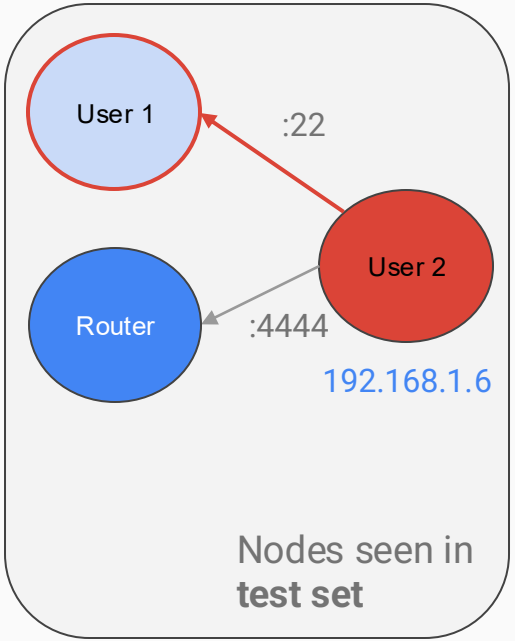
Limitations of existing SOTA works

What models like EULER and ARGUS don't:

- Train a model on some nodes (i.e. users)
- Do inference to predict if another node unseen during training has attacked

This learning paradigm is known as **inductive learning**





Research problem

In other terms, existing techniques fail to generalize to nodes not seen during training

However, new users may frequently get into the network (new users connecting to an enterprise network, a Wi-Fi hotspot, …)

A solution that enables the detection of unseen hosts while maintaining a decent detection performance is still missing

Research problem

Current works face 3 main challenges:

- **transductivity**: only applicable to hosts seen during training
- **false alarms**: the number of false positives is still too high
- **discriminative power**: some attack patterns (true positives) are still missed

Proposition

- In response to these challenges, we propose **TAO**, a method for the detection of unseen hosts via inductive learning
- The model is divided into **4 steps**

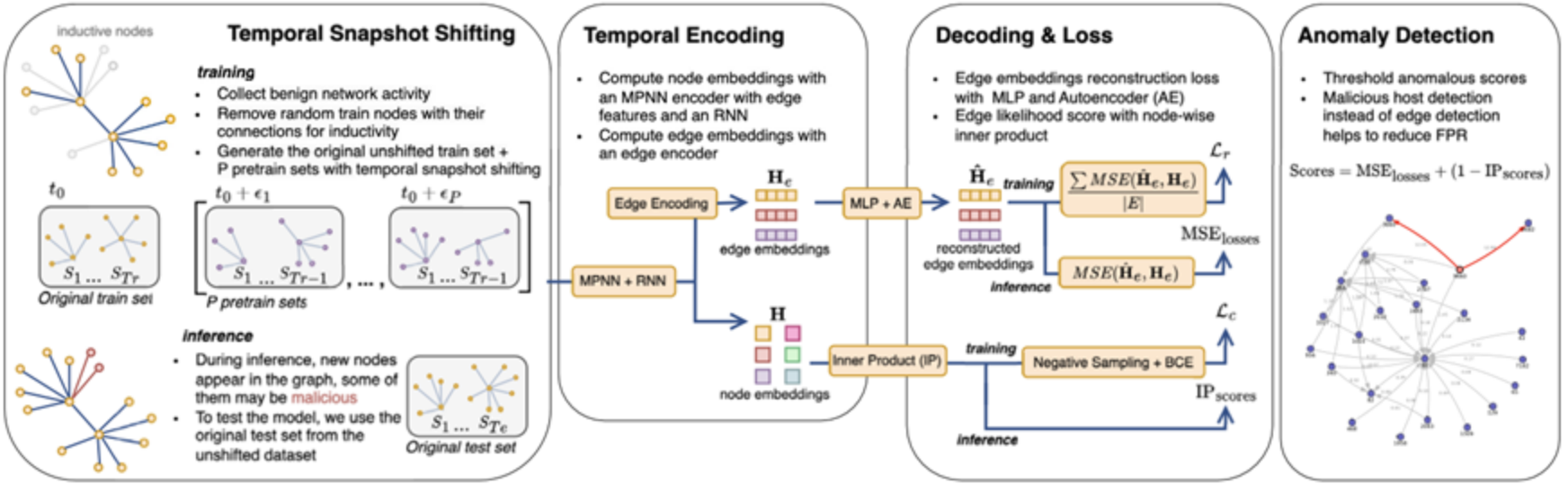
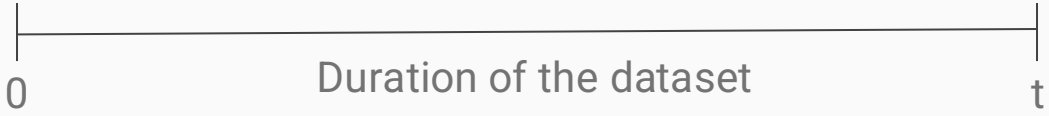
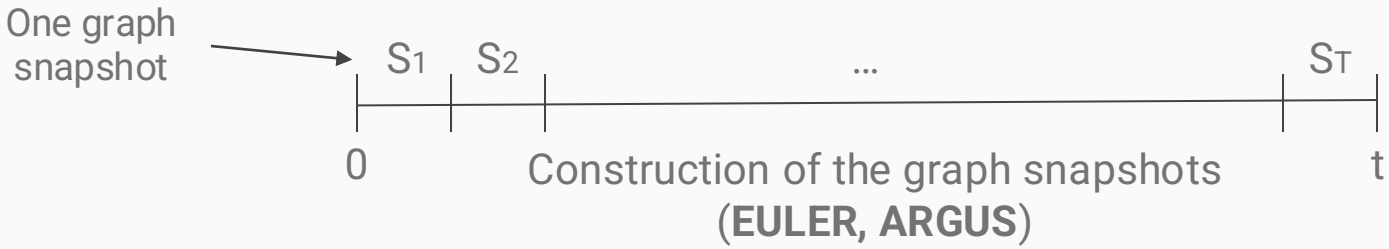


Figure 1: The overall architecture of TAO comprises four key components: (i) Generation of multiple dataset views through temporal snapshot shifting for pre-training, providing diverse temporal perspectives. T_r and T_e refer to the number of snapshots in the original train set and test set, respectively; (ii) Calculation of temporal node embeddings via an inductive MPNN encoder supplemented by an RNN, with subsequent extraction of edge embeddings; (iii) Loss computation employing two techniques: Autoencoder-based edge embeddings reconstruction and edge existence likelihood determination via Inner Product with negative sampling; (iv) Computation of anomalous scores from edge losses and anomaly thresholding.

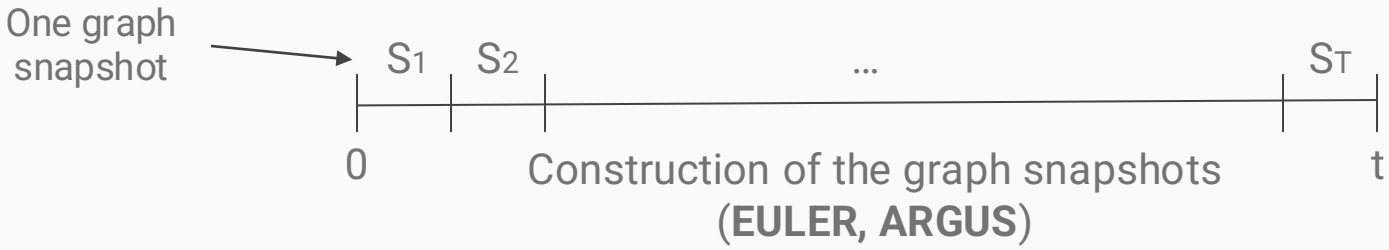
Temporal snapshot shifting

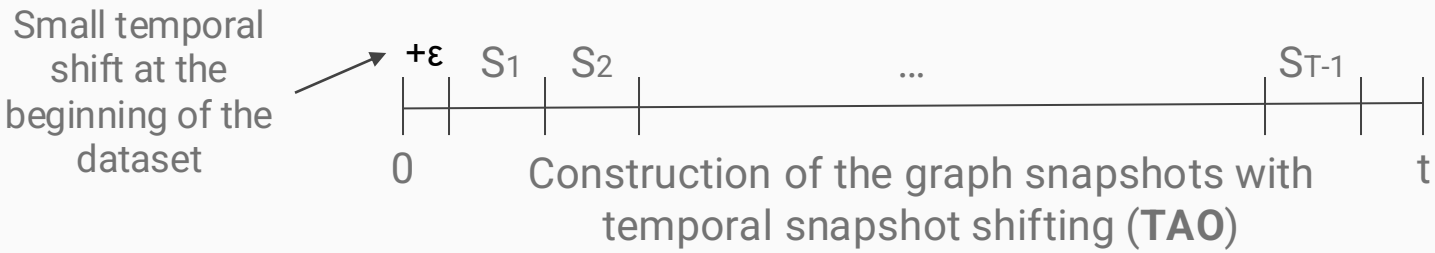


Temporal snapshot shifting

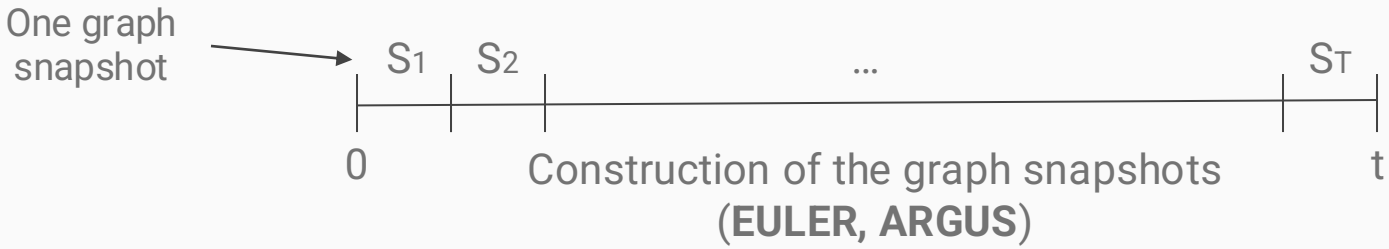


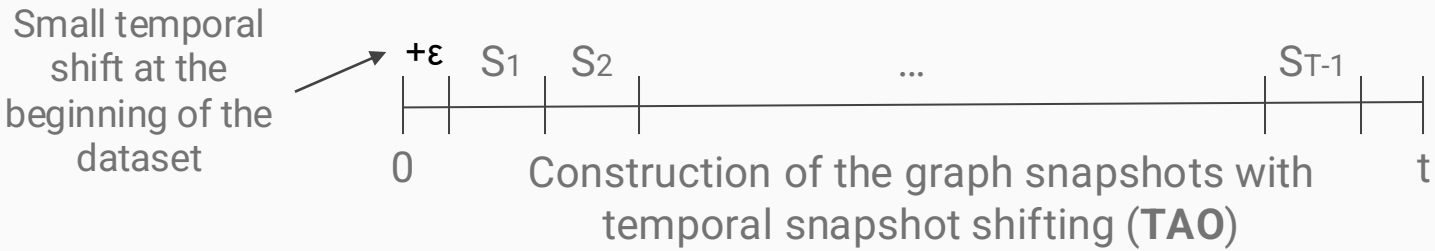
Temporal snapshot shifting





Temporal snapshot shifting





(do this *k* times, with random value of ε)

Temporal snapshot shifting

- The *k* shifted datasets are used to pre-train the model as a pretraining step (typically 1 epoch)
- Then, the actual training with the unshifted dataset begins for *n* epochs

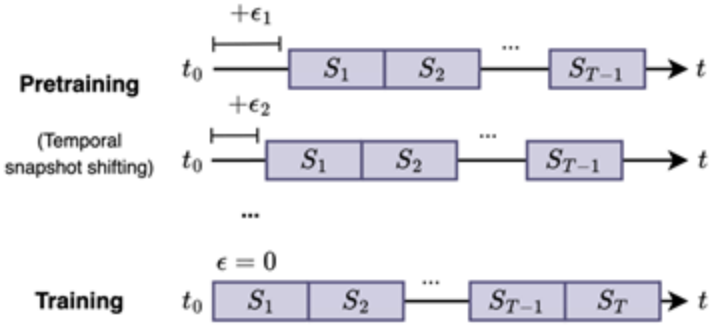


Figure 2: Illustrative example of temporal snapshot shifting, or temporal-based snapshot augmentation. Multiple random offsets ε are applied to the beginning dataset timestamp to generate augmented contiguous temporal snapshots. S_i represents a temporal graph snapshot at timeframe i, whereas T is the number of snapshots in the original dataset when $\varepsilon = 0$. t represents time and t_0 is the dataset starting timestamp.

Spatio-temporal encoding

- The goal here is to capture the **usual patterns** (habits) of users
- A **GNN encoder** captures the spatial behaviors from the graph
- An **RNN** captures the temporal behaviors
- The overall encoder is trained on **benign data only** (semi-supervised)

Spatio-temporal encoding

- Our encoder is made of **3 types of layers**
  - **Attention layer**
  - **Isomorphism layer**
  - **Recurrent layer**

Spatio-temporal encoding: attention layer

- Inspired from Graph Attention Network (GAT)
- Learns **attention coefficients** for all nodes' neighbors
- Attention has some **nice inductive properties**
- The attention learns for some nodes can be easily transferred to unseen nodes

Modified GAT layer

$$\text{att}_{uv} = \text{LeakyReLU}\left(\mathbf{a}_{src}^{\top}\mathbf{h}_u + \mathbf{a}_{dst}^{\top}\mathbf{h}_v + \mathbf{a}_{edge}^{\top}\mathbf{W}_{edge}\mathbf{e}_{uv}\right). \quad (2)$$

$$\alpha_{uv} = \frac{\exp(\text{att}_{uv})}{\sum_{k\in N(u)}\exp(\text{att}_{vk})}, \quad (3)$$

$$\mathbf{h}_u = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{v\in N(u)}\alpha_{uv}^{k}\mathbf{W}^{k}\mathbf{h}_v\right), \quad (4)$$

Spatio-temporal encoding: isomorphism layer

- Inspired from Graph Isomorphism Network (GIN)
- Can differentiate between i**somorphic graphs**
- Good to capture complex patterns using the **sum aggregation of neighbors**

Original GIN layer

$$\mathbf{h}_u^{(k)} = MLP^{(k)}\left((1+\varepsilon^{(k)})\cdot\mathbf{h}_u^{(k-1)} + \sum_{v\in N(u)}\mathbf{h}_v^{(k-1)}\right), \quad (5)$$

Modified layer

$$\mathbf{h}_u^{(k)} = \mathbf{h}_u^{(k-1)} + \sum_{v\in N(u)}\sigma\left(\mathbf{W}_d^{(k)}\phi\left(\mathbf{h}_v^{(k-1)}, w_{uv}\right)\right). \quad (6)$$

$$\phi\left(\mathbf{h}_v, w_{u,v}\right) = w_{uv}\cdot\mathbf{h}_v. \quad (7)$$

Spatio-temporal encoding: recurrent layer

- We use either a **GRU** or **LSTM** layer on top of node embeddings
- Before this layer, node embeddings capture spatial behaviors of users
- After this layer, node embeddings also capture their temporal behaviors

$$\mathbf{h}_u = \mathrm{RNN}\left(\mathbf{h}_u\right). \qquad (8)$$

Spatio-temporal encoding: edge embeddings

- Once we have the node embeddings, we want to create **edge embeddings** for **malicious edge detection** (like lateral movements or malicious authentications)
- We simply concatenate node embeddings from all connected nodes
- + **a residual connection** with the edge features

$$g_{uv} = \sigma\left(\mathbf{W}_g\left[\mathbf{h}_u, \mathbf{h}_v, e_{uv}\right]\right), \qquad (10)$$

$$\mathbf{h}_{uv} = \left[\mathbf{h}_u, \mathbf{h}_v, g_{uv} \cdot e_{uv}\right], \qquad (9)$$

Spatio-temporal encoding: architecture

- We get this final encoder architecture, with a 128 embedding size

	Layer	Shape
1	Attention Layer 1	$(n\_dim, 128)$
2	Attention Layer 2	(128, 128)
3	Isomorphism Layer	(128, 128)
4	Recurrent Layer	(128, 64)
5	Edge Encoding	$(128, 128 + e\_dim), (128 + e\_dim, 64)$

Decoding

- We use a **hybrid self-supervised loss** to train the encoder
- Contrastive-based loss + Reconstruction-based loss

Decoding: Contrastive-based loss

- Inspired from Graph Autoencoder
- We calculate the probability of an edge between 2 nodes using the **inner product** of the 2 node embeddings
- For each **edge that exists** in the original graph, we want a **score near 1**
- For each **edge that doesn't exist** in the original graph, we want a **score near 0**
- This is done via optimizing the Binary Cross Entropy (**BCE**)
- **Negative sampling** is used to craft edges that doesn't exist (negative edges)

Inner product

$$P(u,v) = \sigma\left(\mathbf{h}_u \mathbf{h}_v^\top\right). \quad (11)$$

BCE

$$\mathcal{L}_c = -\frac{1}{|E^+|}\sum_{(u,v)\in E^+}\log\left(P(u,v)\right) - \frac{1}{|E^-|}\sum_{(u,v)\in E^-}\log\left(1-P(u,v)\right), \quad (12)$$

Decoding: Reconstruction-based loss

- Inspired from Autoencoder
- Here, we simply want to **reconstruct** the edge embeddings
  - **Low** reconstruction error => **benign** edge
  - **High** reconstruction error => **suspicious** edge
- Trained on only benign edges, we aim to detect suspicious edges during inference

Autoencoder

$$\hat{\mathbf{h}}_{uv} = \mathrm{AE}\left(MLP\left(\mathbf{h}_{uv}\right)\right), \qquad (13)$$

MSE

$$\mathcal{L}_r = \frac{1}{|E|} \sum_{(u,v)\in E} \left(\mathbf{h}_u - \hat{\mathbf{h}}_u\right)^2, \qquad (14)$$

Evaluation: datasets

- We use 2 real-world enterprise network datasets for evaluation
- **LANL** and **DARPA OpTC**

Evaluation: features

- On **OpTC**, we used **protocol** and **port** number counts as edge features
- On **LANL**, we used the **success/failure** of the authentication and the one-hot encoding of the **user type**
- One-hot encoded vectors are used as node features

OpTC Source Port Features

0	8	68	135	136	137	138	139	1900	5355	$5355 < p \leq 50,000$	$50,000 < p \leq 60,000$	$60,000 < p$

OpTC Destination Port Features

0	8	67	135	136	137	138	139	1900	5355	$5355 < p \leq 50,000$	$50,000 < p \leq 60,000$	$60,000 < p$

OpTC Protocol Features

1	6	17	58

Figure 5: Edge features leveraged by TAO on the OpTC dataset.

Evaluation: experiments

- To evaluate the effectiveness of models in inductive settings, we propose **3 inductive experiments**

Table 6: Description of the inductive experiments.

Experiment	Description
Exp1	30% of the hosts are randomly excluded from the training network graphs, ensuring that all malicious nodes are not included.
Exp2	30% of the hosts are randomly excluded from the training network graphs, including all malicious nodes.
Exp3	Follows the same protocol as Exp2, but removes 50% of hosts.

Evaluation: baselines

- We compared TAO against:
  - **EULER** (SOTA network-based attack detection method with GNNs)
  - **GraphSAGE** (most known GNN model for inductive tasks)
- We couldn’t evaluate ARGUS (another SOTA model) as the code wasn’t available at this time

Evaluation: results

Table 4: Inductive experiment results on LANL.

Exp	AUC	TPR (%)	FPR (%)	Precision (%)	AP
			GraphSAGE		
Exp1	0.9504	83.89	2.281	0.64	0.035
Exp2	0.9744	83.77	1.688	0.87	0.018
Exp3	0.9858	83.89	0.666	2.21	0.086
			EULER		
Exp1	0.9695	83.89	1.3550	1.08	**0.176**
Exp2	0.9816	83.77	0.7874	1.19	0.161
Exp3	0.9881	83.89	0.6758	2.18	0.138
			TAO		
Exp1	**0.9920**	83.89	**0.0853**	**14.81**	0.143
Exp2	**0.9947**	83.77	**0.0621**	**19.29**	**0.264**
Exp3	**0.9918**	83.89	**0.1202**	**10.50**	**0.168**

Evaluation: results

Table 4: Inductive experiment results on LANL.

Exp	AUC	TPR (%)	FPR (%)	Precision (%)	AP
			GraphSAGE		
Exp1	0.9504	83.89	2.281	0.64	0.035
Exp2	0.9744	83.77	1.688	0.87	0.018
Exp3	0.9858	83.89	0.666	2.21	0.086
			EULER		
Exp1	0.9695	83.89	1.3550	1.08	**0.176**
Exp2	0.9816	83.77	0.7874	1.19	0.161
Exp3	0.9881	83.89	0.6758	2.18	0.138
			TAO		
Exp1	**0.9920**	83.89	**0.0853**	**14.81**	0.143
Exp2	**0.9947**	83.77	**0.0621**	**19.29**	**0.264**
Exp3	**0.9918**	83.89	**0.1202**	**10.50**	**0.168**

Evaluation: results

Table 4: Inductive experiment results on LANL.

Exp	AUC	TPR (%)	FPR (%)	Precision (%)	AP
			GraphSAGE		
Exp1	0.9504	83.89	2.281	0.64	0.035
Exp2	0.9744	83.77	1.688	0.87	0.018
Exp3	0.9858	83.89	0.666	2.21	0.086
			EULER		
Exp1	0.9695	83.89	1.3550	1.08	**0.176**
Exp2	0.9816	83.77	0.7874	1.19	0.161
Exp3	0.9881	83.89	0.6758	2.18	0.138
			TAO		
Exp1	**0.9920**	83.89	**0.0853**	**14.81**	0.143
Exp2	**0.9947**	83.77	**0.0621**	**19.29**	**0.264**
Exp3	**0.9918**	83.89	**0.1202**	**10.50**	**0.168**

Evaluation: results

Table 4: Inductive experiment results on LANL.

Exp	AUC	TPR (%)	FPR (%)	Precision (%)	AP
			GraphSAGE		
Exp1	0.9504	83.89	2.281	0.64	0.035
Exp2	0.9744	83.77	1.688	0.87	0.018
Exp3	0.9858	83.89	0.666	2.21	0.086
			EULER		
Exp1	0.9695	83.89	1.3550	1.08	**0.176**
Exp2	0.9816	83.77	0.7874	1.19	0.161
Exp3	0.9881	83.89	0.6758	2.18	0.138
			TAO		
Exp1	**0.9920**	83.89	**0.0853**	**14.81**	0.143
Exp2	**0.9947**	83.77	**0.0621**	**19.29**	**0.264**
Exp3	**0.9918**	83.89	**0.1202**	**10.50**	**0.168**

Evaluation: results

Table 4: Inductive experiment results on LANL.

Exp	AUC	TPR (%)	FPR (%)	Precision (%)	AP
			GraphSAGE		
Exp1	0.9504	83.89	2.281	0.64	0.035
Exp2	0.9744	83.77	1.688	0.87	0.018
Exp3	0.9858	83.89	0.666	2.21	0.086
			EULER		
Exp1	0.9695	83.89	1.3550	1.08	**0.176**
Exp2	0.9816	83.77	0.7874	1.19	0.161
Exp3	0.9881	83.89	0.6758	2.18	0.138
			TAO		
Exp1	**0.9920**	83.89	**0.0853**	**14.81**	0.143
Exp2	**0.9947**	83.77	**0.0621**	**19.29**	**0.264**
Exp3	**0.9918**	83.89	**0.1202**	**10.50**	**0.168**

Table 5: Inductive experiment results on OpTC.

Exp	AUC	TPR (%)	FPR (%)	Precision (%)	AP
			GraphSAGE		
Exp1	0.5354	83.05	68.21	0.0002	2e−6
Exp2	0.6442	83.05	37.31	0.0004	2e−6
Exp3	0.8909	83.05	15.02	0.001	1e−5
			EULER		
Exp1	0.8387	83.05	16.28	0.0009	5e−6
Exp2	0.7262	83.05	33.86	0.0005	3e−6
Exp3	0.7576	83.05	30.01	0.0005	3e−6
			TAO		
Exp1	**0.9998**	**86.44**	**0.0074**	**2.08**	**0.249**
Exp2	**0.9999**	83.05	**0.0072**	**2.05**	**0.271**
Exp3	**0.9999**	83.05	**0.0276**	**0.54**	**0.268**

Evaluation: results

Table 4: Inductive experiment results on LANL.

Exp	AUC	TPR (%)	FPR (%)	Precision (%)	AP
			GraphSAGE		
Exp1	0.9504	83.89	2.281	0.64	0.035
Exp2	0.9744	83.77	1.688	0.87	0.018
Exp3	0.9858	83.89	0.666	2.21	0.086
			EULER		
Exp1	0.9695	83.89	1.3550	1.08	**0.176**
Exp2	0.9816	83.77	0.7874	1.19	0.161
Exp3	0.9881	83.89	0.6758	2.18	0.138
			TAO		
Exp1	**0.9920**	83.89	**0.0853**	**14.81**	0.143
Exp2	**0.9947**	83.77	**0.0621**	**19.29**	**0.264**
Exp3	**0.9918**	83.89	**0.1202**	**10.50**	**0.168**

Table 5: Inductive experiment results on OpTC.

Exp	AUC	TPR (%)	FPR (%)	Precision (%)	AP
			GraphSAGE		
Exp1	0.5354	83.05	68.21	0.0002	2e−6
Exp2	0.6442	83.05	37.31	0.0004	2e−6
Exp3	0.8909	83.05	15.02	0.001	1e−5
			EULER		
Exp1	0.8387	83.05	16.28	0.0009	5e−6
Exp2	0.7262	83.05	33.86	0.0005	3e−6
Exp3	0.7576	83.05	30.01	0.0005	3e−6
			TAO		
Exp1	**0.9998**	**86.44**	**0.0074**	**2.08**	**0.249**
Exp2	**0.9999**	83.05	**0.0072**	**2.05**	**0.271**
Exp3	**0.9999**	83.05	**0.0276**	**0.54**	**0.268**

Evaluation: results

Table 4: Inductive experiment results on LANL.

Exp	AUC	TPR (%)	FPR (%)	Precision (%)	AP
			GraphSAGE		
Exp1	0.9504	83.89	2.281	0.64	0.035
Exp2	0.9744	83.77	1.688	0.87	0.018
Exp3	0.9858	83.89	0.666	2.21	0.086
			EULER		
Exp1	0.9695	83.89	1.3550	1.08	**0.176**
Exp2	0.9816	83.77	0.7874	1.19	0.161
Exp3	0.9881	83.89	0.6758	2.18	0.138
			TAO		
Exp1	**0.9920**	83.89	**0.0853**	**14.81**	0.143
Exp2	**0.9947**	83.77	**0.0621**	**19.29**	**0.264**
Exp3	**0.9918**	83.89	**0.1202**	**10.50**	**0.168**

Table 5: Inductive experiment results on OpTC.

Exp	AUC	TPR (%)	FPR (%)	Precision (%)	AP
			GraphSAGE		
Exp1	0.5354	83.05	68.21	0.0002	2e−6
Exp2	0.6442	83.05	37.31	0.0004	2e−6
Exp3	0.8909	83.05	15.02	0.001	1e−5
			EULER		
Exp1	0.8387	83.05	16.28	0.0009	5e−6
Exp2	0.7262	83.05	33.86	0.0005	3e−6
Exp3	0.7576	83.05	30.01	0.0005	3e−6
			TAO		
Exp1	**0.9998**	**86.44**	**0.0074**	**2.08**	**0.249**
Exp2	**0.9999**	83.05	**0.0072**	**2.05**	**0.271**
Exp3	**0.9999**	83.05	**0.0276**	**0.54**	**0.268**

Evaluation: results

Table 4: Inductive experiment results on LANL.

Exp	AUC	TPR (%)	FPR (%)	Precision (%)	AP
			GraphSAGE		
Exp1	0.9504	83.89	2.281	0.64	0.035
Exp2	0.9744	83.77	1.688	0.87	0.018
Exp3	0.9858	83.89	0.666	2.21	0.086
			EULER		
Exp1	0.9695	83.89	1.3550	1.08	**0.176**
Exp2	0.9816	83.77	0.7874	1.19	0.161
Exp3	0.9881	83.89	0.6758	2.18	0.138
			TAO		
Exp1	**0.9920**	83.89	**0.0853**	**14.81**	0.143
Exp2	**0.9947**	83.77	**0.0621**	**19.29**	**0.264**
Exp3	**0.9918**	83.89	**0.1202**	**10.50**	**0.168**

Table 5: Inductive experiment results on OpTC.

Exp	AUC	TPR (%)	FPR (%)	Precision (%)	AP
			GraphSAGE		
Exp1	0.5354	83.05	68.21	0.0002	2e−6
Exp2	0.6442	83.05	37.31	0.0004	2e−6
Exp3	0.8909	83.05	15.02	0.001	1e−5
			EULER		
Exp1	0.8387	83.05	16.28	0.0009	5e−6
Exp2	0.7262	83.05	33.86	0.0005	3e−6
Exp3	0.7576	83.05	30.01	0.0005	3e−6
			TAO		
Exp1	**0.9998**	**86.44**	**0.0074**	**2.08**	**0.249**
Exp2	**0.9999**	83.05	**0.0072**	**2.05**	**0.271**
Exp3	**0.9999**	83.05	**0.0276**	**0.54**	**0.268**

Evaluation: results

- To reduce false positives, we tried the **node detection** task
- We also r**educed the recall upper bound**

Table 7: Experimental results of host detection with low FPR

	Edge Detection			Host Detection		
Exp	TPR (%)	FPR (%)	Precision (%)	TPR (%)	FPR (%)	Precision (%)
			OpTC			
Exp1	42.4	0.0003	18.2	66.6	0.0000	100.0
Exp2	42.4	0.0003	18.2	66.6	0.0000	100.0
Exp3	42.4	0.0003	18.9	66.6	0.0008	66.7
			LANL			
Exp1	56.5	0.040	19.5	74.6	0.017	6.1
Exp2	42.2	0.030	19.7	67.8	0.012	7.3
Exp3	46.4	0.051	14.2	60.7	0.024	3.8

Evaluation: results

- To reduce false positives, we tried the **node detection** task
- We also r**educed the recall upper bound**

Table 7: Experimental results of host detection with low FPR

	Edge Detection			Host Detection		
Exp	TPR (%)	FPR (%)	Precision (%)	TPR (%)	FPR (%)	Precision (%)
			OpTC			
Exp1	42.4	0.0003	18.2	66.6	0.0000	100.0
Exp2	42.4	0.0003	18.2	66.6	0.0000	100.0
Exp3	42.4	0.0003	18.9	66.6	0.0008	66.7
			LANL			
Exp1	56.5	0.040	19.5	74.6	0.017	6.1
Exp2	42.2	0.030	19.7	67.8	0.012	7.3
Exp3	46.4	0.051	14.2	60.7	0.024	3.8

Evaluation: results

- To reduce false positives, we tried the **node detection** task
- We also r**educed the recall upper bound**

Table 7: Experimental results of host detection with low FPR

	Edge Detection			Host Detection		
Exp	TPR (%)	FPR (%)	Precision (%)	TPR (%)	FPR (%)	Precision (%)
			OpTC			
Exp1	42.4	0.0003	18.2	66.6	0.0000	100.0
Exp2	42.4	0.0003	18.2	66.6	0.0000	100.0
Exp3	42.4	0.0003	18.9	66.6	0.0008	66.7
			LANL			
Exp1	56.5	0.040	19.5	74.6	0.017	6.1
Exp2	42.2	0.030	19.7	67.8	0.012	7.3
Exp3	46.4	0.051	14.2	60.7	0.024	3.8

Evaluation: ablation study

- On LANL, **Temporal Snapshot Shifting** successfully **improves precision** on inductive tasks
- However, on OpTC it doesn't

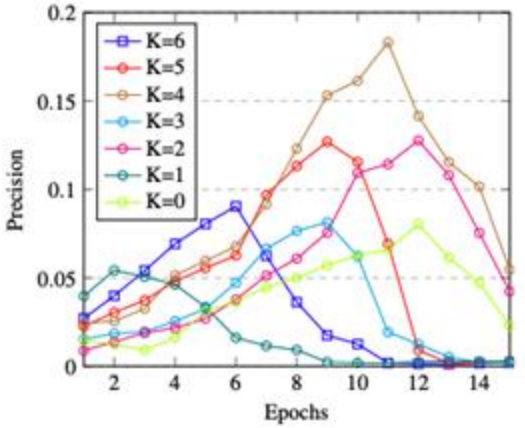


Figure 4: Precision of TAO on LANL (Exp2), with respect to K, the number of preprocessing datasets generated with temporal snapshot shifting.

Evaluation: ablation study (LANL)

Table 13: Ablation Study of TAO on LANL.

Encoder	Decoder	Pretraining	RNN	AUC	TPR (%)	FPR (%)	Precision (%)	AP
				Exp1				
×	✓	✓	✓	**0.993**	83.89	0.129	10.35	0.097
✓	×	✓	✓	0.7675	83.89	49.86	0.03	0.001
✓	✓	×	✓	0.9899	83.89	0.124	10.68	0.192
✓	✓	✓	×	0.9212	83.89	0.894	1.63	0.051
✓	✓	✓	✓	0.9920	83.89	**0.085**	**14.81**	**0.143**
				Exp2				
×	✓	✓	✓	0.992	83.77	0.108	12.13	0.203
✓	×	✓	✓	0.7714	83.77	46.93	0.032	0.001
✓	✓	×	✓	0.9943	83.77	0.101	12.83	0.198
✓	✓	✓	×	0.9734	83.77	0.775	1.89	0.032
✓	✓	✓	✓	**0.9947**	83.77	**0.062**	**19.29**	**0.264**
				Exp3				
×	✓	✓	✓	**0.9952**	83.89	0.13	10.24	0.083
✓	×	✓	✓	0.7632	83.89	46.19	0.03	0.001
✓	✓	×	✓	0.9917	83.89	0.319	4.52	0.103
✓	✓	✓	×	0.9738	83.89	0.867	1.71	0.026
✓	✓	✓	✓	0.9918	83.89	**0.120**	**10.50**	**0.168**

Evaluation: ablation study (OPTC)

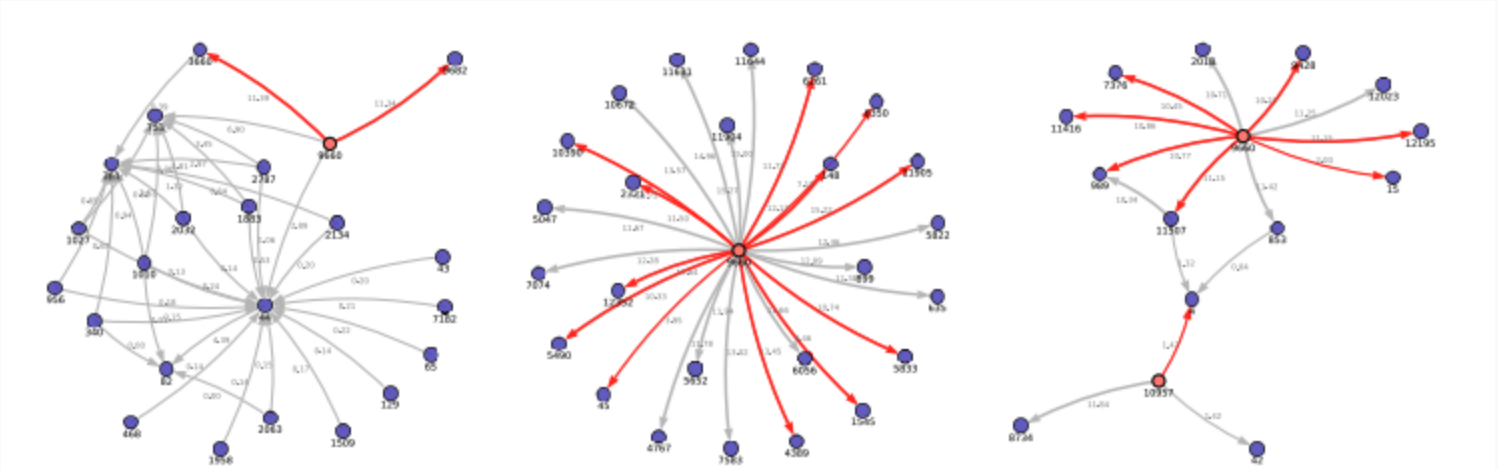
Table 14: Ablation Study of TAO on OpTC.

Encoder	Decoder	Pretraining	RNN	AUC	TPR (%)	FPR (%)	Precision (%)	AP
				Exp1				
×	✓	✓	✓	0.9997	83.05	**0.005**	**2.70**	0.071
✓	×	✓	✓	0.9993	83.05	0.147	0.10	0.306
✓	✓	×	✓	0.9991	83.05	0.156	0.09	0.271
✓	✓	✓	×	**0.9998**	84.75	0.026	0.57	**0.426**
✓	✓	✓	✓	**0.9998**	**86.44**	0.007	2.08	0.249
				Exp2				
×	✓	✓	✓	0.9987	83.05	0.220	0.07	0.123
✓	×	✓	✓	0.9994	83.05	0.042	0.01	**0.306**
✓	✓	×	✓	0.9995	83.05	0.022	0.70	0.247
✓	✓	✓	×	0.9993	83.05	0.015	0.96	0.108
✓	✓	✓	✓	**0.9999**	83.05	**0.007**	**2.05**	0.271
				Exp3				
×	✓	✓	✓	0.9942	83.05	1.087	0.01	0.060
✓	×	✓	✓	0.9992	83.05	0.192	0.01	0.261
✓	✓	×	✓	0.9991	83.05	0.156	0.01	**0.272**
✓	✓	✓	×	0.9997	**84.75**	**0.020**	**0.77**	0.108
✓	✓	✓	✓	**0.9999**	83.05	0.028	0.54	0.268

Future work

- Substitute the one-hot encoded node features by actual inductive features
- Find a better thresholding method than the AUC-ROC + upper bound recall
- Deeper analysis on how reducing the FPR

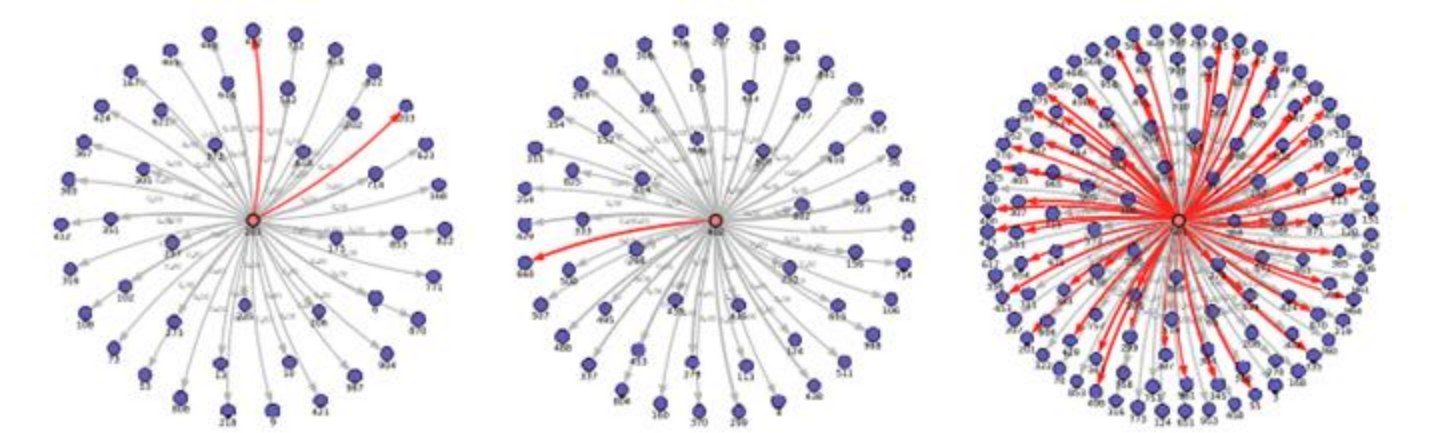
Bonus: nice explainability graphs



(a) Test Snapshot 1 (First Malicious Event) (b) Test Snapshot 164 (many FPs) (c) Test Snapshot 262 (missing a TP)

Figure 6: Examples of explainability graphs generated on LANL.

Bonus: nice explainability graphs



(a) Test Snapshot 7 (First Malicious Event) (b) Test Snapshot 7 (Second Malicious Event) (c) Test Snapshot 51 (All TPs detected, 0 FP)

Figure 7: Examples of explainability graphs generated on OpTC. Due to the large degree of nodes in OpTC, only 15% of benign edges are kept for the clarity of graph visualizations.

The End

Any questions?