



Threat Models for Data Privacy in ML

Privacy-preserving ML | Adversaries by what they see (API, weights, updates, server, operator), attack goals (membership, attribute, reconstruction, linkage), base rates, the DP bound on attacks, LINDDUN per element, and a one-page threat model

#ThreatModel #Privacy #LINDDUN #MembershipInference #TrustAssumptions

Why a threat model first

Protecting a house		Protecting training data
Who might break in: burglar, guest, cleaner	≈	Adversary: API user, other client, server, cloud operator, insider
What each can see: street, living room, keys	≈	View: outputs, weights, updates, memory
What they want: jewels, papers	≈	Goal: membership, attribute, reconstruction, linkage
Whom you trust with a key	≈	Trusted party: curator, aggregator, chip vendor
Lock, safe, alarm	≈	DP, secure aggregation, HE, TEE
WHERE THE ANALOGY BREAKS A burglar takes the jewels once; data leaks can be repeated by every query, combined with other datasets years later, and cannot be taken back.		

A threat model is a burglary plan read by the defender

Before Threat modelling for privacy, can you explain...

	Not yet	Partly	Yes
MUST KNOW (ε, δ)-DP definition Learn: responsible-ai 04	☐	☐	☐
MUST KNOW Membership inference and memorisation Learn: responsible-ai 04	☐	☐	☐
HELPFUL STRIDE and data-flow diagrams Learn: cyber course: threat modelling	☐	☐	☐
HELPFUL Bayes' rule with base rates Learn: maths: distributions and Bayes	☐	☐	☐

Check before the sheet

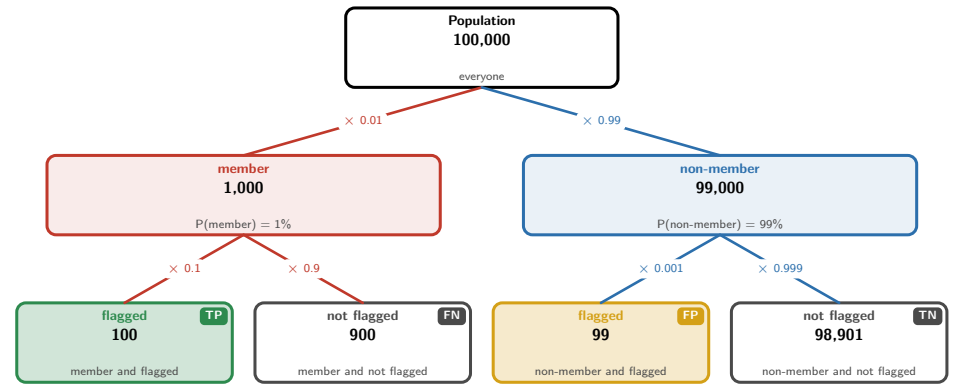
Adversary	Sees	Typical attack	Answer
API user (black box)	outputs, scores, generated text	membership inference, extraction of memorised text, model stealing	DP training, output rounding, rate limits
Weight holder (white box)	parameters, gradients on demand	stronger membership and reconstruction	DP training; never assume weights stay secret
FL server, aggregator	every client's update	gradient inversion, property inference	secure aggregation (04) + DP (02)
Other FL clients	the global model each round	infer other clients' data from model changes; poisoning	DP on the aggregate, robust aggregation (03)
Cloud or infrastructure operator	memory, disks, network of the machine	read data and model in use	HE (05), TEE (06)

Data recipient	a released dataset or synthetic data	linkage, attribute inference	DP release (02, 07), not ad hoc anonymisation (08)
Insider	logs, prompts, backups, feature stores	browse or exfiltrate	access control, minimisation, audit

Adversaries by what they see. Each row is a separate threat model; a system usually faces several

Attack goals and the numbers that matter

- Property inference — a fact about the whole dataset (40 % of records from one hospital).
- Membership inference — one bit per person: was Alice's record used? Sensitive when the dataset itself is sensitive (a cancer registry).
- Attribute inference — Alice's diagnosis, given her other attributes; also possible from a model trained without her.
- Reconstruction — whole records: memorised text, images rebuilt from gradients, rows rebuilt from too many exact statistics.
- Linkage, re-identification — a name attached to a supposedly anonymous record (sheet 08).



Evidence "flagged": $TP + FP = 100 + 99 = 199$



$$P(\text{member} \mid \text{flagged}) = 100 / 199 = 50.3\%$$

Natural frequencies: $P(H|E) = TP / (TP + FP) = 100 / (100 + 99) = 0.503$ (50.3%)
 Bayes' rule: $P(H|E) = P(E|H) \cdot P(H) / [P(E|H) \cdot P(H) + P(E|\neg H) \cdot P(\neg H)] = 0.1 \cdot 0.01 / (0.1 \cdot 0.01 + 0.001 \cdot 0.99) = 0.503$
 Prior 1% $\rightarrow$ posterior 50.3%: the evidence multiplies the odds by the likelihood ratio $LR = P(E|H) / P(E|\neg H) = 0.1 / 0.001 = 100$
 Without the evidence: $P(H|\neg E) = FN / (FN + TN) = 900 / (900 + 98,901) = 0.9\%$
 Where: H: member (the hypothesis); E: flagged (the evidence); $\neg$: not; P(H) = prior, the base rate before seeing E; P(E|H) = sensitivity (true-positive rate); P(E| $\neg H$) = false-positive rate; P(H|E) = posterior; TP, FN, FP, TN: counts of true / false positives / negatives among N = 100,000 people (or items)

Why low base rates make attacks look worse than they are: an attack with TPR 10 % at FPR 0.1 % tested on records of which 1 % are training members. Half of its 'member' verdicts are wrong

Highest true-positive rate any attack can reach at a chosen false-positive rate under (ϵ , δ)-DP

FORMULA $TPR = \exp(\epsilon \cdot \text{FPR}) + \delta$

WHERE

- TPR = upper bound on the attack's true-positive rate (members correctly flagged) (0-1)
- ϵ = ϵ of the trained model, for the same privacy unit as the attacked record (none)
- FPR = false-positive rate the attacker accepts (non-members wrongly flagged) (0-1)
- δ = δ of the guarantee (probability)

WORKED

$\epsilon = 1$
 $\text{FPR} = 0.001$
 $\delta = 1e-05$
 $TPR = \exp(1) \times 0.001 + 1e-05$
 $TPR = 0.002728$ (0.27 %)

RESULT $TPR = 0.002728$ (0.27 %)

READ IT At $\epsilon = 1$ no attack can flag more than about 0.27 % of members while wrongly flagging 0.1 % of non-members: barely better than guessing. At $\epsilon = 8$, $e^8 \approx 2981$ and the bound is vacuous at this FPR (above 1).

MISTAKES

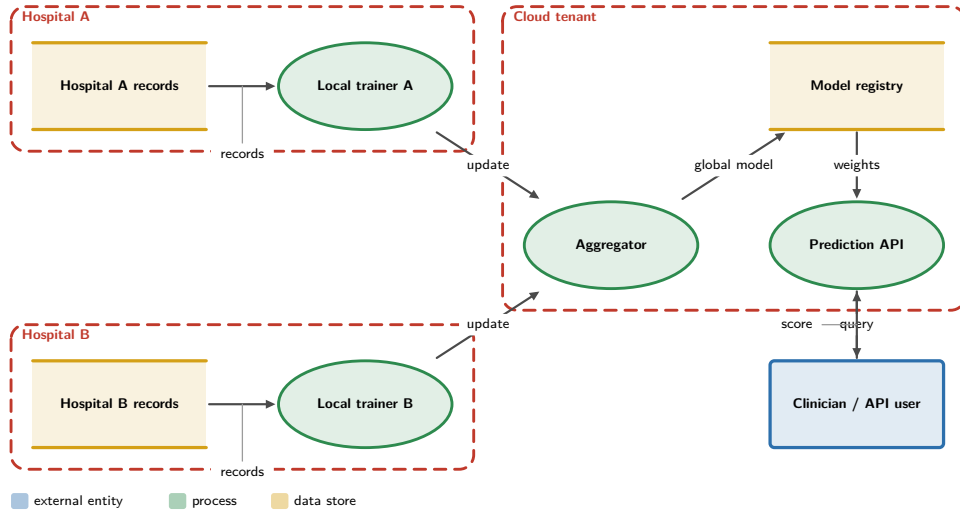
- The bound is worst case: real attacks on DP models usually stay far below it, which is why ϵ of 4-8 can still resist practical attacks
- A bound for example-level ϵ says nothing about a user with 500 examples: use the user-level ϵ
- δ must stay much smaller than $1 / \text{number of records}$, or the $+$ δ term dominates

What DP promises against any membership attack (hypothesis-testing view of DP, Kairouz et al., 2015)

Measure attacks the right way

Report TPR at FPR 0.1 % or lower on a balanced, same-distribution member/non-member split, against a reference such as LiRA-style calibrated attacks. An AUC of 0.6 averaged over everyone can hide a few outliers that are identified with near certainty: privacy is about the most exposed person, not the average.

Model the system element by element



Cross-silo federated training between two hospitals and a cloud aggregator, with the trust boundaries a threat model must name

	L Linking	I Identifying	N Non-repudiation	D Detecting	DD Data disclosure	U Unawareness	NC Non-compliance
Hospital records	Med	High	Low	Low	High	Med	High
Update A $\rightarrow$ aggregator	Med	Med	--	Low	High	Low	Med
Aggregator	Med	Med	--	Med	High	Low	Med
Model registry	Low	Med	--	Low	Med	Low	Med
Prediction API	Med	Med	Low	High	Med	Med	Med

Low... Crit risk level -- not applicable data store $\rightarrow$ data flow process

LINDDUN instead of STRIDE: privacy threats per element of the diagram above (high / medium / low = residual risk without extra controls; -- = not applicable)

Reading the matrix: the update flow and the aggregator carry **data disclosure** (gradient inversion) $\rightarrow$ secure aggregation; the API carries **detecting** (membership: 'was this patient in the study?') $\rightarrow$

DP training; the stores carry **identifying** and **non-compliance** → minimisation, retention, a DPIA.
A cell without a named control is an open risk, not a solved one.

Gradient inversion by the aggregator

RISK HIGH

LIKELIHOOD 2/4

SEVERITY 4/4

WHO

Curious cloud operator · Curiosity, resale

HOW

Reads per-client updates

TARGET

Patient images

IMPACT

Reconstructed scans of identifiable patients

CONTROLS

Secure aggregation

Client-side clipping + DP noise

DETECT

None from inside the protocol: prevent, do not detect

Membership probe on the API

RISK HIGH

LIKELIHOOD 2/4

SEVERITY 3/4

WHO

Insurer or journalist · Learn who is in a disease cohort

HOW

Thousands of crafted queries

TARGET

Cohort membership

IMPACT

Disclosure of a diagnosis

CONTROLS

DP-SGD with stated ϵ

Return labels, not raw scores

Query rate limits

DETECT

Query-pattern monitoring

Poisoned update from one silo

RISK MEDIUM

LIKELIHOOD 1/4

SEVERITY 3/4

WHO

Compromised participant · Backdoor the shared model

HOW

Crafted local update

TARGET

Model integrity

IMPACT

Targeted misdiagnosis

CONTROLS

Update norm clipping

Robust aggregation

Participant vetting

DETECT

Per-round anomaly scores on updates

Three scenarios from the same system, rated on likelihood × severity before extra controls

Write it down: the one-page threat model

Fields every privacy threat model states

- ✓ baseline / mandatory ☐ recommended – verify in your context
- ✓ Assets and privacy unit: which records, and is one unit an example, a user, a device or a hospital?
- ✓ Adversaries and their view (one line each, from the table above)
- ✓ Adversary capability: honest-but-curious or malicious; how many parties may collude

- ✓ Trusted parties and why they deserve it (contract, audit, attestation)
- ✓ Auxiliary information assumed: none, public registers, or everything except the target (DP’s worst case)
- ☐ **MUST** Attack goals in scope and out of scope (e.g. model stealing out of scope)
- ☐ **MUST** Chosen controls per threat and the residual risk accepted, signed by an owner
- ☐ **SHOULD** Review trigger: new release channel, new data source, new participant

Interview trap

'We use federated learning, so the data is private.' Ask: private from whom? FL hides raw data from the server’s disk, not from the server’s analysis of updates, not from API users of the final model, and not from other participants who see the global model.

MYTH

FACT

✗ The attacker only sees our API, so weights do not matter

→

✓ Assume weights eventually leak or are released
Insiders, backups, edge deployment and open-weight releases turn black-box threats into white-box ones.

✗ Nobody has the auxiliary data to re-identify

→

✓ Auxiliary data appears later and is cheap
Linkage attacks used public voter rolls, IMDb ratings and social media (sheet 08).

✗ Encryption in transit and at rest covers ML privacy

→

✓ It protects storage and wires, not computation or outputs
Data is decrypted in memory for training, and the trained model can still leak.

✗ An honest-but-curious model is pessimistic enough

→

✓ It is a stated assumption, not a worst case
Malicious servers can send crafted models to clients to make their updates leak more.

Common threat-model mistakes

Tadata | tadata.fr | contact@tadata.codes

Key takeaways

01 **Name the adversary and its view**
API user, weight holder, server, operator, recipient

02 **Name the trusted party**
every technique moves trust, none removes it

03 **State the privacy unit**
example, user, device or silo

04 **Mind base rates**
TPR at very low FPR on realistic member shares

05 **DP bounds every attack**
 $TPR \leq \epsilon \hat{\epsilon} \times FPR + \delta$

Remember

Q1 **Name four adversary views in an ML system.**

A Black-box outputs, white-box weights, client updates (FL server), machine memory (cloud operator); also released datasets and logs.

Q2 **Membership vs attribute inference?**

A Membership: was this record in training? Attribute: what is this person's sensitive value, given their other attributes?

Q3 **Attack with TPR 10 %, FPR 0.1 %, 1 % members: what share of its 'member' flags are right?**

A About 50 %: 100 true hits vs 99 false ones per 100,000 records tested.

Q4 **Under $\epsilon = 1$, $\delta = 1e-5$, maximum TPR at FPR 0.1 %?**

A $e^1 \times 0.001 + 0.00001 \approx 0.0027$, about 0.27 %.

Q5 **What are the seven LINDDUN categories?**

A Linking, Identifying, Non-repudiation, Detecting, Data disclosure, Unawareness and uninter-venability, Non-compliance.

Q6 **Which control answers gradient inversion by the server?**

A Secure aggregation so the server sees only the sum, plus clipping and DP noise so even the sum reveals little.

Q7 **Why assume weights will leak?**

A Insiders, backups, edge deployment and later open releases turn a black-box model into a white-box one.

Q8 **What does DP assume about auxiliary information?**

A The worst case: the attacker may know every record except the target's, so the guarantee holds whatever they learn later.

Q9 **Why is 'honest-but-curious' a choice, not a fact?**

A It assumes the party follows the protocol; a malicious server can craft models that make client updates leak more.