ProvDP: Differential Privacy for System Provenance Dataset

Kunal Mukherjee[1], Jonathan Yu, Partha De[1], and

Dinil Mon Divakaran[2]

[1]Department of Computer Science, The University of Texas at Dallas
[2]Institute for Infocomm Research, A*STAR, Singapore

ACNS 2025

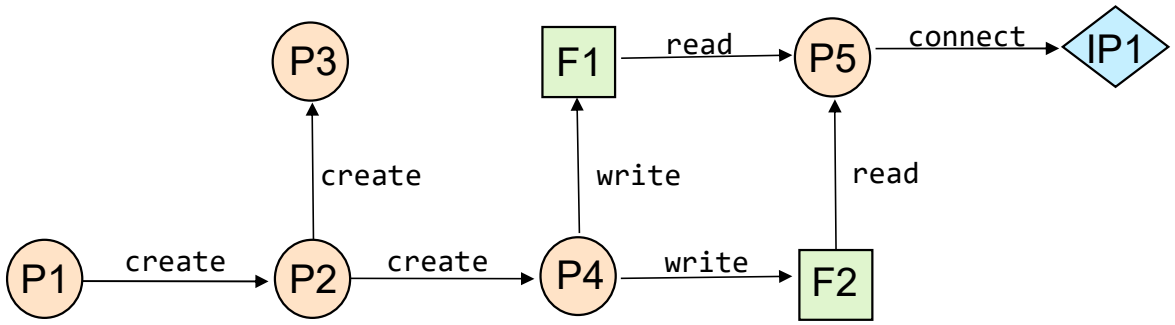
Agenda

- **Background**
- Differential Privacy for Graphs
- ProvDP: Graph to Tree Conversion
- ProvDP: Pruning Algorithm
- ProvDP: Grafting Algorithm
- Evaluation
- Discussion

Background: Dynamic Defense against Stealthy Attacks

- **System Provenance** championed as a *host-based* dynamic defense
  - Influential works [Hassan '19, Wang '20, Han '21]

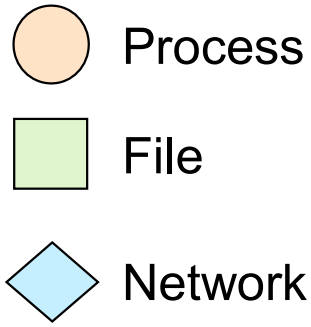
- System Provenance *causally* connects system resources
  - Captures *dynamic* control and data dependencies



How can system Provenance help detect stealthy attacks?

Background: Provenance Graph Schema

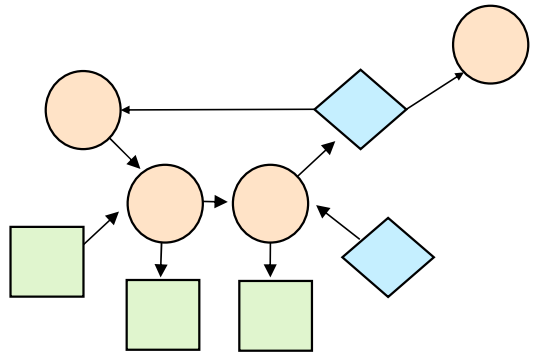
Nodes



Example metadata:
- process: pid, cmd
- file: path, permissions
- network: ip/port

Edges

From \ To	Process	File	Network
Process	Create Kill	Write	Write
File	Read	Illegal	Illegal
Network	Read	Illegal	Illegal



Example metadata:
- timestamp
- file/network: bytes written/read
- process: cmd

Agenda

- Background
- **Differential Privacy**
- ProvDP: Graph to Tree Conversion
- ProvDP: Pruning Algorithm
- ProvDP: Grafting Algorithm
- Evaluation
- Discussion

Differential Privacy

- Differential Privacy (DP): Mathematical definition for privacy
- Assures users that, within some budget ϵ, we cannot infer their data is part of our dataset.
- Lower ϵ means more private, so we should add more noise
- DP algorithms are composable

$$\frac{Pr[M(D) \in S]}{Pr[M(D') \in S]} \leq e^{\epsilon}$$

- M is a privacy-preserving mechanism
- D, D' are "neighboring" datasets
- S ⊆ Range(M)
- ϵ is the "privacy budget"

Differential Privacy for Graphs

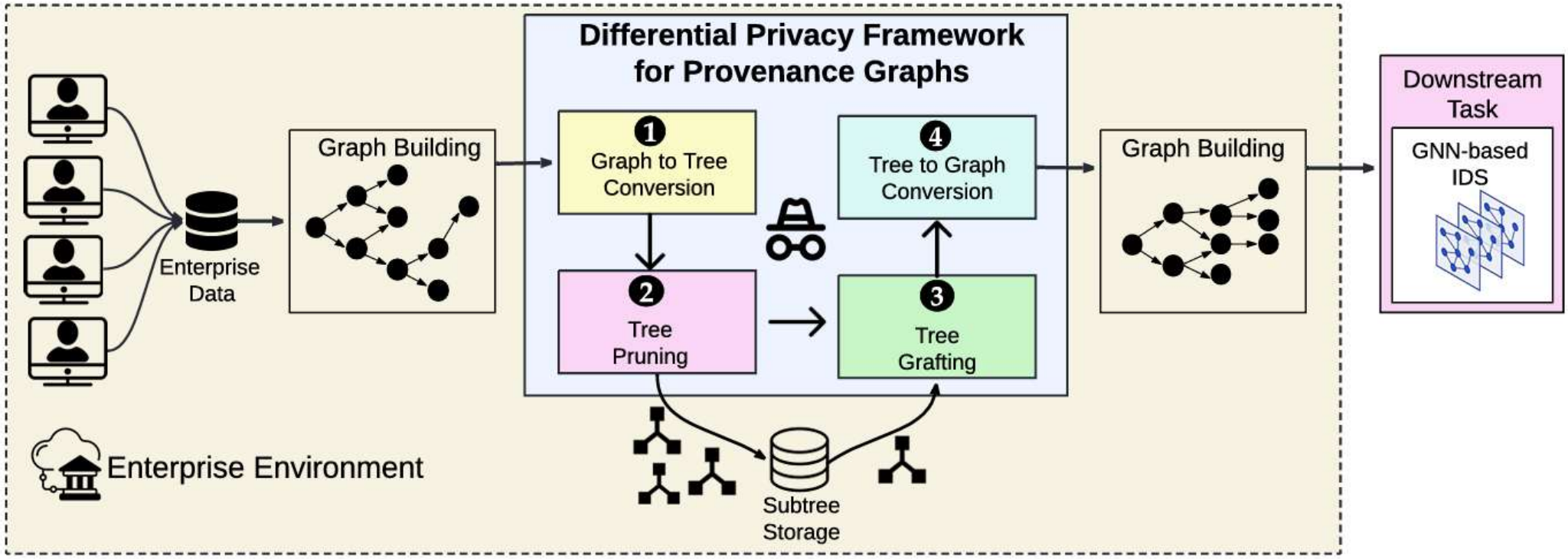
- Dependent of the definition of “neighbors” (D and D’)
- Edge-level DP: D and D’ differ in a single edge
- Node-level DP: D and D’ differ in a single node (and its edges)
  - Stronger guarantee than edge dp
- *Subtree-level DP*: D and D’ differ in a **subtree**
  - Stronger guarantee than node and edge DP

$$\frac{Pr[M(D) \in S]}{Pr[M(D') \in S]} \leq e^{\epsilon}$$

Agenda

- Background
- Differential Privacy
- **ProvDP: Differential Privacy Framework**
- ProvDP: Graph to Tree Conversion
- ProvDP: Pruning Algorithm
- ProvDP: Grafting Algorithm
- Evaluation
- Discussion

ProvDP: Differential Privacy Framework



Privacy Budget Allocation

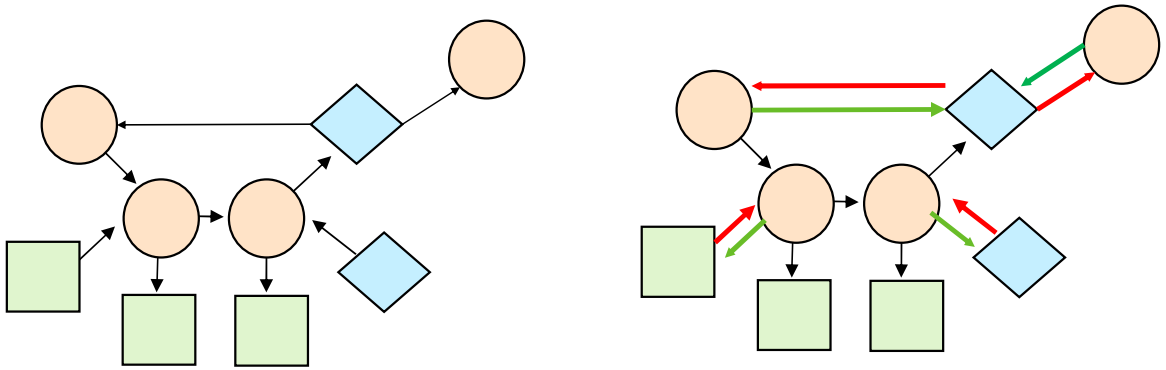
- Pruning and Grafting are separate differentially private mechanisms
- $\epsilon = \epsilon_1 + \epsilon_2 =$ total privacy budget
- $\delta \in [0, 1]$ controls allocation of budget
- Pruning $\epsilon_1 = \delta\epsilon$
- Grafting $\epsilon_2 = (1 - \delta)\epsilon$

Agenda

- Background
- Differential Privacy
- ProvDP: Differential Privacy Framework
- **ProvDP: Graph to Tree Conversion**
- ProvDP: Pruning Algorithm
- ProvDP: Grafting Algorithm
- Evaluation
- Discussion

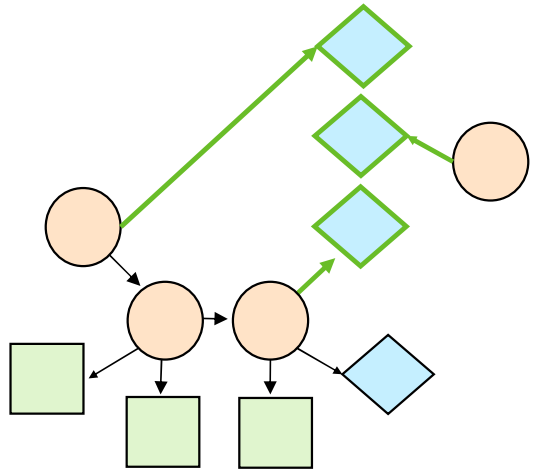
ProvDP: Graph to Tree Conversion

1. Break cycles: Invert outgoing edges from file/network nodes
   - Edge can be restored from metadata
   - Graph is now acyclic



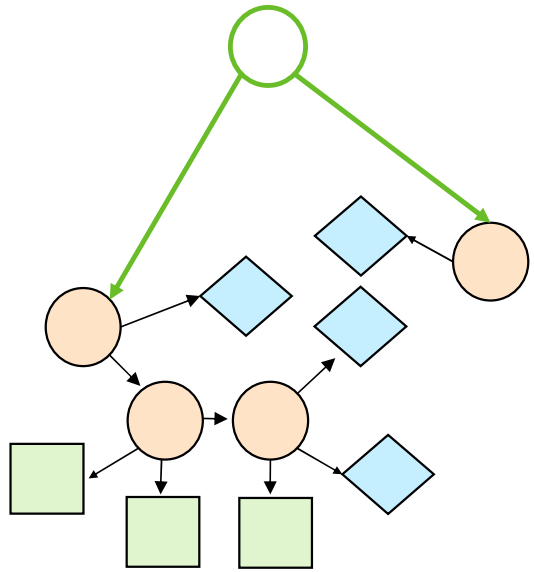
ProvDP: Graph to Tree Conversion

1. Break cycles:
   - Invert outgoing edges from file/network nodes
   - Edge can be restored from metadata
   - Graph is now acyclic
2. Remove lattice structure:
   - Duplicate file/network nodes for each edge
   - Can be restored from file path / IP address / port
   - Removes lattice structure



ProvDP: Graph to Tree Conversion

1. Break cycles:
   - Invert outgoing edges from file/network nodes
   - Edge can be restored from metadata
   - Graph is now acyclic
2. Remove lattice structure:
   - Duplicate file/network nodes for each edge
   - Can be restored from file path / IP address / port
   - Removes lattice structure
   - Graph is now a forest
3. Forest to tree:
   - Connect all process roots to a virtual root node
   - Graph is now a tree



Agenda

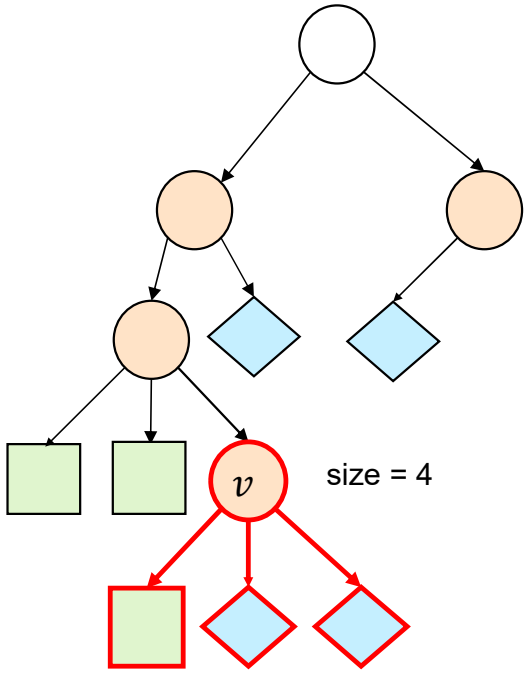
- Background
- Differential Privacy
- ProvDP: Differential Privacy Framework
- ProvDP: Graph to Tree Conversion
- **ProvDP: Pruning Algorithm**
- ProvDP: Grafting Algorithm
- Evaluation
- Discussion

ProvDP: Pruning Algorithm

- Run on each graph inside dataset
- Starting at the root node, traverse the graph.
- Randomly prune subtree rooted at node v
  - $S(v)$ is a function of subtree size, out-degree, depth, height
  - Each feature is weighted by $\alpha, \beta, \gamma, \eta$, respectively.

$$P(\text{prune } v) = \frac{1}{1 + e^{\epsilon_1 / 2S(v)}}$$

- For each pruned subtree:
  - Mark v for and store the subtree size s_v
  - Store pruned subtree, along with its parent node and edge



Agenda

- Background
- Differential Privacy
- ProvDP: Differential Privacy Framework
- ProvDP: Graph to Tree Conversion
- ProvDP: Pruning Algorithm
- **ProvDP: Grafting Algorithm**
- Evaluation
- Discussion

ProvDP: Grafting Algorithm

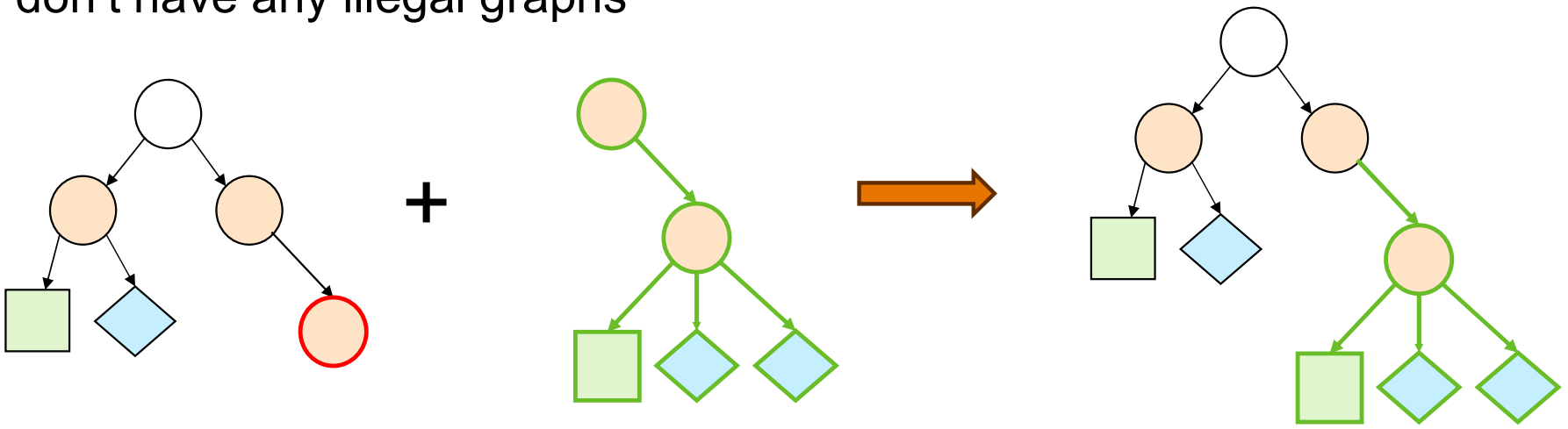
- Run on each graph in the dataset, **after pruning all graphs**
  - After pruning, we have a bucket of all pruned subtrees B
- Randomly replace all marked nodes
  - Perturb original size s_v by adding noise: $\tilde{s}_v = s_v + Lap(\frac{1}{\epsilon_2})$
  - Randomly sample a subtree from B
  - Each subtree $t \in B$ has probability $\mathrm{p}_t = x(1 + |\tilde{s}_v - s_t|)$ of being chosen
    - Normalization factor $x = 1/\sum_{t \in B} p_t$

ProvDP: Grafting Notes

- When pruning, store the subtree, and its parent relation
- Replacing the incoming edge guarantees we don't have any illegal graphs

To

From	Process	File	Network
Process	Create Kill	Write	Write
File	Read	Illegal	Illegal
Network	Read	Illegal	Illegal

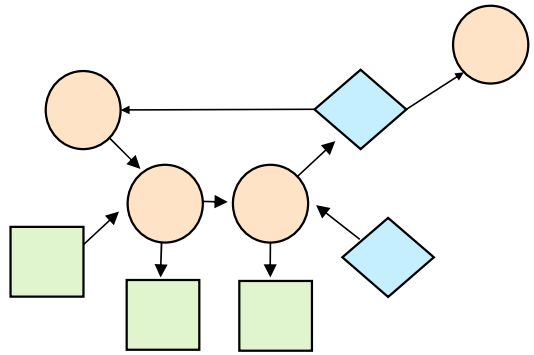


Agenda

- Background
- Differential Privacy
- ProvDP: Differential Privacy Framework
- ProvDP: Graph to Tree Conversion
- ProvDP: Pruning Algorithm
- ProvDP: Grafting Algorithm
- **Evaluation**
- Discussion

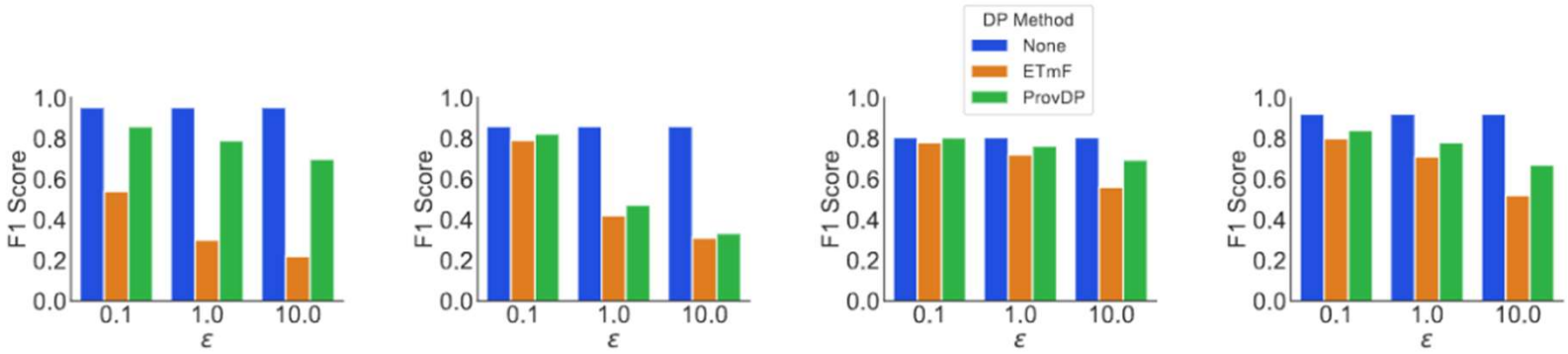
Evaluation: Baseline (Extended Top-m Filter)

- Top-m Filter (TmF) [Nguyen '15]
  - Edge-differentially private
  - Designed for undirected graphs
  - Efficiently adds LaPlace noise to adjacency matrix
- Problem: randomly creating edges can lead to illegal provenance graphs
- Extended Top-m Filter (ETmF)
  - Input: Source and Destination vertices V_s, V_d
  - Perturb adjacency matrix of subgraph only containing nodes in V_s, V_d
  - Run ETmF for all possible legal source, destination pairs



Evaluation: IDS Performance

- ProvDP adds noise more strategically under the same budget



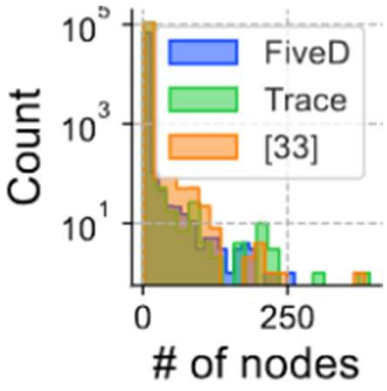
(a) FiveD (b) Trace (c) Theia (d) [33]

Fig. 3: Detection performance of GNN-based IDS using different privacy budgets.

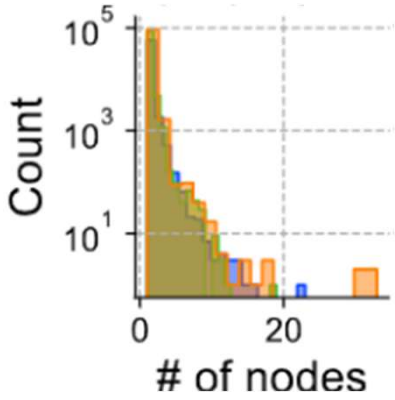
Evaluation: Privacy Budget

Count of subtrees pruned, grouped by subtree size

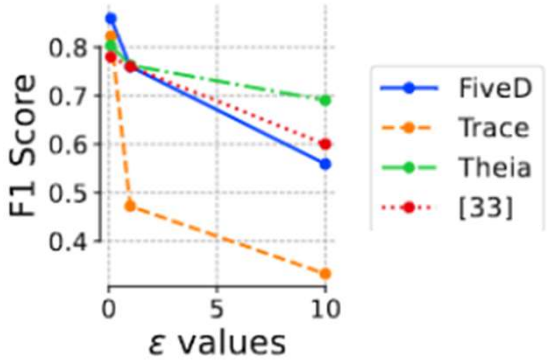
$\epsilon = 0.1$



$\epsilon = 1$



Intrustion Detection performance



Agenda

- Background
- Differential Privacy
- ProvDP: Differential Privacy Framework
- ProvDP: Graph to Tree Conversion
- ProvDP: Pruning Algorithm
- ProvDP: Grafting Algorithm
- Evaluation
- **Discussion and Limitation**

Discussions and Limitations

- Real-world adaptation
- Scalability
- Generalization to alternative IDS models

THANK YOU

KUNAL MUKHERJEE*
kunmukh@gmail.com
www. KUNMUKH.com

*** In Academic Job Market from Fall 2026**

References

Wagner & Soto ’02 - Wagner, David, and Paolo Soto. "Mimicry attacks on host-based intrusion detection systems." *Proceedings of the 9th ACM Conference on Computer and Communications Security*. 2002.

Tan & Maxion ’03 - Tan, Kymie MC, and Roy A. Maxion. "Determining the operational limits of an anomaly-based intrusion detector." *IEEE Journal on selected areas in communications* 21.1 (2003): 96-110.

Velickovic ’17 - Veličković, Petar, et al. "Graph attention networks." *arXiv preprint arXiv:1710.10903* (2017).

Hassan ’19 - Hassan, Wajih Ul, et al. "Nodoze: Combatting threat alert fatigue with automated provenance triage." *network and distributed systems security symposium*. 2019.

Dang ’19 - Dang, Fan, et al. "Understanding fileless attacks on linux-based iot devices with honeycloud." *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*. 2019.

Ying ’19 - Ying, Zhitao, et al. "Gnnexplainer: Generating explanations for graph neural networks." *Advances in neural information processing systems* 32 (2019).

Wang ’20 - Wang, Qi, et al. "You Are What You Do: Hunting Stealthy Malware via Data Provenance Analysis." *NDSS*. 2020.

Han ‘21 - Han, Xueyuan, et al. "{SIGL}: Securing Software Installations Through Deep Graph Learning." *30th USENIX Security Symposium (USENIX Security 21)*. 2021.

Barr-Smith ’21 - Barr-Smith, Frederick, et al. "Survivalism: Systematic analysis of windows malware living-off-the-land." *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2021.

Zeng ‘22 - Zeng, Jun, et al. "Shadewatcher: Recommendation-guided cyber threat analysis using system audit records." *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2022.

Colonial – Easterly, Jen “The Attack on Colonial Pipeline: What We’ve Learned & What We’ve Done over the Past Two Years: CISA.” Cybersecurity and Infrastructure Security Agency CISA, 8 Aug. 2023, www.cisa.gov/news-events/news/attack-colonial-pipeline-what-weve-learned-what-weve-done-over-past-two-years.

SolarWinds - “The Solarwinds Cyber-Attack: What You Need to Know.” *CIS*, 9 Nov. 2021, www.cisecurity.org/solarwinds.

Nguyen, H.H., Imine, A., Rusinowitch, M.: Differentially private publication of social graphs at linear cost. In: Proceedings of the 2015 IEEE/ACM Interna tional Conference on Advances in Social Networks Analysis and Mining 2015. p. 596–599. ASONAM ’15, Association for Computing Machinery, New York, NY, USA(2015).