Increasing the Cost of Model Extraction with Calibrated Proof of Work

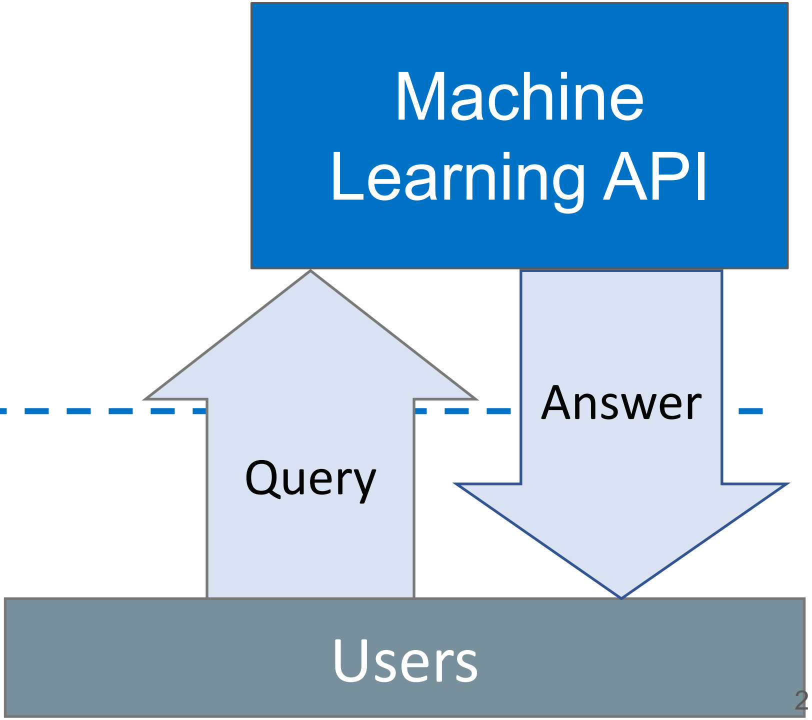
Ahmad Kaleem, Adam Dziedzic, Lucy Lu, Nicolas Papernot

VECTOR INSTITUTE

cleverhans

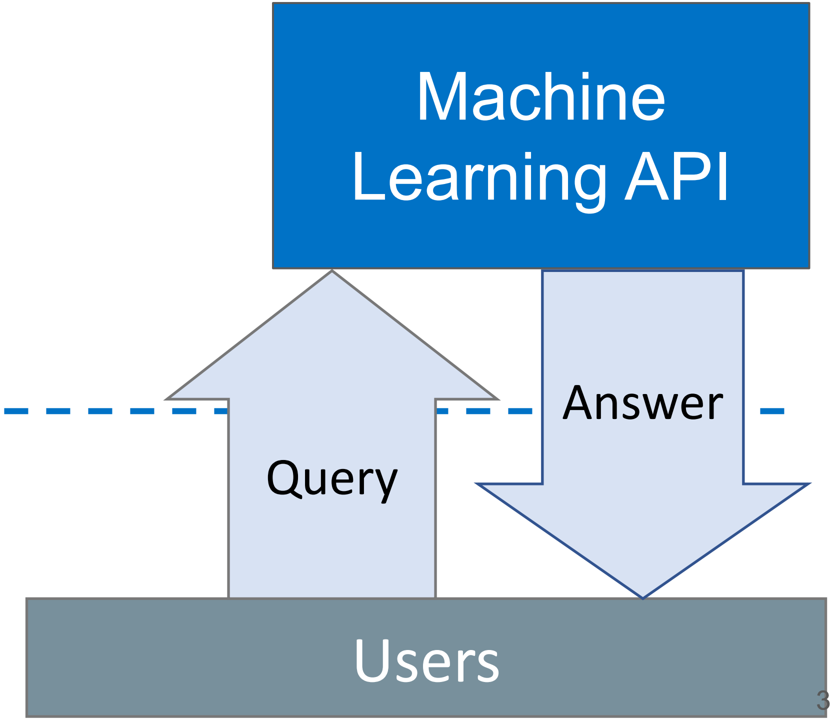
UNIVERSITY OF TORONTO

Annotate data using Machine Learning APIs

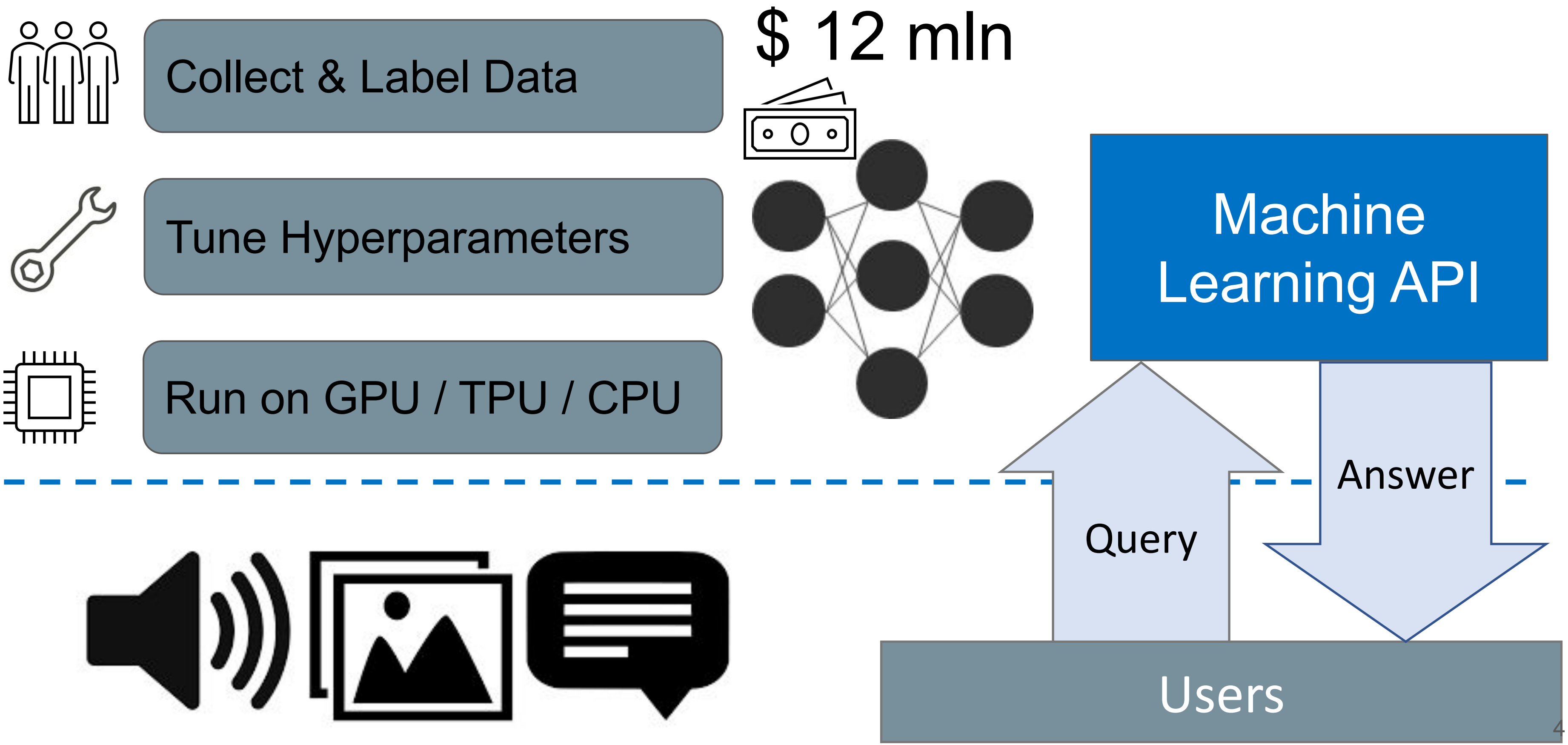


Train models for Machine Learning Services

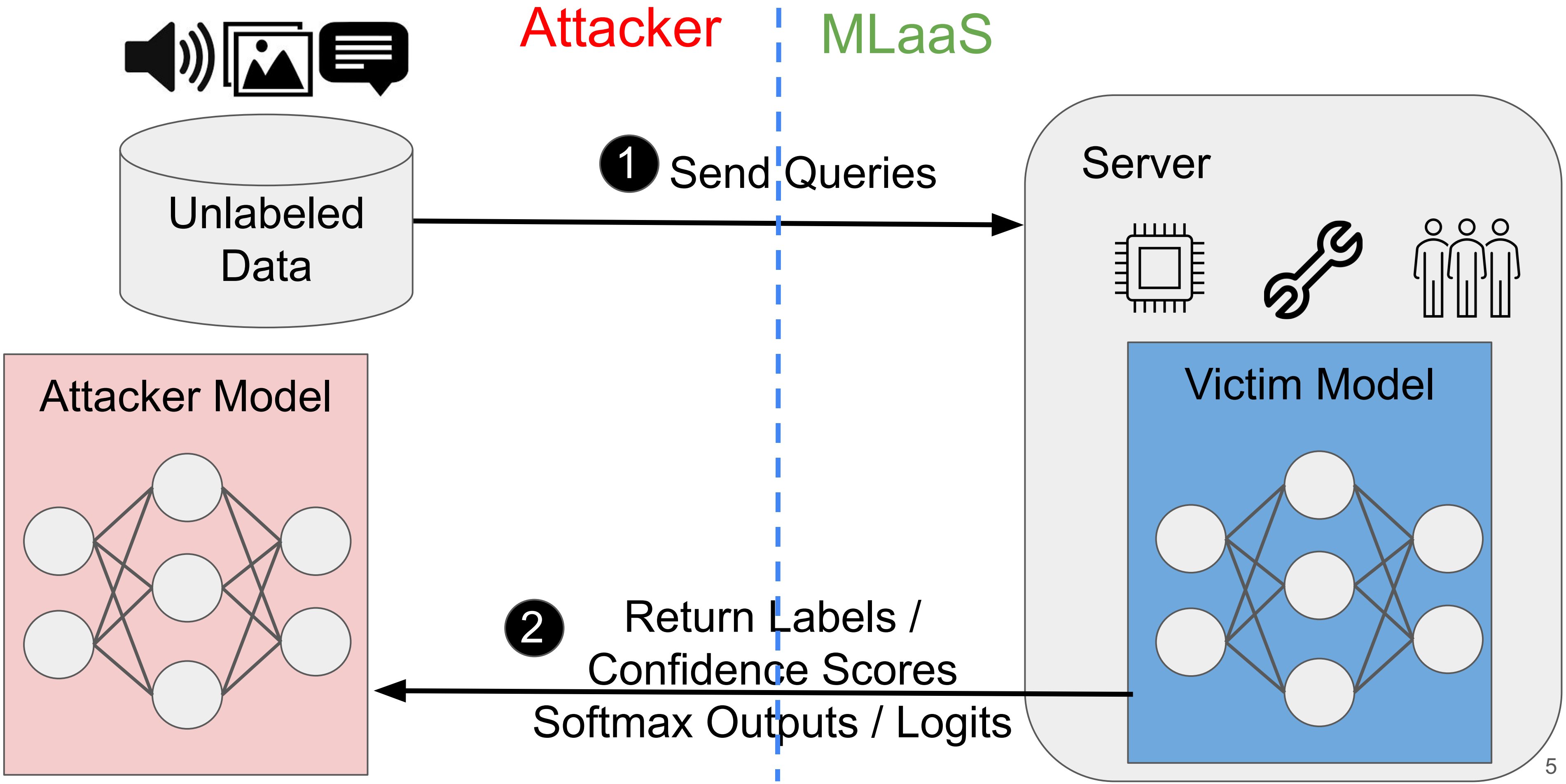
- Collect & Label Data
- Tune Hyperparameters
- Run on GPU / TPU / CPU



Train models for Machine Learning Services

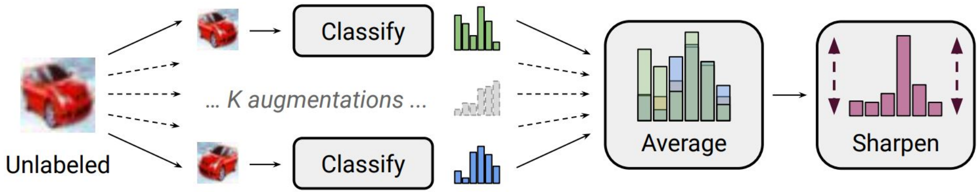


Model Extraction Attacks against MLaaS



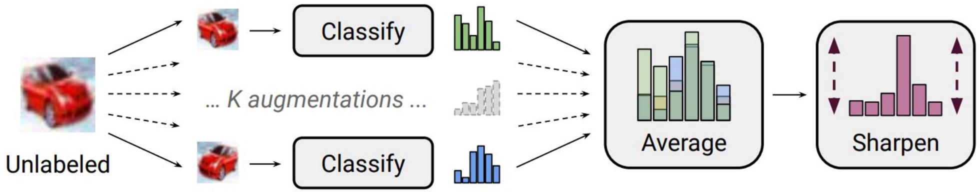
1. Current attacks & defenses
2. Our defense method based on proof-of-work
3. Empirical evaluation
4. Conclusions & Future work

Overview of Model Extraction Attacks

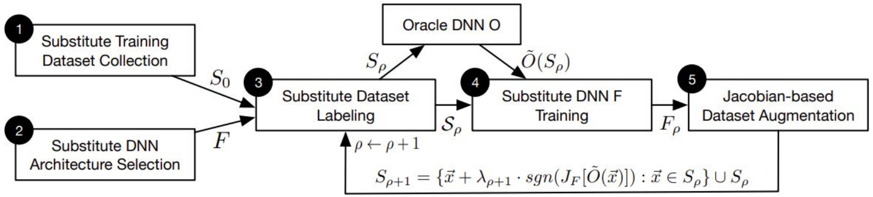


MixMatch Extraction

Overview of Model Extraction Attacks

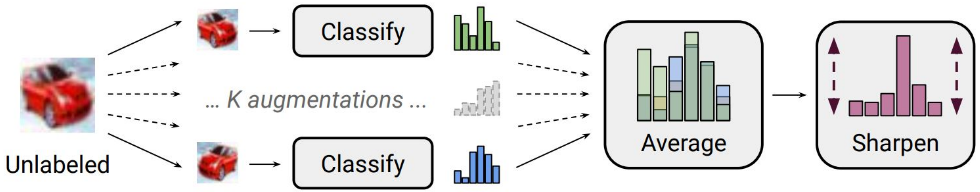


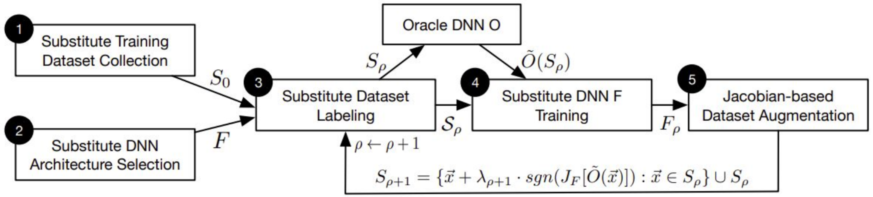
MixMatch Extraction



Jacobian-based Data Augmentation

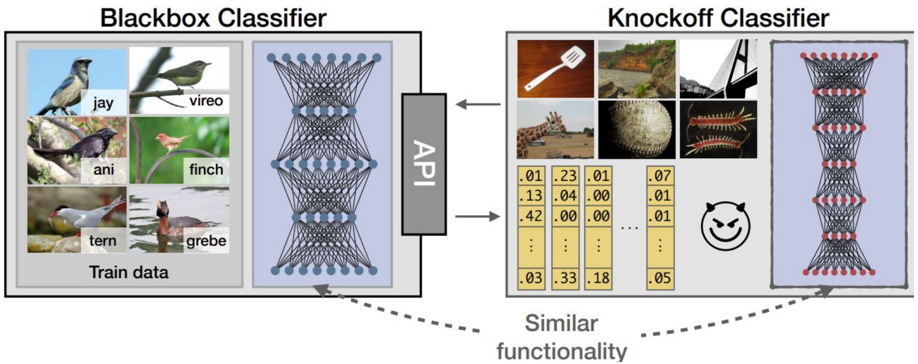
Overview of Model Extraction Attacks





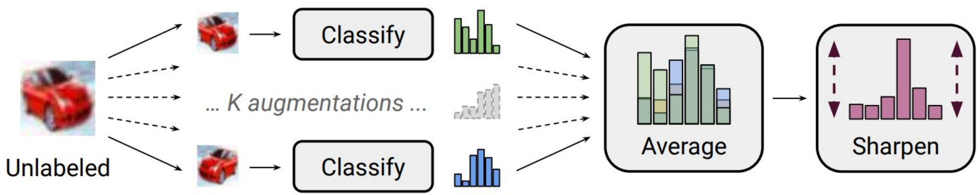
MixMatch Extraction

Jacobian-based Data Augmentation

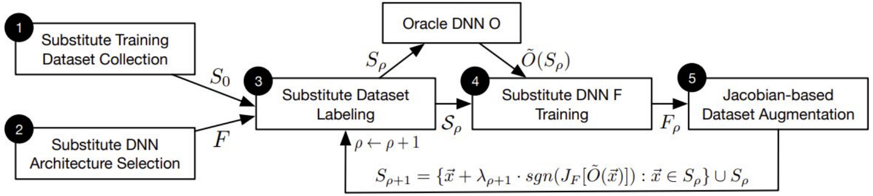


Knockoff Nets

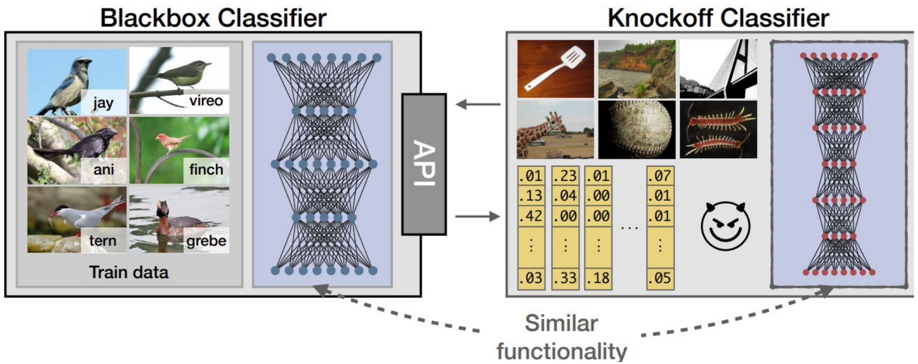
Overview of Model Extraction Attacks



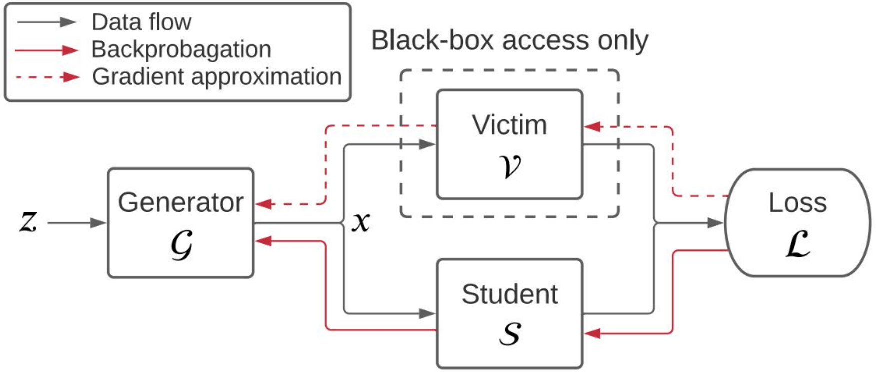
MixMatch Extraction



Jacobian-based Data Augmentation



Knockoff Nets



Data Free Model Extraction

Comparison between Model Extraction **Attacks**

Feature / Attack	Upfront Cost	Query Type	# of Queries CIFAR-10	Goal
MixMatch	High	In-distribution	< 8K	Accuracy
Jacobian	Moderate	Limited In-distribution	80K	Fidelity
Knockoff Nets	Low	Natural (not In-distribution)	50K	Accuracy
Data Free	None	Synthetic	20M	Accuracy

Active Defenses

Perturb outputs

Detect the attack

- Adaptive Misinformation (Kariyappa & Qureshi 2020)
- Prediction Poisoning (Orekondy et al. 2020)
- PRADA (Juuti et al. 2019)

Reactive Defenses

Verify model training

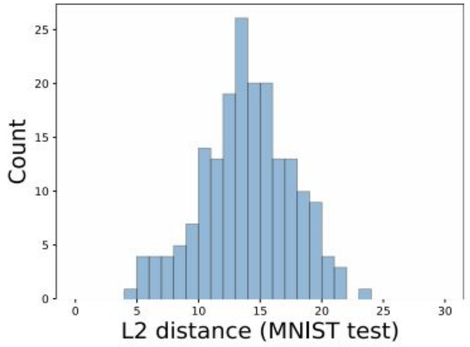
Identify if a trained model was stolen

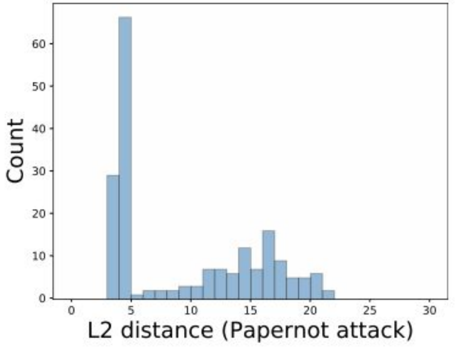
- Watermarking (Jia et al. 2020)
- Dataset Inference (Maini et al. 2021)
- Proof of Learning (Jia et al. 2021)

Examples of **Defenses** against Model Extraction

Active: PRADA

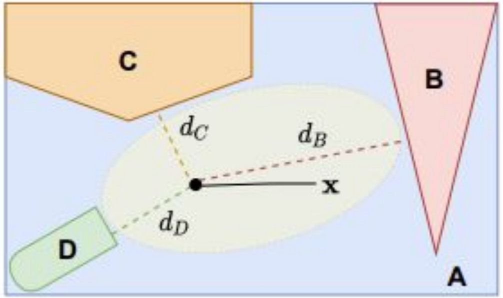
Detect Distribution Shift



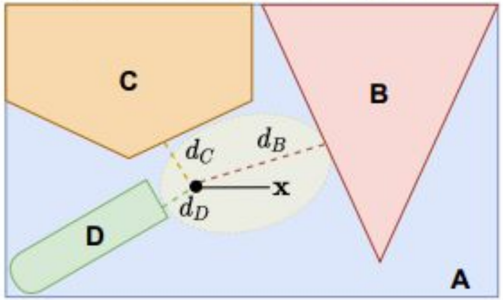


Reactive: Dataset Inference

Resolve Model Ownership

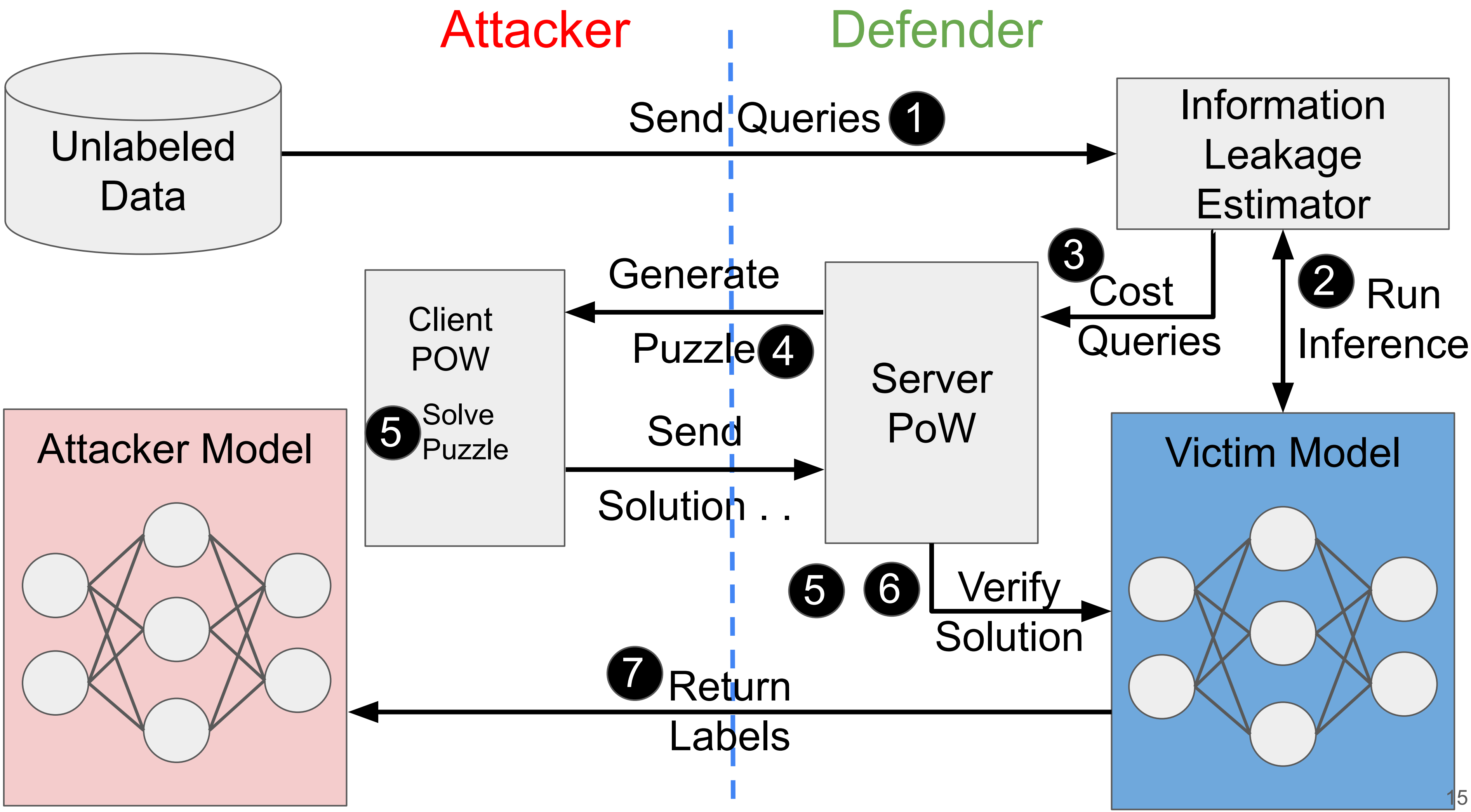


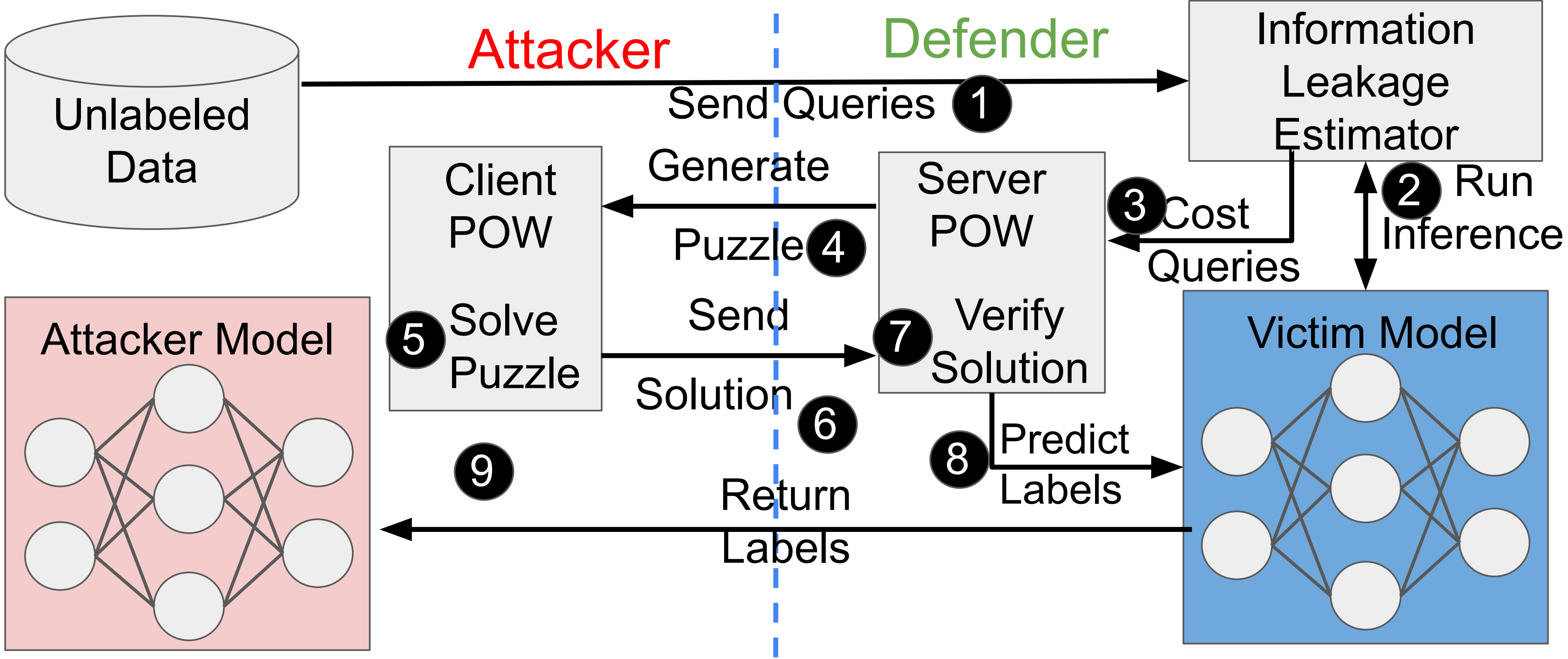
(a) If $\mathbf{x}$ is in training set

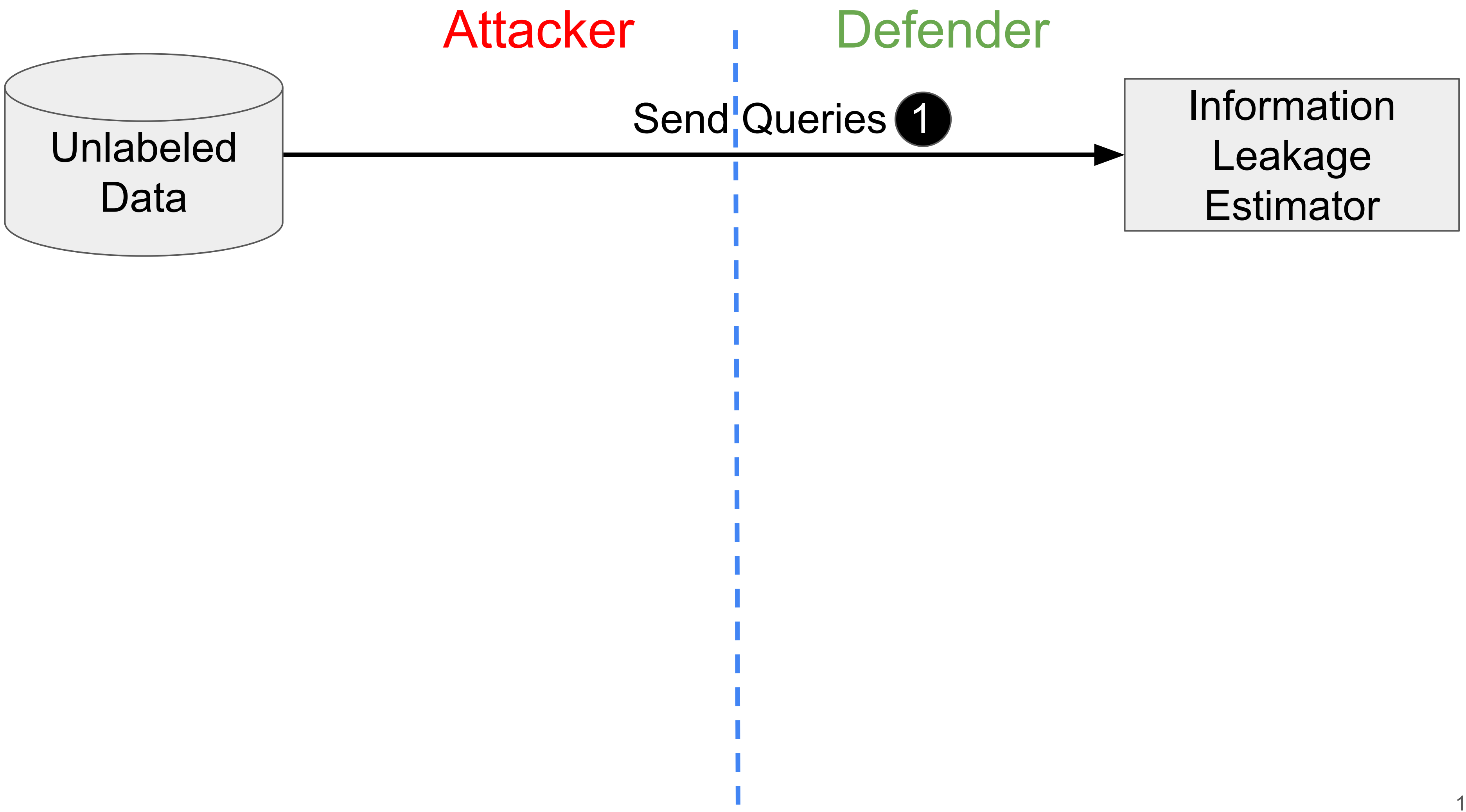


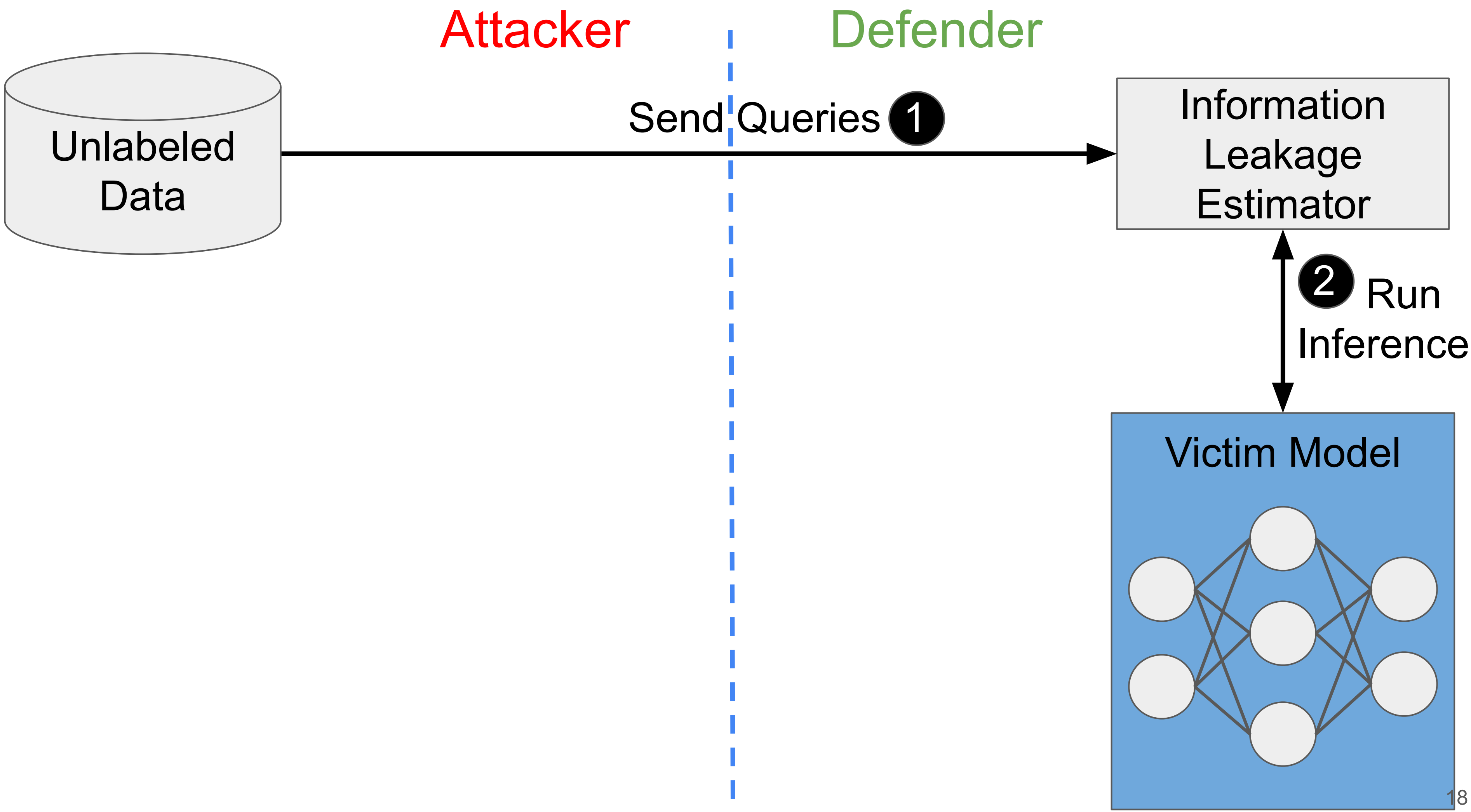
(b) If $\mathbf{x}$ is not in training set

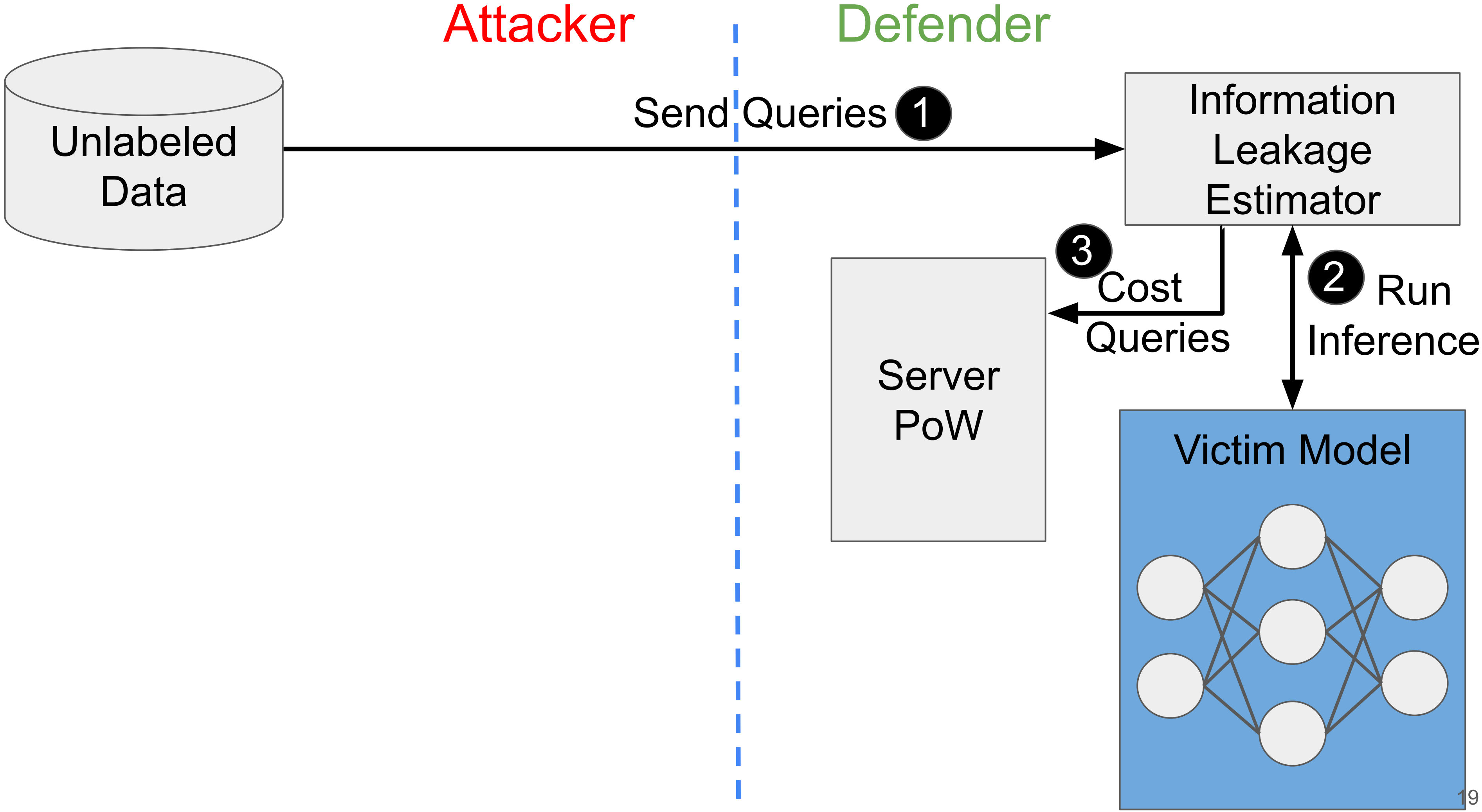
1. Current attacks & defenses
2. **Our defense method based on proof-of-work**
3. Empirical evaluation
4. Conclusions & Future work


Attacker
Defender
Unlabeled Data
Send Queries 1
Information Leakage Estimator
2 Run Inference
3 Cost Queries
Generate Puzzle 4
Client POW
5 Solve Puzzle
Send Solution . .
Server PoW
5 6 Verify Solution
Attacker Model
Victim Model
7 Return Labels

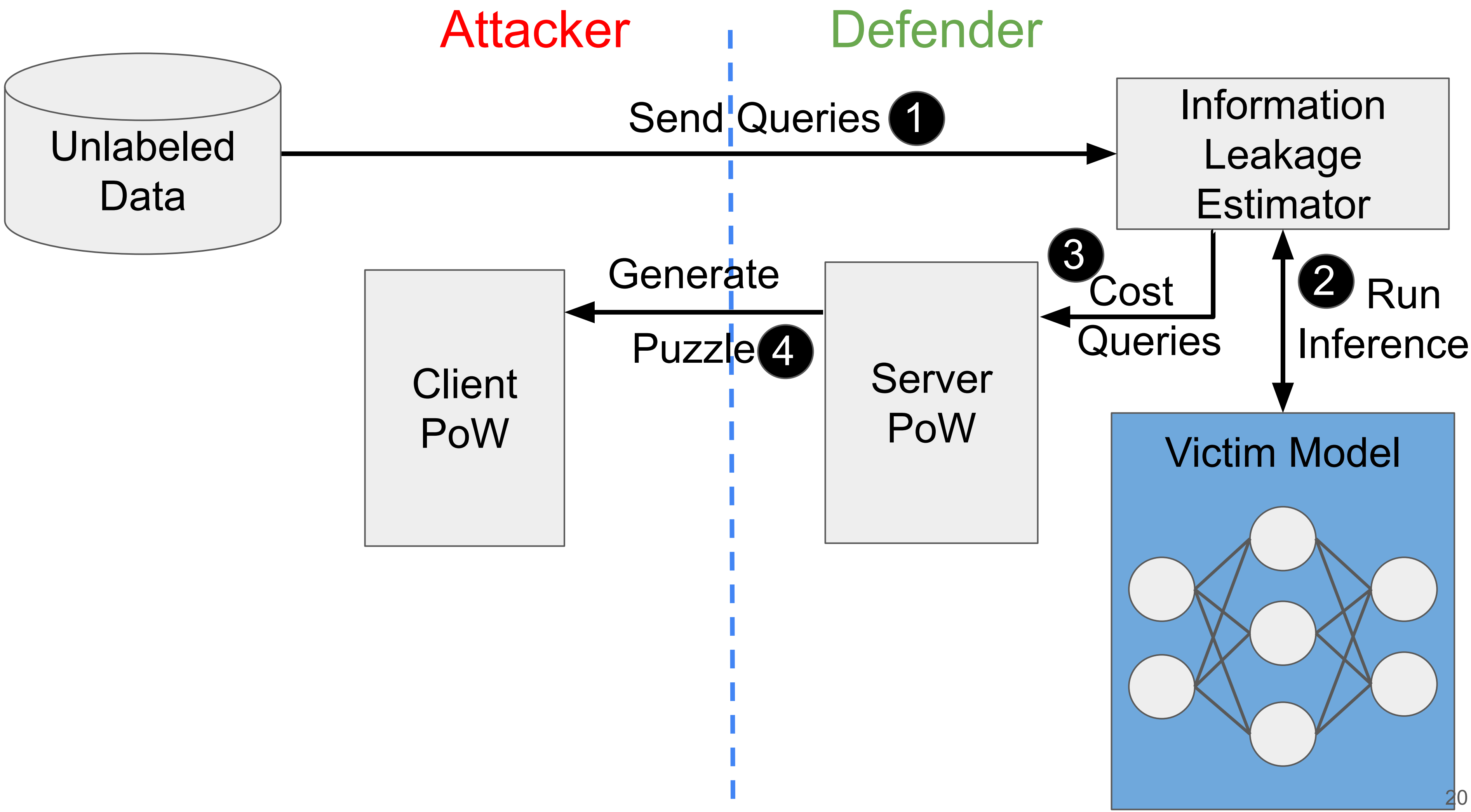

Unlabeled Data
Attacker
Defender
Information Leakage Estimator
Send Queries
1
2
Run Inference
Client POW
Generate Puzzle
4
Server POW
3
Cost Queries
5
Solve Puzzle
Send Solution
6
7
Verify Solution
Attacker Model
Victim Model
8
Predict Labels
9
Return Labels

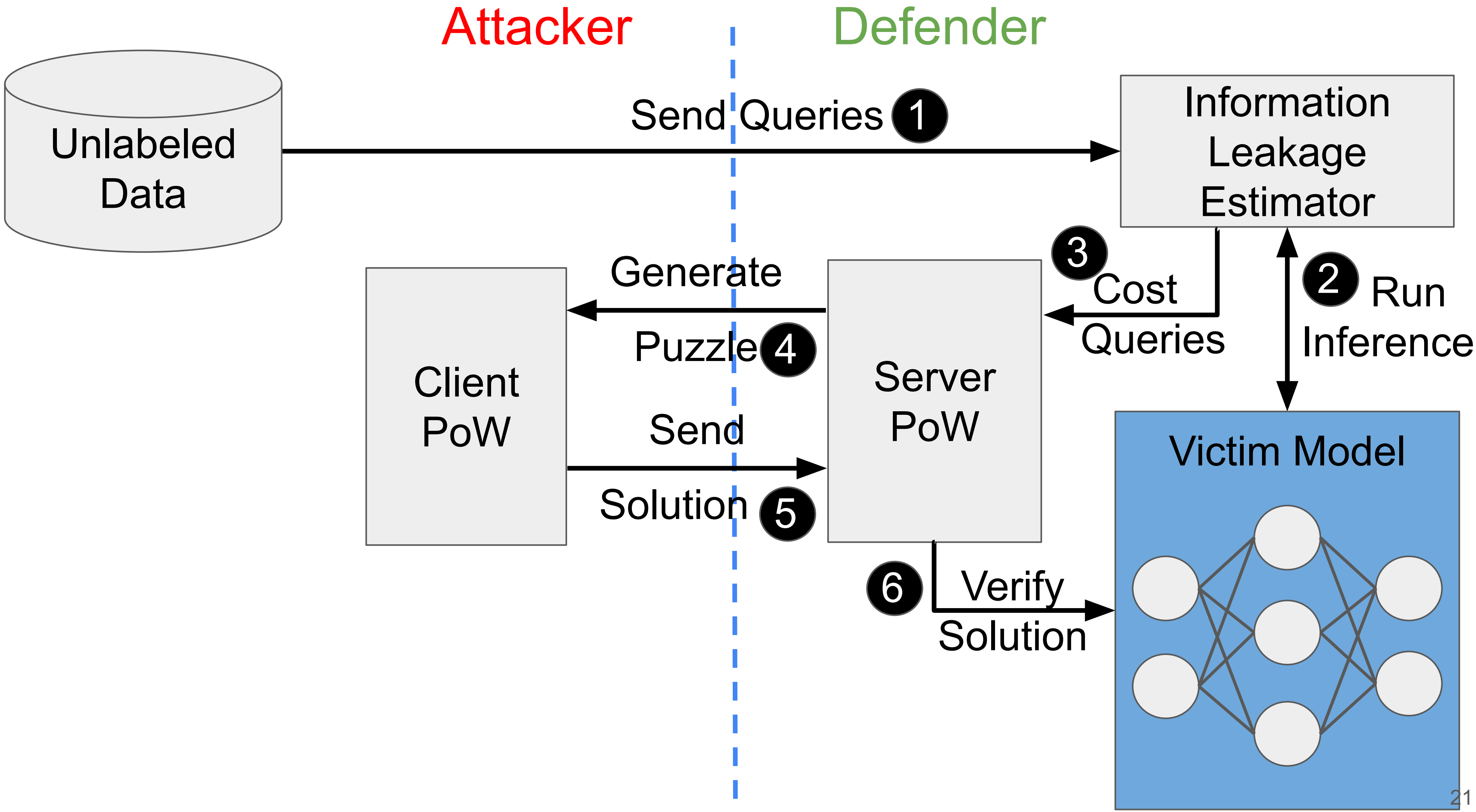

Attacker
Defender
Unlabeled Data
Send Queries 1
Information Leakage Estimator

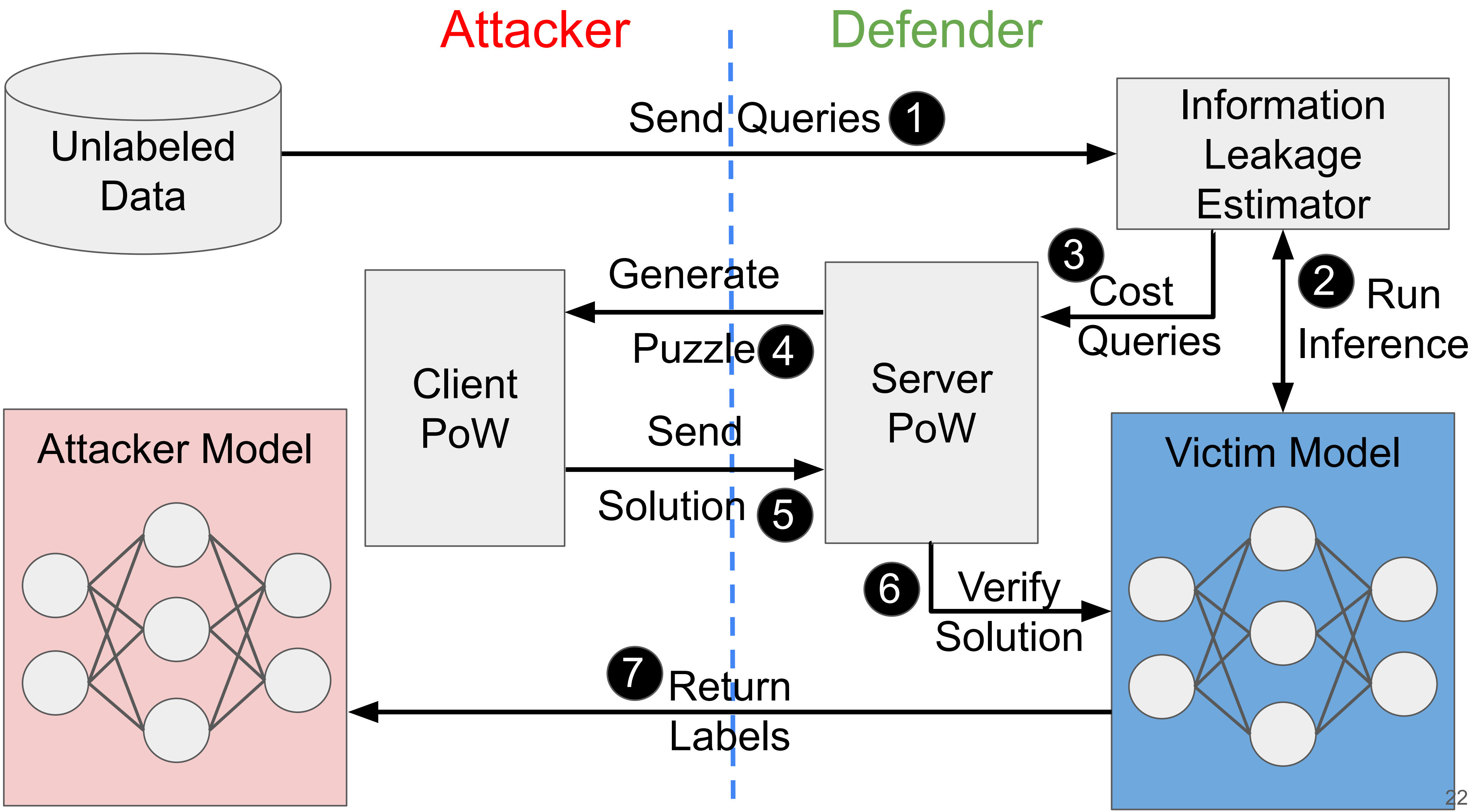

Attacker
Defender
Unlabeled Data
Send Queries 1
Information Leakage Estimator
2 Run Inference
Victim Model



Attacker
Defender
Unlabeled Data
Send Queries
1
Information Leakage Estimator
3
Cost Queries
2
Run Inference
Server PoW
Victim Model


Attacker
Defender
Unlabeled Data
Send Queries 1
Information Leakage Estimator
3 Cost Queries
2 Run Inference
Generate Puzzle 4
Client PoW
Server PoW
Victim Model


Attacker
Defender
Unlabeled Data
Send Queries 1
Information Leakage Estimator
3 Cost Queries
2 Run Inference
Generate Puzzle 4
Client PoW
Server PoW
Send Solution 5
Victim Model
6 Verify Solution

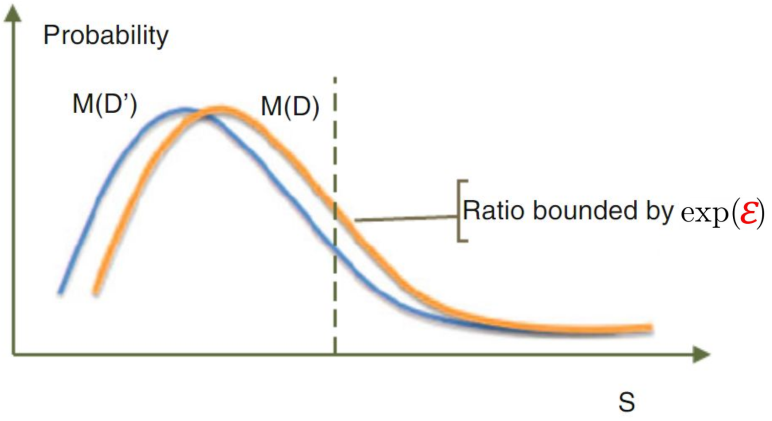

Attacker
Defender
Unlabeled Data
Send Queries 1
Information Leakage Estimator
2 Run Inference
3 Cost Queries
Generate Puzzle 4
Server PoW
Client PoW
Send Solution 5
6 Verify Solution
Attacker Model
Victim Model
7 Return Labels

How to compute the query cost per user?

Entropy: $$-\sum_i P_\theta(y_i|x)\log P_\theta(y_i|x)$$

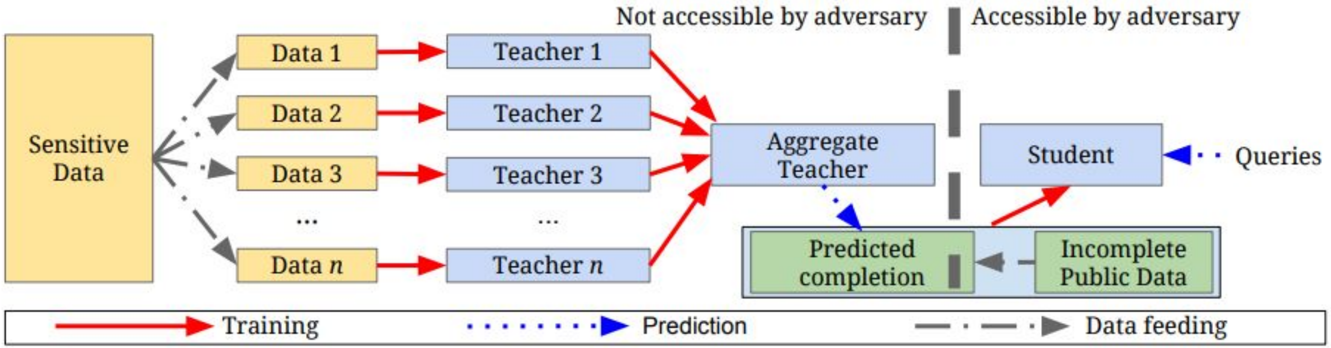
Gap: $$1-(P_\theta(y_1|x)-P_\theta(y_2|x))$$

Differential Privacy:

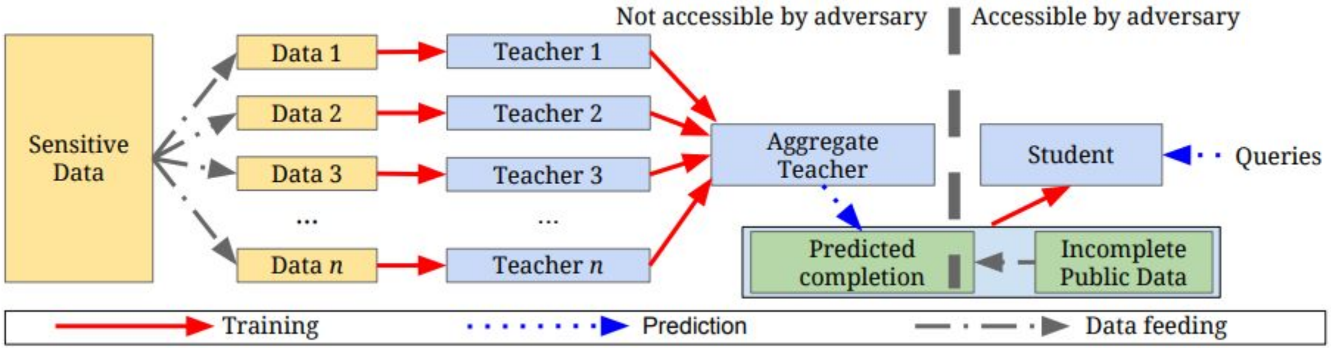


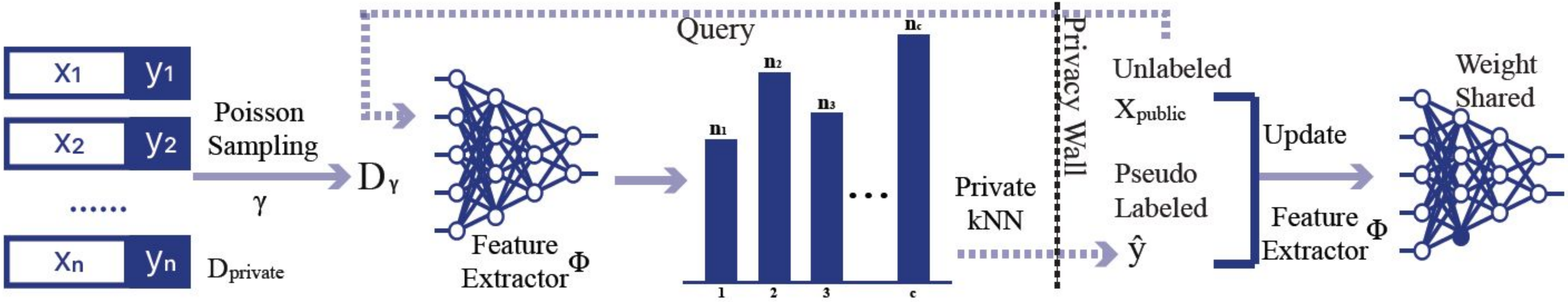
$$Pr[M(D)\in S]\leq \exp(\varepsilon)\cdot Pr[M(D')\in S]+\delta$$

Compute Privacy: from an **Ensemble of Models** with PATE to a **Single Model** with Private kNN

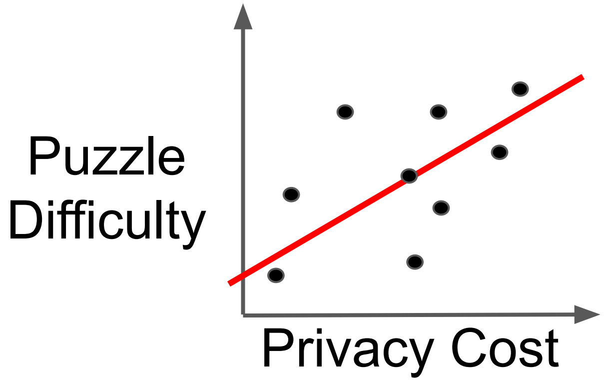


Compute Privacy: from an **Ensemble of Models** with PATE to a **Single Model** with Private kNN





Map from Privacy Cost to Puzzle Difficulty



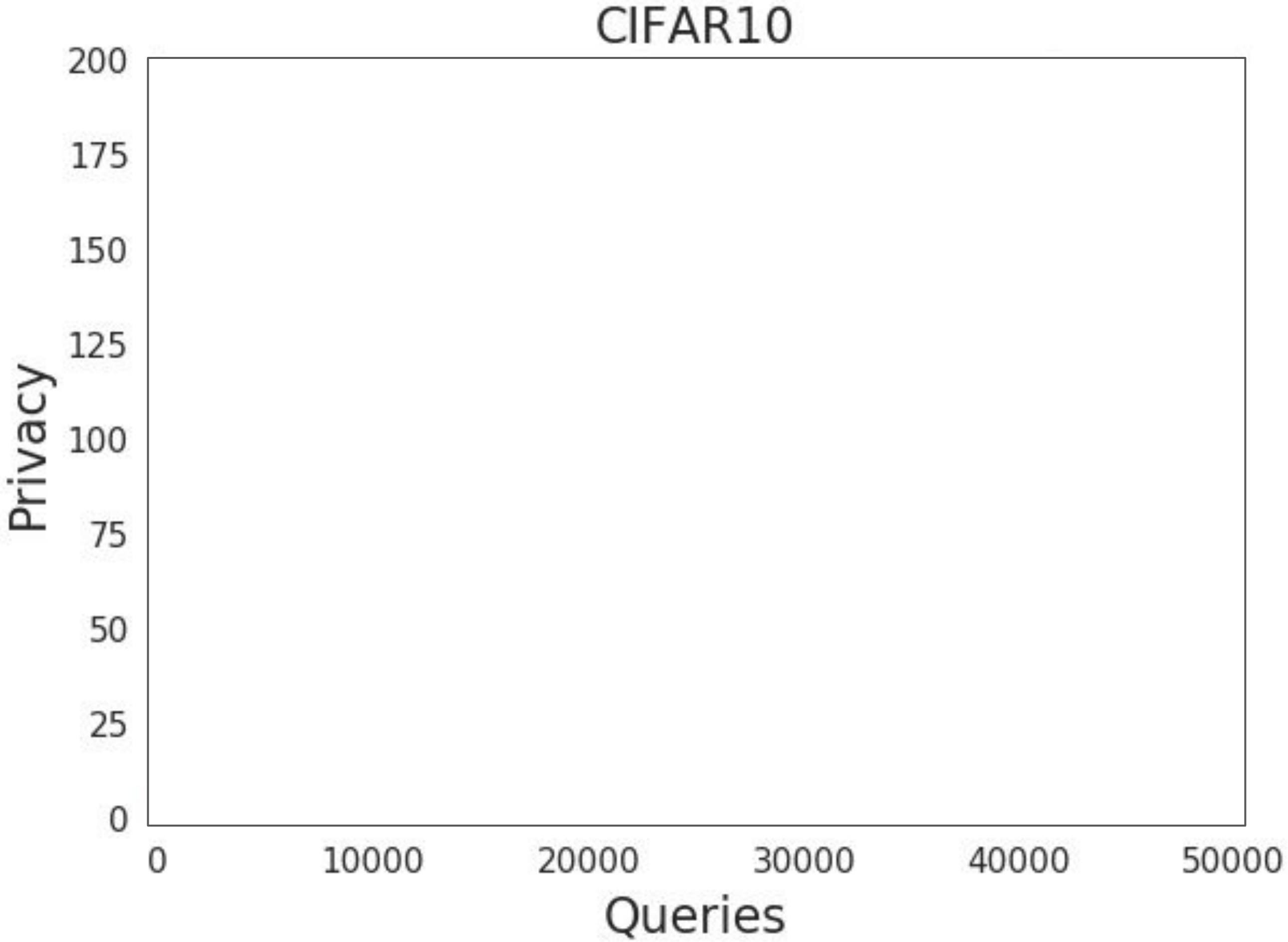
Linear Model - map from the Privacy Cost of a user to Desired Query Time ~2X for legitimate users and then to the Difficulty of the Puzzle (# of leading zero bits in HashCash).

New Query:

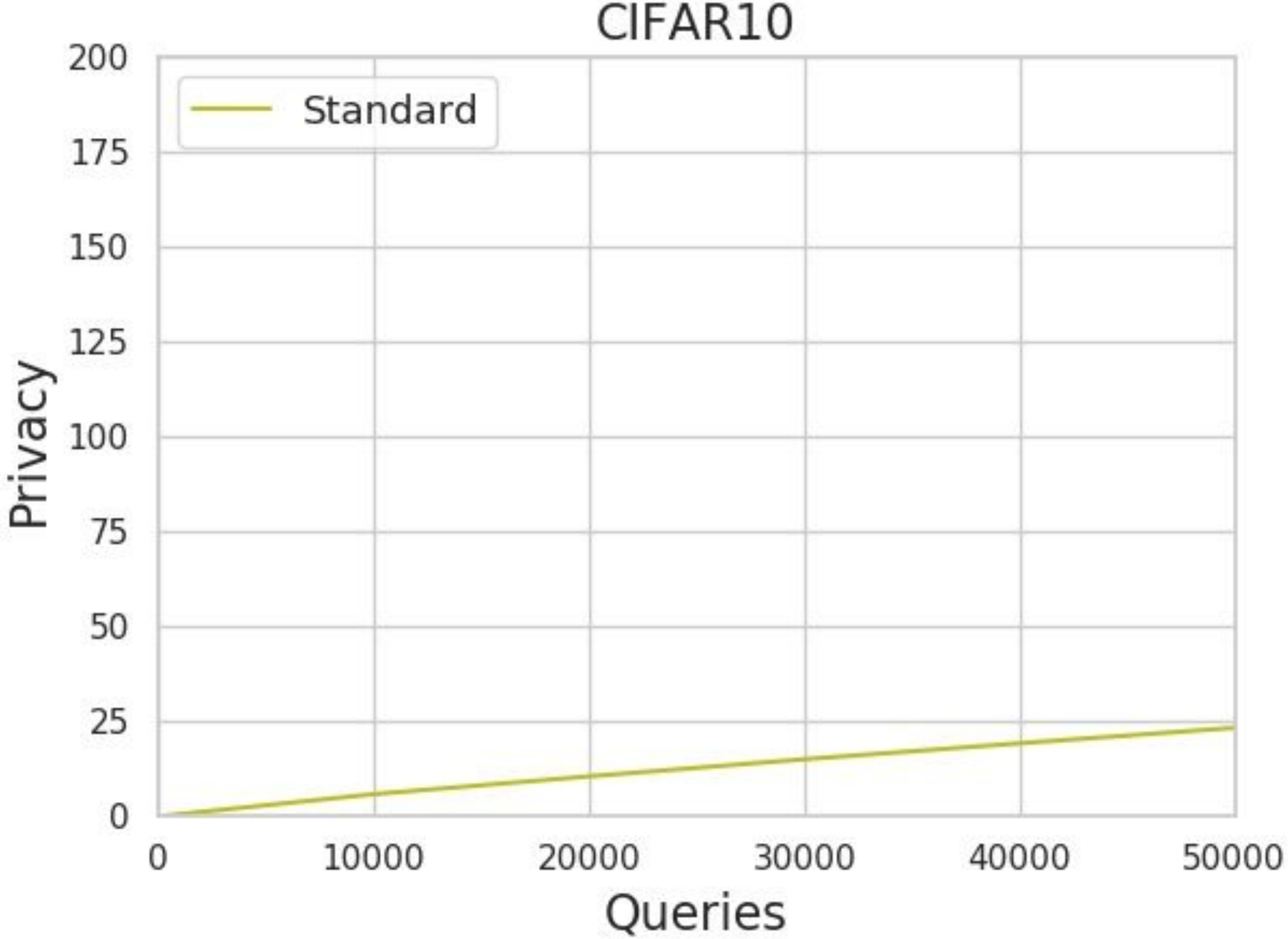
Puzzle Difficulty = Model(Privacy cost)

1. Current attacks & defenses
2. Our defense method based on proof-of-work
3. **Empirical evaluation**
4. Conclusions & Future work

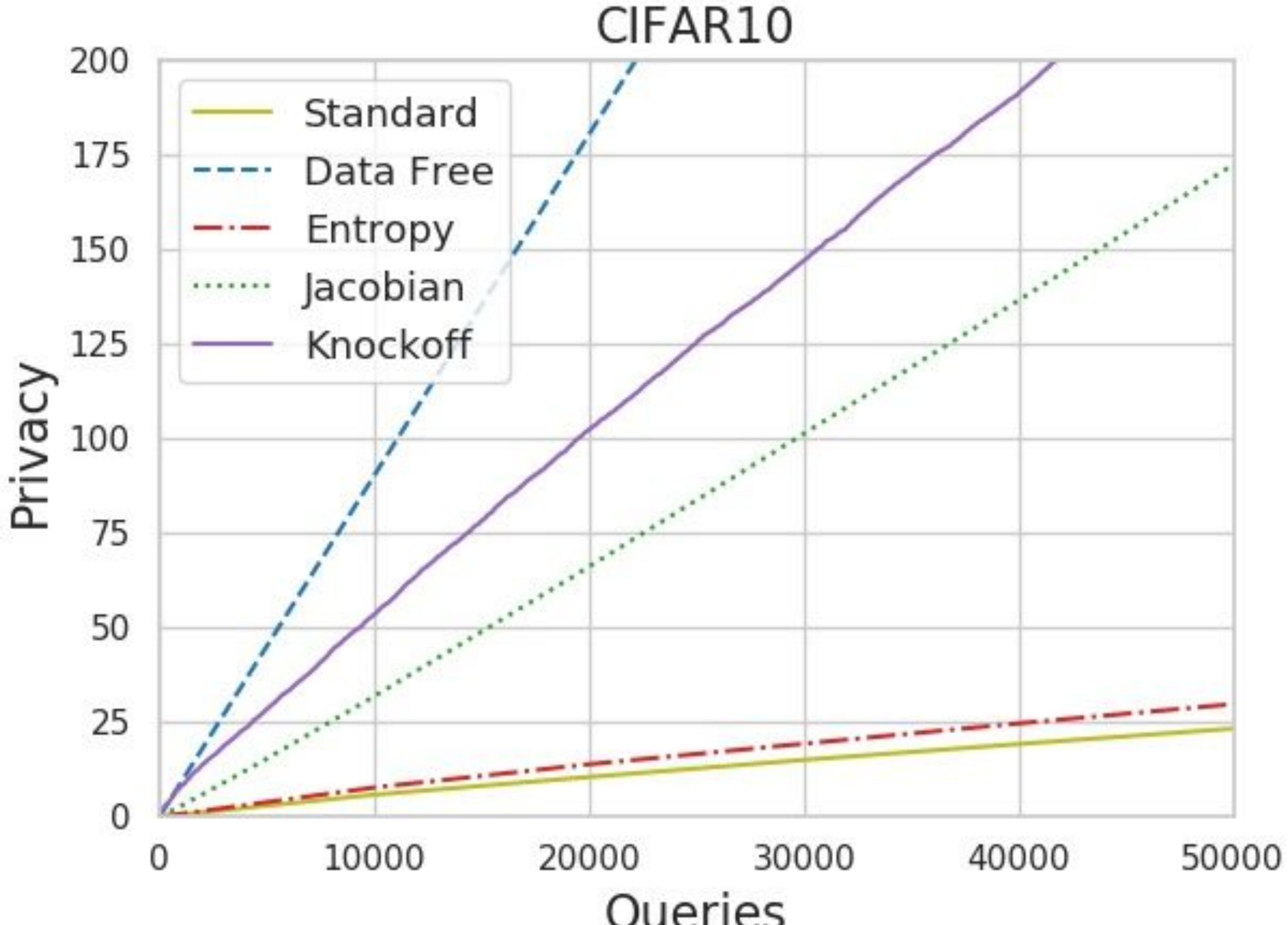
User’s Privacy cost vs # of Queries



Legitimate user's Privacy cost vs # of Queries



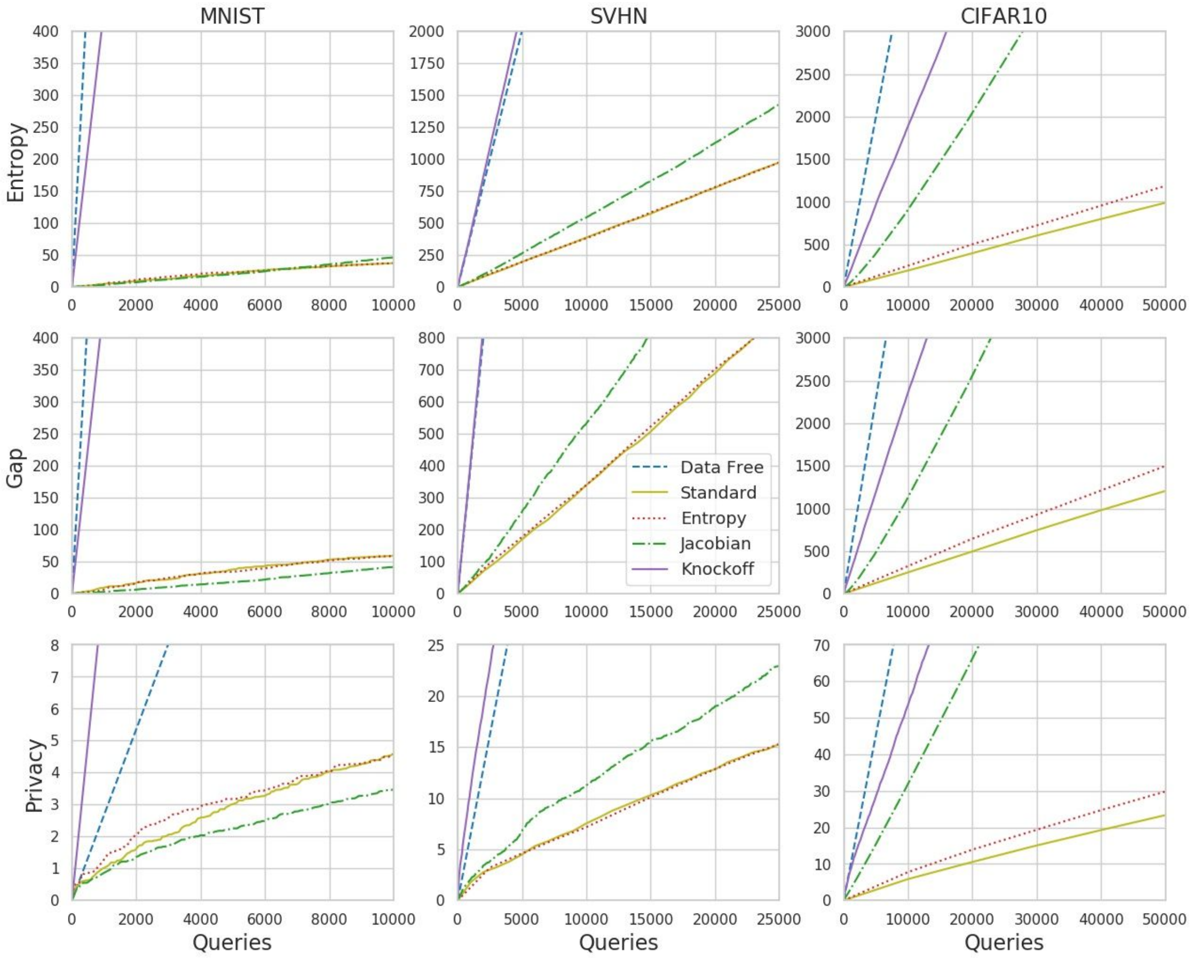
Attackers' Privacy cost vs # of Queries



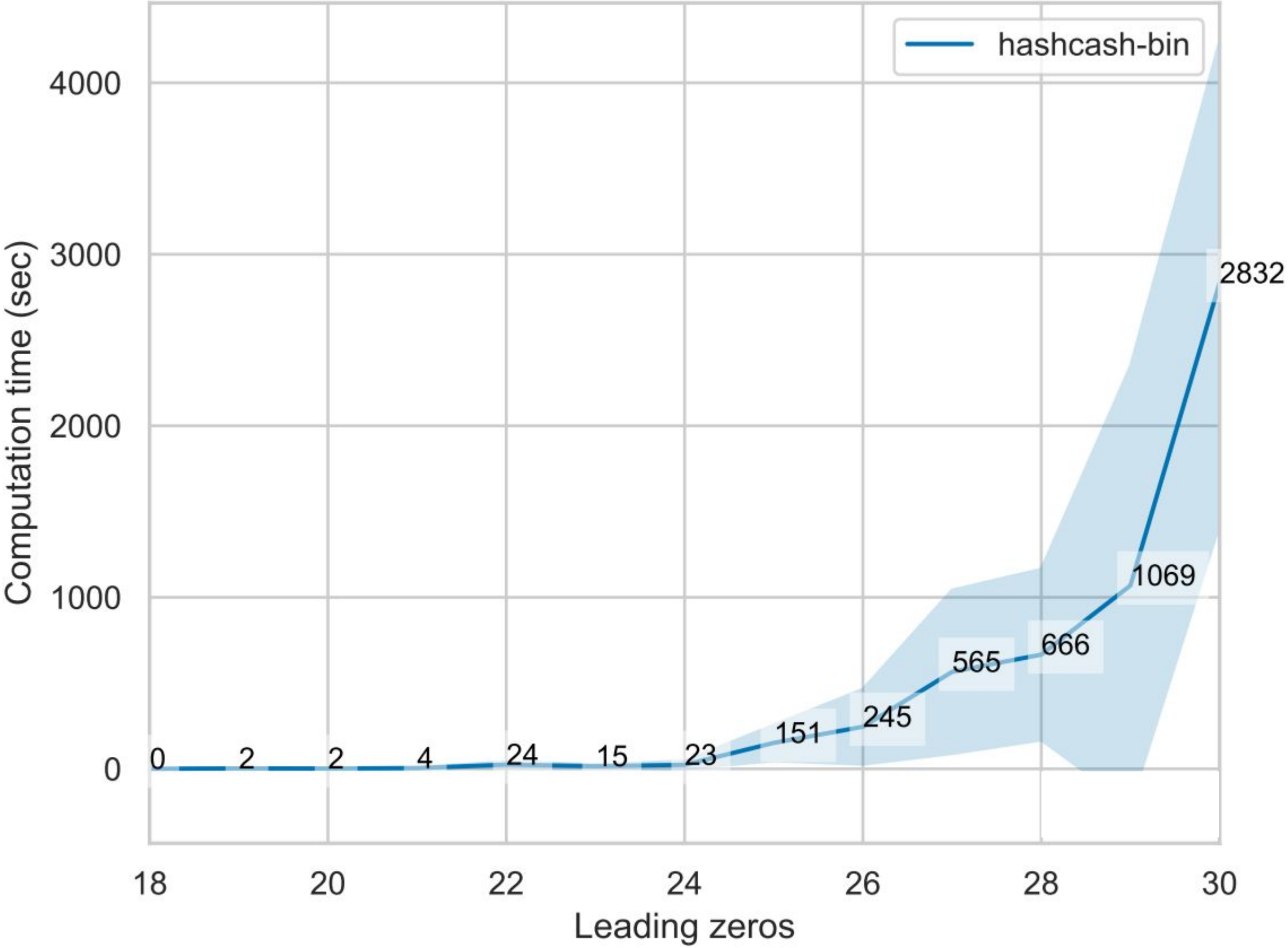
1. Privacy cost gives better distinction between legitimate users & attackers.

2. Attacker can estimate Entropy & Gap much easier.

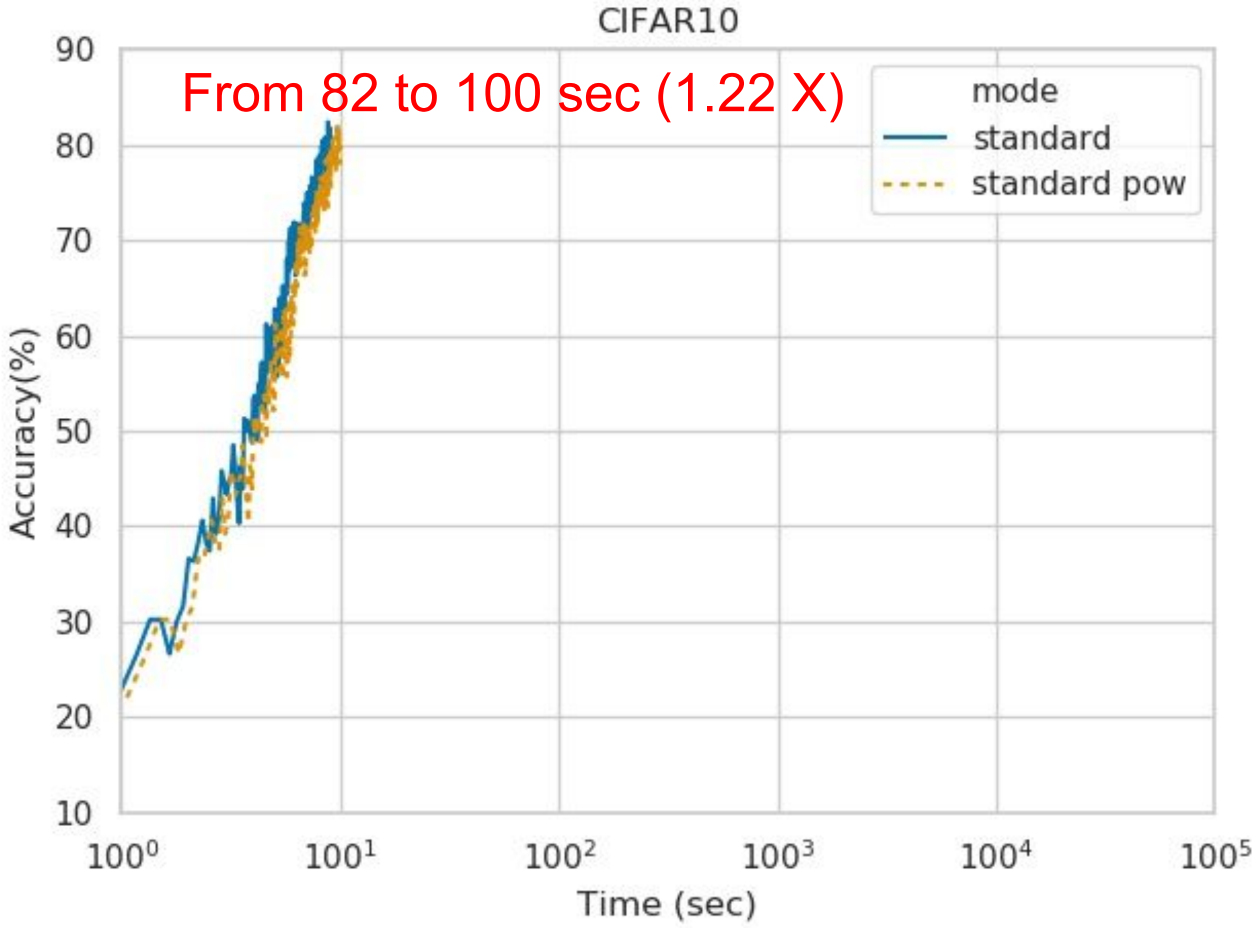
3. Similar performance on: MNIST, Fashion MNIST, SVHN, CIFAR10, ImageNet.



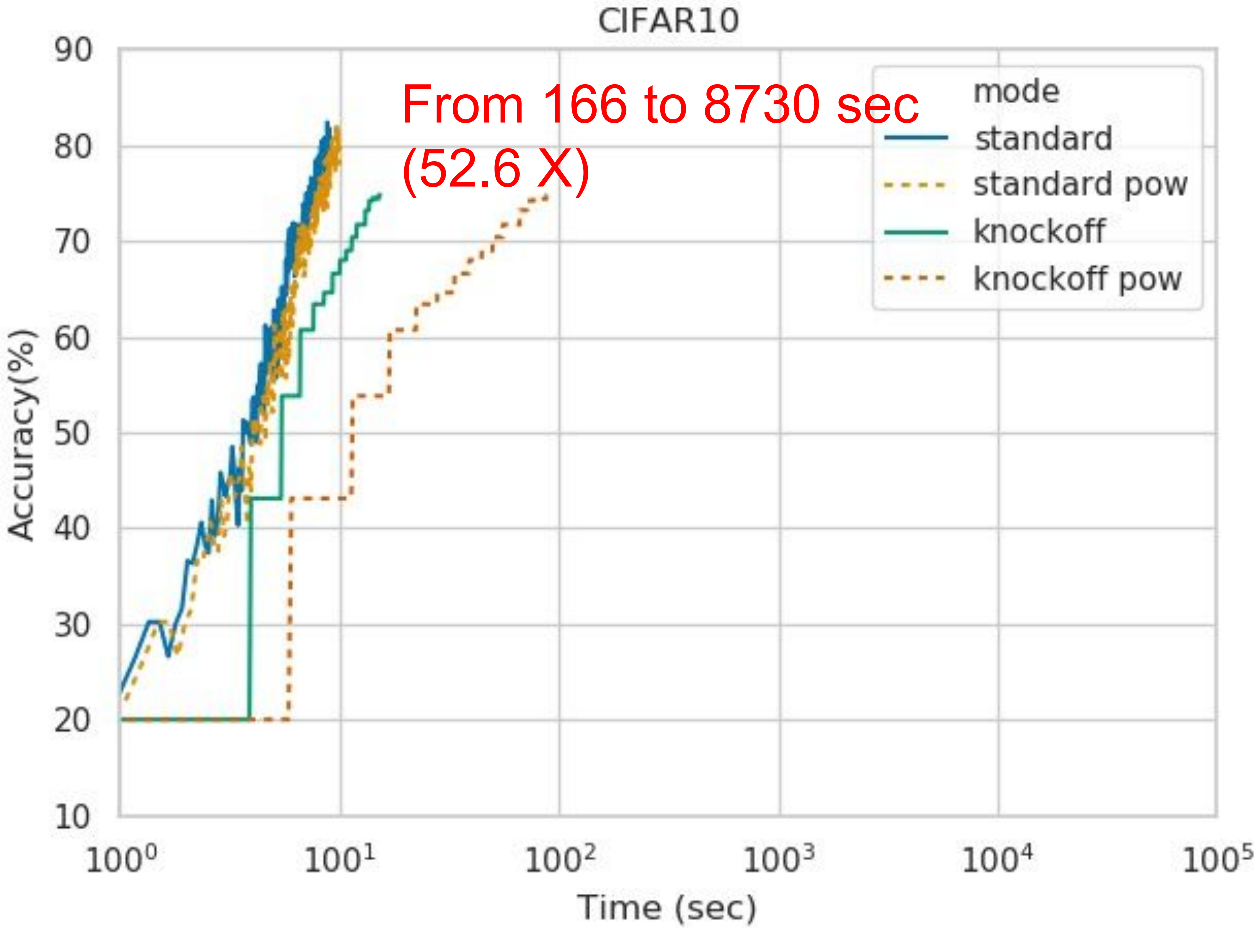
HashCash cost function for proof-of-work



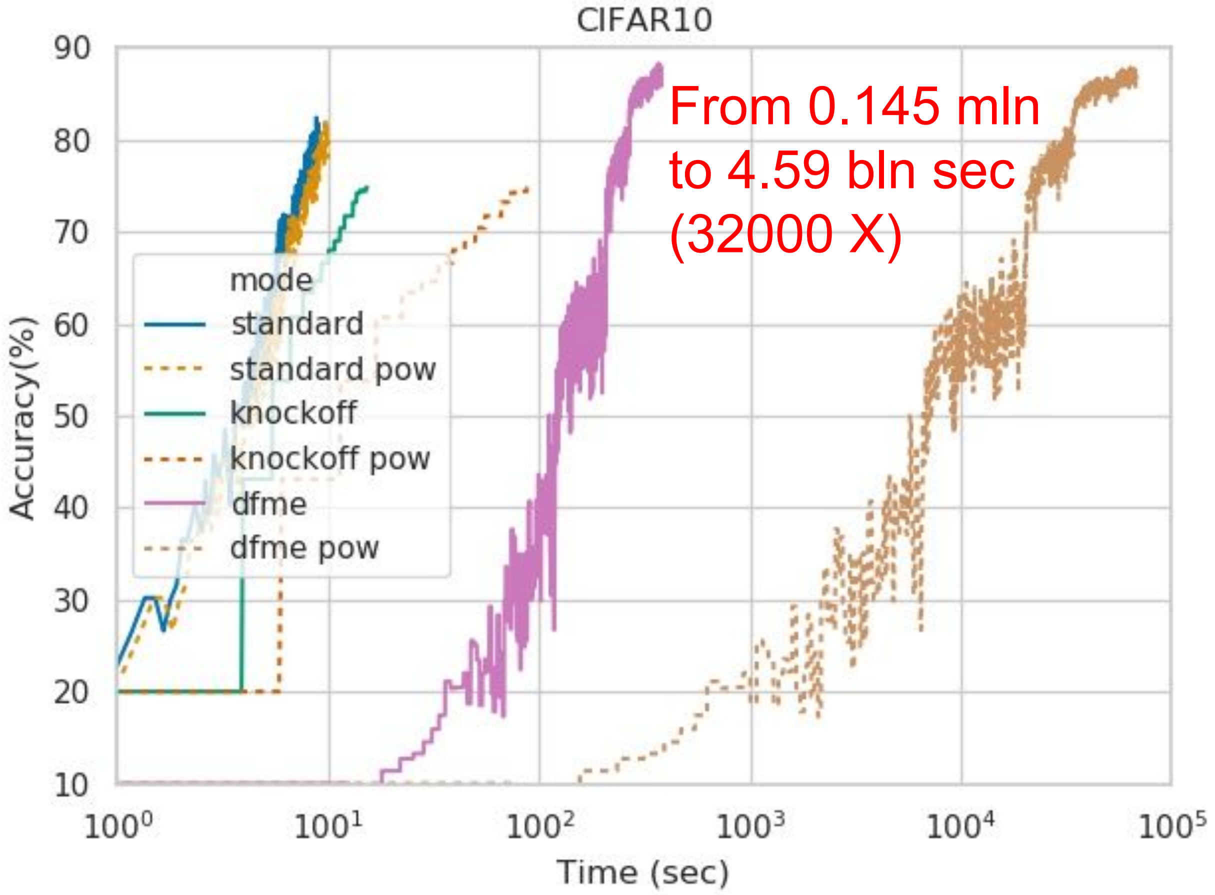
Increased query time for legitimate users with PoW



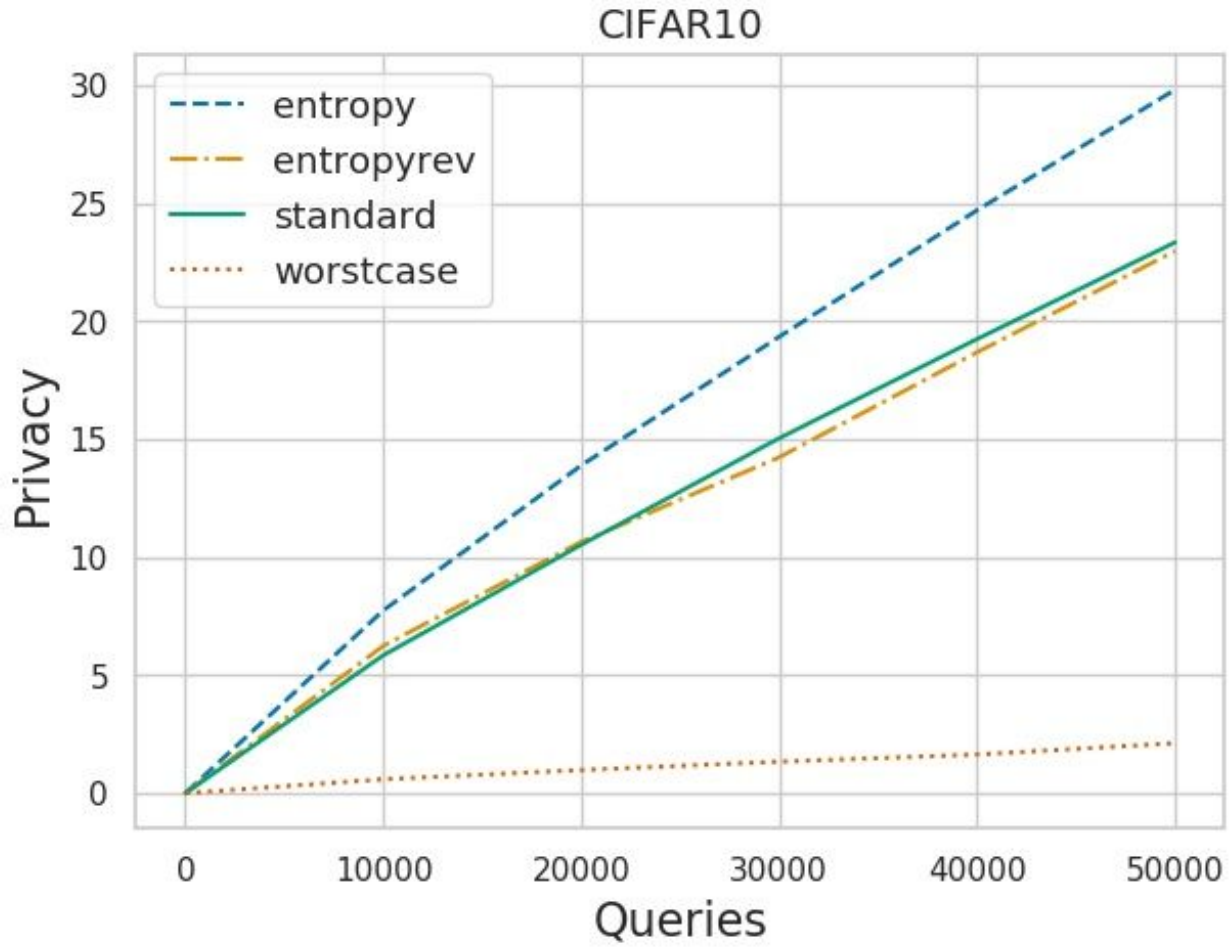
Increasing query time of Knockoff attack using PoW



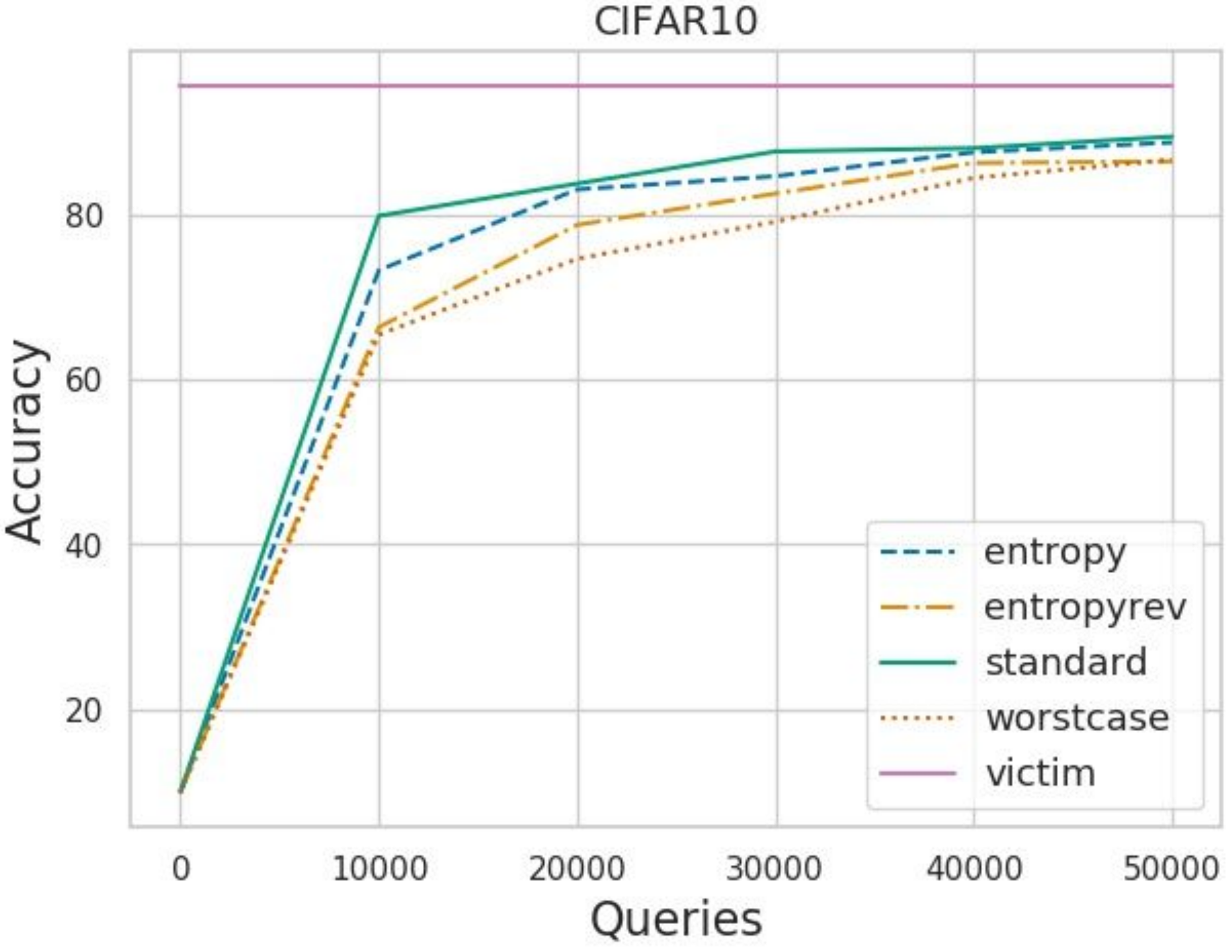
Increasing query time of Data Free using PoW



Privacy cost of **adaptive attacks** against our PoW



Accuracy of **adaptive attacks** against our PoW



1. Current attacks & defenses
2. Our defense method based on proof-of-work
3. Empirical evaluation
4. **Conclusions & Future work**

Conclusions

1. **New defense** against **Model Extraction Attacks** - prevent adversaries from stealing a model exposed via a public API.
2. Use **privacy cost** to measure the amount of information leakage from a set of queries. Store the cost per user.
3. **Proof-of-work mechanism** adaptively increases the computation time of querying API based on users’ cost with:
    a. No impact on a model’s owner;
    b. Negligible overhead for legitimate users (~2X);
    c. High increase in the querying time for many attackers (up to 3 orders of magnitude).

Future Work, Suggestions & Questions

1. Next steps: harness the **state-of-the-art out-of-distribution detection methods** to detect out-of-distribution queries, increase the users' cost and refrain from answering such queries.
2. How to determine the **difficulty of the puzzle based on users' privacy cost** in a more **general way** (hardware independent)?
3. How to design a cost function that **does not reveal the difficulty of a puzzle** before it is solved?
4. What **other attacks** should we test against?
5. What **other defenses** should we compare with?
6. How to design a **better adaptive attack**?