Google

Secure AI Framework

# Google's Approach to Protecting Privacy in the Age of AI

Authors
Kareem Amin, Reiner Critides, Julien Freudiger, Andreas Terzis, Royce Wilson

## Abstract

AI products introduce new privacy challenges. Finding the right privacy solution is central to developing innovative products, especially as AI models increasingly handle user data. In this paper, we propose a framework to reason about privacy in AI, and discuss how Privacy Enhancing Technologies (PETs) enable novel user experiences by reducing privacy risks in the AI development lifecycle. We argue that privacy protections are not inherently at odds with utility; in contrast, we discuss how building privacy into products from the start can create better, more trustworthy experiences for everyone.

# Table of contents

# 1. Introduction

Larger datasets generally improve AI model performance[1]. As a result, there has been a race to train on as much data as possible. However, this increasing data volume, especially when it includes user data, amplifies privacy risks: AI models can sometimes output training data[2] or infer sensitive information about individuals. Because of these privacy risks, we must find reliable ways to protect user privacy throughout the AI development lifecycle.

Mitigating these risks is a challenge. This paper focuses on technological solutions, and explores how Privacy Enhancing Technologies (PETs) play a crucial role in achieving privacy by design. PETs are tools that have been developed to mitigate privacy risks starting in the 1980s[3]. In the age of AI, PETs can help mitigate risks by reducing the need for data collection (using federated training or multi-party computations), by better protecting collected data (using homomorphic encryption and differential privacy), by reducing memorization[4] risks when training (with techniques such as differentially private training, and privacy testing), and by reducing exposure when deploying models (using on-device serving and server-side trusted execution environments). There is also research to quantify the privacy risks of generative models and effectiveness of PETs in protecting such models[5] [6].

Discussions about deploying PETs in products are often framed in terms of trade-offs, defined by their performance on technical benchmarks. These benchmarks typically compare the impact a PET has on a deployment's usefulness against a zero-privacy baseline — a deployment with no technical privacy protections. Because privacy protection is always an essential product requirement, we argue that a zero-privacy deployment is not a realistic product configuration. As a consequence, this type of analysis is largely theoretical. In contrast, we posit that the most important decisions happen when fitting the right PET to the product or to the right place in the development workflow.

In this work, we move beyond considering PETs as imposing a zero-sum trade-off between privacy and utility. We assert that selecting the right PET is *a considered choice* for your

[1] arxiv.org/abs/2001.08361
[2] arxiv.org/abs/2311.17035
[3] www.freehaven.net/anonbib/cache/chaum-mix.pdf
[4] We intend a restricted definition of "memorization": whether a model can be induced to generate near-facsimiles of some training examples when prompted with appropriate instructions. Models do not "contain" bit-wise or code-wise copies of their training data. Rather, if a model can be induced to generate close copies of certain training examples by supplying appropriate instructions to guide the model's statistical generation processes, then that model is said to have "memorized" those examples. This is an area of active ongoing research.
[5] arxiv.org/abs/2202.07646
[6] ieeexplore.ieee.org/abstract/document/9152761

product, and that deploying a PET often enables the utility enjoyed by users in the first place. When product owners choose PETs, they make a decision that captures their product vision and reflects their values to end users. In this paper, we provide a structured approach to explore and review PETs when deploying AI. Specifically, we rely on Google's privacy work over the past decade to introduce best practices that can help organizations develop and use AI models while safeguarding user privacy.

Popular times and live busyness information[7] in Google Maps are a good example of how PETs have been deployed to build products outside of the AI space that delight users. Gboard[8] is another example that shows how to take advantage of PETs to achieve a great AI product. As we explore how PETs can be used to protect privacy in AI products, we argue that there is a unique opportunity for organizations to innovate in the privacy space and, as a result, improve industry practices and user experiences. PETs can reconcile AI's need for data with the fundamental right to privacy, enabling breakthroughs that benefit everyone without putting individuals at risk.

---

## PETs shift the paradigm from data scarcity towards product enablement.

---

We foresee a virtuous cycle where new PET adoption in AI products leads to the improvement of PETs, incentivizing others to adopt them and benefiting organizations and users alike.

The remainder of this paper is organized as follows. Section 2 provides a conceptual framework for reasoning about privacy in the age of AI. We provide a background on privacy, the AI development supply chains and risks, and highlight key PETs as emerging solutions in AI. Section 3 discusses challenges with deploying PETs in practice. Section 4 provides guidance to facilitate the adoption of PETs in AI systems.

[7] blog.google/products/maps/maps101-popular-times-and-live-busyness-information
[8] research.google/blog/improving-gboard-language-models-via-private-federated-analytics

# 2. Background

This section provides an overview of privacy, AI systems, and the AI supply chain. It addresses the considerations of data privacy, the architecture of AI systems, and the complexities of the AI supply chain. Understanding these elements is essential for navigating the industry's current state of AI development and deployment.

## 2.1 Data flows in AI systems

**Training data can take many forms.** When we think of data in an AI system, we might think of large *training datasets* for model development. These datasets may utilize data from a variety of sources, including public (often web-based), licensed, and/or user data. Model trainers may also use *synthetic data*: artificially generated data designed to mimic the properties of real-world data (Figure 1).

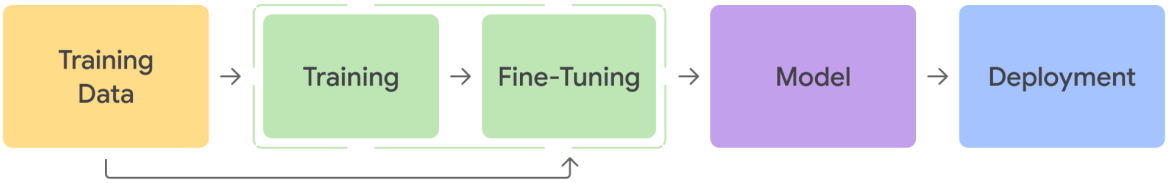


*Figure 1: A simple AI development process. A developer gathers data, and uses it for training and fine-tuning an AI model. The developer, or a different actor, then deploys the AI model.*

**Training data is not the only data that enters an AI system.** When an AI model is deployed, the *context window*, which includes *user prompts* and *retrieved context*, represents additional sources and forms of data that must be considered (Figure 2). User prompts are data submitted by users to guide the generation process, and retrieved context refers to additional data about the user or the world that an AI system may search for to guide its response.

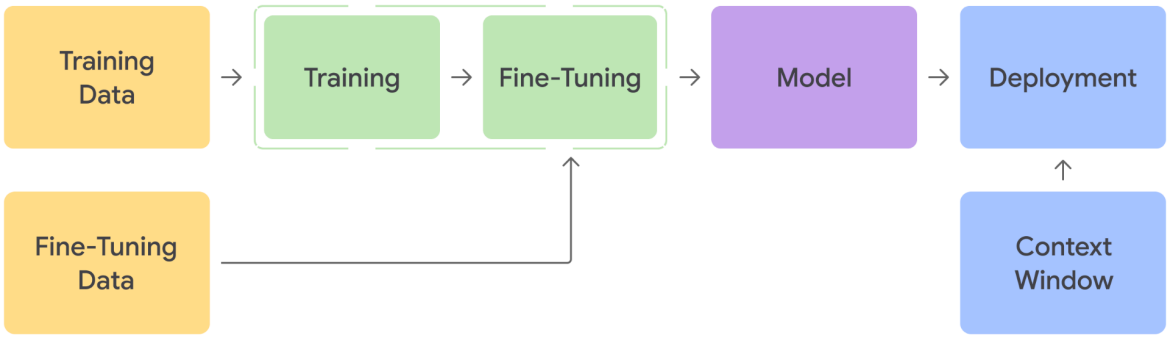


*Figure 2: Fine-tuning data may be different from training data. Additionally, during deployment, the context window (which may contain user prompts and retrieved context) may be included to generate outputs.*

Additionally, a deployed model may incorporate user-provided feedback about the generations produced from a model (Figure 3). For example, when users interact with a model and provide explicit thumbs-up or thumbs-down feedback, this user feedback can help improve a model by giving it concrete, user-focused examples. Note that such feedback may include user-generated, free-form content, which in turn may contain sensitive information.

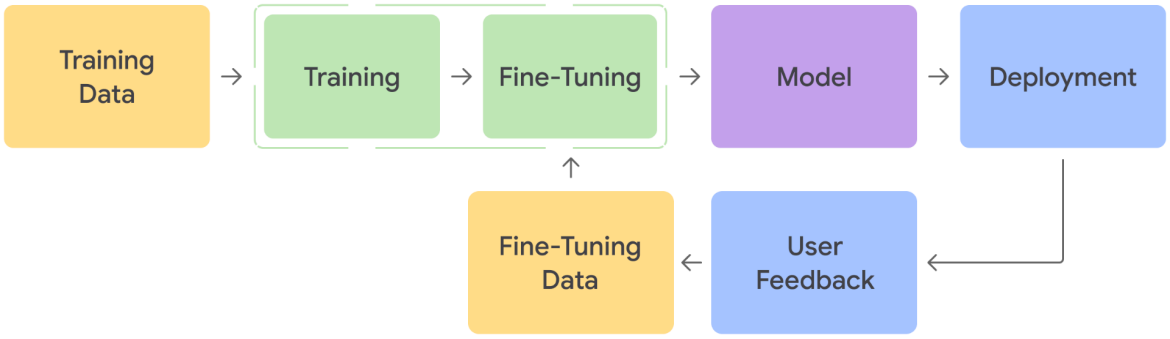


*Figure 3: Additionally, data may be collected during deployment, such as user feedback. This additional data may be used to fine-tune future models.*

As we can see, **data enters the AI system at many points and may involve different actors.** AI models are increasingly developed in two stages: pretraining and fine-tuning. Fine-tuning typically uses additional data to further train a general model for a specific task and may be done by a different actor (person or company) than the one that pretrained the model. Fine-tuning may create different technical risks than pretraining as fine-tuning datasets may be smaller and are used to train the model closer to deployment. During deployment, user data may enter the system through both user prompts and user feedback. In both cases, it's clear that the originator of the data is a specific user, though that data may be handled by yet

another actor (the developer) that might differ from the actor who trained / fine-tuned the underlying model.

**Each stage of the AI supply chain manages, processes, and transforms data in different ways.** The AI supply chain incorporates a diverse range of data types, each requiring specific handling. For example, in many AI systems, user prompts are stored separately from pretraining and fine-tuning data and are not by default part of the training pipeline.

**Mitigations may be applied across the supply chain**; the choice of mitigation used depends on the types and uses of data. For example, if non-public data is present only in the fine-tuning dataset, mitigations can be directed to fine-tuning (for example, using differentially private training) and deployment. As another example, deploying the model directly onto user devices can significantly reduce the risks associated with user-created prompts, as all prompts and generations could remain on device. For a more thorough discussion of mitigations and where they may be applied, see Sections 2.3 and 3.

## 2.2 Privacy overview

Privacy is a broad concept that captures many expectations about how user data is handled. When user data is used to develop AI products, there is a risk that data capable of identifying a user, or important facts about a user, is exposed to an unintended audience. PETs are particularly good at addressing this class of risks, and we therefore focus our attention on it. However, we acknowledge that PETs are not a panacea and that several other important aspects of privacy are not addressed by mitigating this risk alone.

---

PETs are tools that can be used to meet user expectations.

---

Determining whether a user's data surfaces in an unintended way is highly context-dependent. The following is a non-exhaustive list of relevant factors. It is important to note that these factors are not mutually exclusive and often interrelated. For example, a good legal analysis might consider promises made to the user as well as user expectations. However, we offer these as a helpful checklist for the product designer to develop confidence that their product vision does not contain obvious privacy problems.

**Legal and regulatory requirements**. These set bright lines for how data may be used. However, there is often a compelling business case to foster user trust by going beyond legal or regulatory minimums.
**Promises made to the user**. One must consider all communications made to the user (including privacy policies, notices, and disclosures). Technology platforms have a responsibility to ensure user data is used or collected in a manner consistent with these user communications.
**User expectations**. Users should not be surprised with how their data is used. This means regularly evaluating whether data practices would surprise or confuse an average user, given the context of the service. If a user would be surprised to learn how their data is being handled, it's a strong signal to reconsider the approach.
**Exceeding expectations**. Growing a business requires not just building products that users love, but maintaining their trust. Safeguarding a user's data beyond what they expected can help foster trust, expand a company's user-base, and drive continued product adoption.
**Ethical obligations**. Certain privacy protections are simply the right thing to do[9]. Certain types of data such as demographic information, location, or personal correspondences can be used to cause real harm to users. Technology platforms are ethically obligated to defend against those harms.

Note that violating any of these can meaningfully hurt the business, as users will seek alternatives when they are unpleasantly surprised. Conversely, exceeding user expectations and developing privacy-forward products can increase user trust and grow the business.

We next discuss how privacy risks tend to appear in AI products. While there are many risks inherent in AI development[10], we highlight the ones that are best mitigated by PETs. These are vectors that might lead to unintended sharing of user data, as discussed in Section 2.1.

### 2.2.1 Sensitive[11] data disclosure

AI models produce outputs that are influenced by training data. In the extreme, these outputs could include examples from the model's training inputs, disclosing the data used to train the model. The model is said to have "memorized" examples from the training data in this situation.

[9] ai.google/responsibility/principles/
[10] saif.google/secure-ai-framework/risks
[11] Throughout this paper, we use the term "sensitive data" to refer to data that some users may wish to keep private. The term is not meant to align with specific definitions for "sensitive" data that exist in various privacy laws.

Sensitive data disclosure may occur under both typical and adversarial usage. Under typical use, data may be inadvertently exposed to an ordinary user who was not seeking it. For example, a user may ask a system about other individuals with medical conditions similar to their own. If an AI system were to respond with information for a specific individual (for example, showing their medical chart), that would be an inadvertent sensitive-data disclosure.

In contrast, an adversarial user of an AI system might deliberately attempt to prompt a model into producing an output that reveals its training data. For example, an adversary might prompt an image generation model with a real person's name and check if the model produces a close copy of a real photo of them. This attack is called "training data extraction" because a successful adversary will have extracted training data points from the model.

The more information an attacker has about a user, the easier it is to extract additional information. If training or fine-tuning data contains sensitive information, such as datasets drawn from medical records, then data extraction is of heightened concern.

### 2.2.2 Inferred sensitive data

Let's take a look at some different ways sensitive information might be inferred by a model despite not containing that specific data.

*Sensitive inference.* An attacker might try to learn something about a user by interacting with an AI system, even if the system does not explicitly release information tied to that user. They might do this through side information about the user and inferential logic. For example, they might deduce that if the AI system has general information about a rare disease, it is because a particular user has it. They might arrive at this inference based on pre-existing suspicions about the user, knowledge about the incidence of the disease, or how the AI model was trained.

*Inferring data membership.* Similarly, an adversary may try to infer a single bit: whether a specific user's data was included in training or fine-tuning. This fact could be informative if data with specific properties were collected. For example, the presence of a user's data in a fine-tuning dataset specific to those subscribed to a particular service leaks that the user subscribed to that service.

*Re-identification.* An adversary may also try to infer the identity of users whose data were used in training without a specific target in mind. This typically involves prompting with various inputs and looking for generated outputs that include identifiable information. For instance, they may prompt a language model with "My email address is: " and observe whether the model produces a plausible email address. If the email address is an actual email address, this might reveal a user's identity.

### 2.2.3 Additional risks

While PETs are best-positioned to address data disclosure and inferred data risks, other risks arise when training AI models. Some concepts create challenges that fall under the broader scope of privacy, but are not usually mitigated by PETs (such as model hallucinations). Others are separate from but deeply related to privacy (such as security). Understanding these distinctions helps us make better decisions about what mitigations make sense.

*Hallucinations.* AI models can hallucinate, making up information about the world which is untrue and not grounded in reality. An incorrect inference about a user might still create a negative user experience and can undermine user trust as much as a correct one. For example, a user may ask: "Tell me more about a friend of mine." If the model generates content that looks legitimate, but is ultimately invented, a user may mistakenly conclude that they are learning legitimate information about someone.

*Correct inferences about individuals.* An AI system is capable of making correct inferences about individuals, without ever seeing their data. For example, a user may ask: "If a person smokes a pack of cigarettes every day, are they likely to get lung cancer?" The model might say "yes," and it might further be the case that this user smokes a pack of cigarettes a day and is likely to be diagnosed with lung cancer. Whether this type of inference is a privacy violation is highly context-dependent and is often not mitigated by simply deploying a PET.

*Security.* Security aims to stop external actors from using a system in a manner unintended by its designers and protects against many threats[12]. Security protections are fundamental building blocks of privacy as they provide technical foundations to achieve privacy protections. For example, security protections may require users to authenticate themselves, making it harder to run adversarial data leakage attacks. Privacy protections further enhance security by addressing privacy-specific risks and can help implement security protections in privacy-preserving ways. For example, a platform may retain user logs to prevent fraud but can mitigate the potential damage from a data breach by retaining logs only for short periods of time (following the privacy principle of data minimization).

*Other types of trust*. As technology platforms play an increasingly important role in our lives, users expect other types of trust. For example, we might desire that platforms do not spread misinformation or illegal content. While these are all important for overall trust, they are often mitigated by technologies other than PETs.

[12] saif.google

## 2.3 PETs

PETs[13] [14] are a diverse set of tools and methods that enable the use of data by reducing privacy risks of data handling. PETs can mitigate risk at any stage of the AI development lifecycle, including data collection, model training, and model deployment. Rapid technical innovations in the fields of AI and privacy interact to jointly drive innovation for PETs, leading to novel technologies for protecting user privacy in cutting edge products.

There is a large variety of PETs that focus on addressing different privacy risks in different ways. For example, many PETs aim to prevent users from being identified in a dataset. Some rely on data obfuscation techniques to reduce the precision of information in a dataset and make it harder to infer who a user may be (for example, data scrubbing). Others rely on perturbing some data selectively to achieve some technical guarantee about users (for example, differential privacy). Still others may not focus on user identification at all. For example, PETs focused on deployment reduce the scenarios when users are not in control of their own data (for example, on-device learning), therefore reducing the potential for privacy incidents.

As a class of technology, PETs also benefit immensely from innovation. For example, synthetic data is a PET that has regained traction in the last few years with the emergence of generative AI (genAI), as genAI suddenly enabled novel approaches to synthetic data generation[15]. Another example is the trusted execution environment, which has been considered for decades but is now benefiting from the tremendous progress in hardware designs that have made it possible to provide technical guarantees about computation[16].

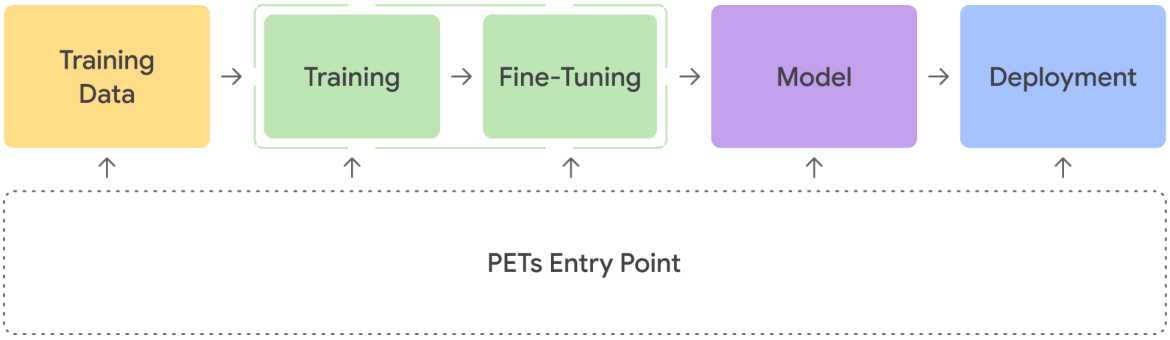


*Figure 4: PETs may be applied at each stage of the AI development process*

[13] weforum.org/publications/the-next-generation-of-data-sharing-in-financial-services-using-privacy-enhancing-techniques-to-unlock-new-value/
[14] petsymposium.org/
[15] research.google/blog/protecting-users-with-differentially-private-synthetic-training-data/
[16] cloud.google.com/confidential-computing/docs/confidential-computing-overview

We can evaluate the privacy risks in any model and system, regardless of whether a PET was used to create it[17] [18] [19]. This can be useful for evaluating the effectiveness of any PET used in the AI development workflow.

Given the variety of PETs and their rapid innovation, it can be challenging to compile a complete list of options. In this section, we outline a non-exhaustive list of PETs for each stage in the AI development lifecycle (Figure 4).

**Data**. At this stage, most PETs focus on preprocessing data to reduce sensitive information within a data set prior to using that data set for training. The main goals of PETs in this category are to 1) reduce the chance that data contains sensitive information, and 2) make it possible to safely share and access datasets from different parties involved in the AI supply chain. Both training data and fine-tuning data may be preprocessed before use.

- Data deduplication: Ensure data items belonging to a single user appear only once in datasets to reduce overrepresentation.
- Data scrubbing: Remove data items such as Personally Identifying Information (PII).
- Data generalization: Rewrite certain data items to make them less specific.
- Data anonymization: Transforms a dataset to mask a user's presence in that dataset. This includes techniques like k-anonymity, which aggregates and removes data items that don't appear a minimum number of times (k). Differentially private algorithms can also rewrite data items to limit how much is revealed about the presence or absence of certain events[20].
- Synthetic data generation: Privacy-preserving synthetic data is a technique that creates a dataset mimicking an original dataset while not including sensitive data. Technologies like differential privacy can ensure the degree to which sensitive data is hidden.

**Training**. PETs can be deployed by modifying the training procedure to reduce privacy risks. These modifications may be made either to training, fine-tuning, or both. The main goals of PETs in this category are to 1) ensure that the training algorithm itself protects user privacy, and to 2) find ways for training algorithms to operate on data that may contain sensitive information.

- AI design selection: Choose an algorithm known to be less prone to sensitive data disclosure or sensitive inference.

[17] arxiv.org/abs/2202.07646
[18] proceedings.neurips.cc/paper/2020/hash/fc4ddc15f9f4b4b06ef7844d6bb53abf-Abstract.html
[19] ieeexplore.ieee.org/abstract/document/9152761
[20] arxiv.org/abs/2304.06929

- Training hyperparameter adjustments: Control how models learn from data, balancing the ability to learn details (preventing underfitting) with ability to generalize (preventing overfitting). This balance is important for both accuracy and protecting privacy by avoiding disclosing specific training examples.
- Private training: Adjust training algorithms to incorporate privacy clauses that prevent learning too much about certain data items, for example, by adding noise to the training process using algorithms such as DP-SGD[21].
- Federated learning: Use distributed learning methods to train on local data, limiting the training data that needs to be shared. For example, with federated learning, user data can be kept on mobile devices, which provide model updates to a centralized server.
- Multi-party Computation (MPC): Rather than combining data in one place, different parties can jointly train over their data while keeping their data private from each other. In MPC, parties refer to different entities which may not trust each other, like different individuals, or different corporations.
- Secure Computing: Adjust the training methods to incorporate security solutions such as access controls, logging, and trusted execution environments to reduce chances of unauthorized access to training data.

**Deployment**. PETs may also be applied at the model deployment stage to mitigate inference time privacy risks. The main goal of PETs in this category is to reduce the exposure of user data in AI workflows.

- Content filtering: Catch user prompts that may cause AI models to leak training data, generate sensitive inferences, or otherwise return harmful results to the user.
- Confidential Computing: Run models in secure environments, such as a trusted execution environment (TEE) that provide hardware-level guarantees, making certain attacks harder. This also offers the ability to provide technical guarantees to users about the processing of their data (for example, attestation that the operator of the server does not look at the model input/output).
- On-device: Run AI models locally on a user's device to remove the need to send input/output to/from a server.
- Multi-Party Computation: AI models receive input in the form of encrypted data, process it, and generate encrypted results that only the end user can access. This ensures user data is never visible to the AI deployment infrastructure. This includes partially Homomorphic Encryption (supporting limited operations on encrypted data) as well as Fully Homomorphic Encryption (supporting all operations).
- Contextual integrity: Instruct AI models, especially those that can make decisions on behalf of users (like Agents), to operate in a way that matches the user context[22].

[21] arxiv.org/abs/1607.00133
[22] arxiv.org/pdf/2408.02373

In practice, there's no one-size-fits-all approach to using PETs. Implementing them involves strategically combining different methods and making trade-offs between privacy guarantees, computing performance, and usability. Furthermore, the protections they offer range from best-effort measures to strict mathematical guarantees. Their effectiveness can also be configured with parameters and may rely on specific assumptions about how users interact with the system. Ultimately, the responsibility falls on the product owners to weigh all these factors and tailor a privacy strategy that is appropriate for their specific needs. These challenges define the decision space when deploying PETs.

# 3. Deploying PETs

In this section, we review common challenges in selecting and using PETs to mitigate privacy risks in AI systems. Like any tool, different PETs may have conflicting requirements, so it is unlikely they can all be used at once. Deciding where and how to deploy PETs is not just about evaluating a zero-sum trade-off between privacy and utility, but is about making a considered choice that reflects a product's vision and values. It requires identifying likely risks, determining how to incorporate PETs into the technical design, and building a helpful product.

## 3.1 Challenges

Several challenges may affect the adoption of PETs in practice. We highlight key implementation challenges: cost effectiveness, level of privacy, and level of utility. However, there may be other aspects that we do not consider here, such as regulatory uncertainty[23].

**Cost effectiveness**. Like any software project, PETs may incur various types of computational, communication, and development costs. All these costs are visible to end users as they may slow down the product experience, and/or increase the price of a product offering.

Computational cost refers to the amount of computing power required. For example, fully homomorphic encryption offers strong guarantees, but requires immense computing power, so fully homomorphic encryption should be considered for specific use cases after careful evaluation of performance aspects.

Communication cost refers to the amount of data exchanged between devices. For example, a simple implementation of Private Information Retrieval (PIR) might download relevant data on a device before making a selection locally, which may be impractical for web search.

[23] services.google.com/fh/gumdrop/preview/misc/pets_whitepaper.pdf

Finally, development cost refers to the engineering time required. Some PETs may involve complex algorithms, requiring novel implementations, dramatically increasing the level of complexity. For example, trusted execution environments require specialized hardware and system architectures. Deploying differentially private algorithms in novel use-cases often requires careful statistical analysis at the cutting edge of research.

Note that the additional costs of implementing PETs may require extra attention in the AI space. Some PETs may have limited cost when applied to small machine learning models or to statistical computations. However, the scale of computation required for other AI systems may balloon this cost.

**Level of utility**. In traditional AI, utility is often measured by model accuracy — a combination of precision (how often is the model right) and recall (of all the things the model should have predicted, how many did it find). It's important to note that measuring utility in genAI models is often not a well-defined task. The vast amount of use cases around genAI make it difficult to define one single metric that captures all the experiences that users are seeking when they use AI. For some use cases, we might strive for factuality (a user asks an LLM a well-defined question); in other use cases a good response has the correct tone (a user asks an LLM to write a letter); in other use cases a good response might require aesthetic evaluation (a user asks an image model to generate a particular picture). For example, if a model is prompted to generate an image of a blue butterfly on a yellow flower, we may consider whether the model successfully generates a blue butterfly on a yellow flower as a notion of accuracy; however, there are elements to the generation that may be implicit (such as a flower having a stem) that may not be explicitly described by the prompt.

Another aspect of utility is the use cases that can be implemented. It's a misconception that PETs work against data utility. In reality, PETs expand utility by enabling valuable applications that are typically blocked by data sensitivity concerns. For example, technologies like differentially private synthetic data make sensitive datasets available for model training, while trusted execution environments allow for secure analysis of sensitive information. Although some PETs might involve a small trade-off in a model's accuracy for one specific task, the impact is relatively minor compared to the use cases PETs enable. In fact, some methods (like DP-SGD) can even improve a model's statistical robustness.

**Types of privacy guarantees**. The privacy guarantees promised by PETs are often incomparable with one another in any formal sense. It boils down to identifying privacy risks of concern and finding the right PET to address them. For example, differential privacy gives a strict mathematical guarantee about privacy, allowing different differentially private algorithms to be compared with one another. However, k-anonymity, which also provides non-trivial

privacy protections to users, cannot usually be understood in terms of the formal guarantees of differential privacy. The same can be said of trusted execution environments compared to k-anonymity, or private information retrieval compared to differential privacy.

Additionally, PETs often have tuning parameters that impact the degree to which they mitigate privacy risks: differential privacy has the parameter epsilon, k-anonymity has the parameter k, and fully homomorphic encryption has the key length. For many PETs, it is possible to trade off the amount of privacy risk mitigation against utility of results by varying these parameters.

The decision space is complicated further since mitigations can enter into the AI development lifecycle in various places. When selecting a PET, we must decide not only which type of privacy, but also how much and where.

**Incorporating privacy into the product vision.** Beyond the technical challenges of implementing PETs, organizations face questions around which privacy protections are right for their product and how privacy risks are prioritized. Privacy considerations are a necessary part of developing any user-facing product. Product owners must think about what specific privacy risks exist at each stage of the product lifecycle, what privacy expectations users may hold, and how these considerations combine into a cohesive product vision.

Overall, while the richness of this decision space can be overwhelming, we argue that it is also where value can be found. By creatively deploying PETs in the right places, organizations can find points in this solution space that unlock positive user experiences while minimizing risks.

## 3.2 Trade-offs

Given those challenges, product owners must decide which PETs to deploy and where. We can visualize any given PETs deployment in a three-dimensional space defined by the axes of privacy, utility and cost effectiveness (Figure 5). In this visualization, compliance obligations set a minimum threshold on the privacy axis. Beyond that, it is up to organizational priorities to decide on the right deployment.

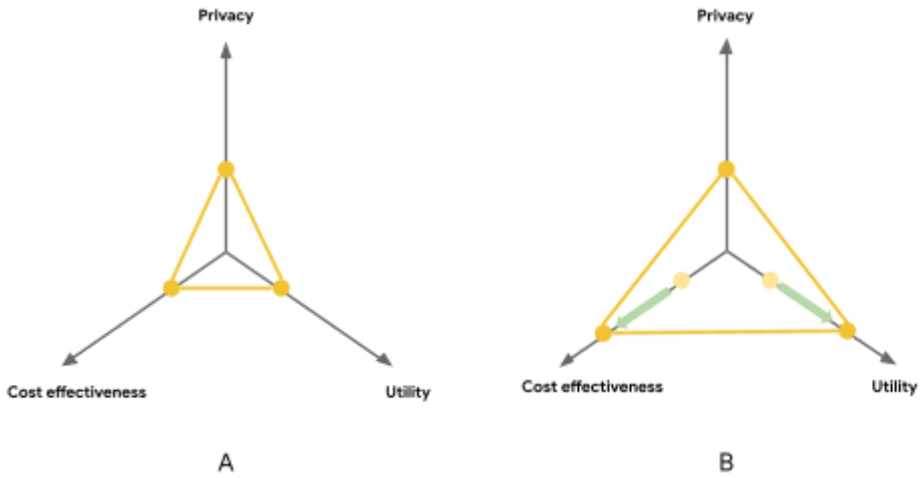


*Figure 5: Visualizing the trade-offs of PETs across privacy, utility, and cost effectiveness. (A) An initial PET deployment may have low cost effectiveness, low utility, and low privacy. (B) As technology improves, the achievable solution space expands, allowing for better outcomes.*

This visualization is meant to help organizations understand the available space of options and find the right trade-off for their products. When exploring options, it is best to consider the whole AI workflow and evaluate the different stages at which PETs may be used. PETs can be layered to achieve a desired balance. It is also possible to test different combinations of PETs and iterate the design over time. Let's walk through a number of examples.

## Example 1: Training AI models on public datasets

A team wants to train an LLM on publicly available data that may contain PII, and release the model via a cloud service without generating responses that could create a privacy incident.

Options:

1. Train the LLM with differential privacy on the public dataset;
2. Train the LLM on a modified version of the public dataset that meets the k-anonymity property;
3. Train the LLM on a modified version of the public dataset with PII filtered out.

Analysis:

- **Risk identification:** The first step is for the team to identify what privacy risks they want to protect against. In this example, even if the likelihood of privacy harm is low as the data is already publicly available, the team may decide that they are concerned about the trained LLM producing PII data.
- **Privacy**: All three options lower privacy risks. Options 1 and 2 offer different kinds of guarantees than option 3 and do not directly address PII. PII filtering (option 3) may make mistakes. Limiting the impact of any single training example (option 1) or removing

data points that do not appear often (option 2) may also limit the extent to which PII influences the trained model.
- **Cost**: Options 1 and 2 incur higher compute and expertise costs than option 3.
- **Utility**: Option 1 and 2 may reduce the amount of data available for training. Option 3 may impact utility, but the impact may be more limited as it is focused on PII.

Conclusion:
In this instance, option 3 may be the most pragmatic choice, provided the team has quality tests to ensure the filtering works well.

## Example 2: Deploying AI models on-device

The same team wants to release the trained model to run on mobile phones without causing user data leaks.

Options:
1. Release the model on the device and run it in a secure enclave (a TEE);
2. Release the model on-device and add input/output filtering mechanisms;
3. Ensure the model does not memorize user data, then release the model on-device with input/output filtering.

Analysis:
- **Risk identification:** The team identifies the risk that an on-device model may be more frequently prompted with less supervision, increasing the chance of AI data leakage by the model.
- **Privacy**: Option 1 (TEE) may make it harder to extract the model but does not address memorization and sensitive data disclosure risks. Option 2 (filtering) helps reduce misbehavior by catching unexpected model inputs and outputs, but is ineffective if the model is extracted from the device. Option 3 directly addresses memorization in addition to maintaining the protections from option 2.
- **Cost**: Option 1 incurs costs to create a TEE. Option 2 requires efforts to develop and deploy device filtering algorithms. Option 3 requires further model-development work.
- **Utility**: Solutions to reduce memorization may impact model utility to varying degrees. Some PETs, like using DP training, may degrade model utility, while other methods, like deduplicating training data, may have a minimal effect on model utility.

Conclusions:
Option 3 may be preferred if the risk of model extraction is sufficiently high and the cost and impact on utility are tolerable, as it provides the most robust privacy protections.

### Example 3: Personalizing genAI models

A team wants to personalize a foundation model for a custom AI experience, but are concerned that the trained model may leak user data.

Options:

1. Train the foundation model directly on user data;
2. Fine-tune the model on user data;
3. Rely on the context window for user data at deployment time;
4. Create individualized models for each user (for example using Low-Rank Adaptation, or LoRA that creates specialized models at inference time).

Analysis:

- **Risk identification:** The team identifies the risk that a personalized model may generate outputs that leak user data.
- **Privacy**: Options 1 and 2 may mix data across users, meaning one user's data could be leaked to another without effective safeguards in place. Options 3 and 4 allow for data provenance and can be implemented so a user's data is never shared with anyone else.
- **Cost**: Option 3 is by far the cheapest as it involves no training or per-user model deployments.
- **Utility**: Each option offers varying utility depending on many factors. The team must carefully measure the utility impact of each approach.

Conclusion:
Options 3 and 4 may be the preferred approaches from a privacy perspective, but the right choice depends on the level of personalization desired and associated costs.

In summary, the effectiveness of PETs hinges on both the context of the AI user experience and the configuration of the AI supply chain. In the next section, we provide guidance to AI practitioners to find the right balance between cost, utility and privacy.

# 4. Guidance for practitioners

Decisions around privacy can be complex and there is no one-size-fits-all solution. However, by following certain best practices, by aspirationally aiming high, an organization can position itself to make the best decisions for its products and users, raise industry standards, and pursue solutions that benefit the whole ecosystem.

When evaluating the use of PETs in products, we should start by considering the balance between privacy protections, user utility, and cost effectiveness. The end goal is to find a

design that meets compliance obligations, is cost effective, and provides a great user experience (Figure 6).

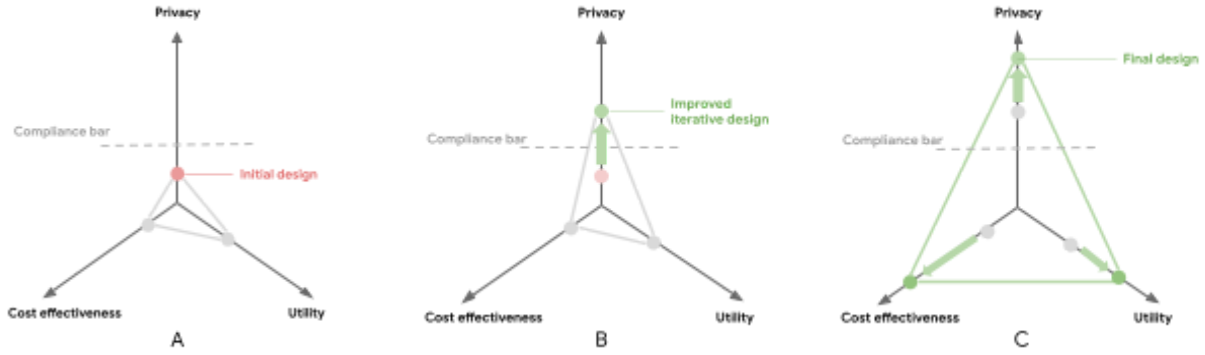


*Figure 6: Illustrative example of an iterative approach to finding innovative privacy solutions that meet compliance and product requirements. (A) An initial design doesn't meet the compliance bar. (B) Innovation through iterative design improvements helps satisfy compliance requirements. (C) The final design iteration improves outcomes across all three dimensions.*

The following guidance is intended to help practitioners achieve this balance.

## 4.1 Incorporate privacy early

"Bolt-on" privacy — developing a system and then reasoning about how to "make it private" — is an antipattern. This approach happens late in the development cycle, when development teams may be less likely to change course and invest effort into deploying a PET. Such an approach can lead to incomplete and ad-hoc privacy solutions.

### 4.1.1 Identify key privacy risks

Before attempting to solve privacy issues, product teams must be clear on the potential privacy risks and which are the most critical for their product. It is ideal to rely on existing lists of risks, such as the SAIF risks[24].

### 4.1.2 Make early privacy decisions

Consider privacy decisions early in the development cycle. When privacy requirements are an organic part of first product design phases, development teams can plan ahead. Protections become part of the requirements that the team uses their skill and creativity to meet.

[24] saif.google

### 4.1.3 Use PETs as early as possible in the AI development lifecycle

It is recommended to use PETs early in the AI development flow. For example, prioritize using a PET on datasets instead of relying only on output filtering. This reduces the overall risk profile by addressing privacy issues as soon as they emerge.

## 4.2 Select the right PETs

There are many approaches to building a privacy-safe product. Some involve good data stewardship like transparency, user consent, data portability controls, data minimization, and deletion. Other techniques are technological, like strong internal access controls, encryption, trusted execution environments, federated learning, differential privacy, k-anonymity, and PII scrubbing.

### 4.2.1 Consider all options

Organizations should make a deliberate decision about which mitigations to use. Avoid a narrow approach that only considers a single approach to mitigate privacy issues. A universal standard (such as requiring k-anonymity of 50 for all data) may be sensible for one deployment and disastrous for another. A narrow policy can create a false sense of security while blocking product development when the PET is a bad match for the context.

Technological innovations in algorithms, training methods, and hardware can alter the tradeoffs for implementing PETs. An advance in one area, such as faster hardware or new AI development, can be leveraged to strengthen privacy protections at reasonable costs. This can transform impractical PETs into viable solutions, unlocking previously nonviable products.

### 4.2.2 Use PETs in conjunction with each other

PETs can be layered to further address potential privacy risks. Google recently released Parfait[25], a GitHub solution that demonstrates state-of-the-art privacy methods and makes it easy for developers to experiment with combining multiple PETs.

### 4.2.3 Take an iterative approach

As features are added and products mature, build in opportunities to re-assess privacy solutions as PETs can also evolve. As a result, it may be possible to improve both privacy and utility simply by using a more advanced PET.

[25] research.google/blog/parfait-enabling-private-ai-with-research-tools/

### 4.2.4 Technology is not a substitute for trust

Implementing a PET is a technical safeguard, not a license to ignore user expectations. The appropriateness of a data use case is as important as the technology protecting it. Organizations should engage with users to ensure the context for using a PET aligns with their expectations.

## 4.3 Demonstrate benefits

There is a need to clarify the benefits obtained through the use of PETs for both end users and the practitioners implementing them.

### 4.3.1 Test PETs

Using PETs can be tricky. There is a risk of misconfiguration that leads to ineffective privacy protection. To mitigate such risks, practitioners should run evaluations to measure the ability of PETs to reduce privacy risks. In other words, PETs should be subject to the same rigor as any other piece of critical infrastructure.

### 4.3.2 Update PETs implementations

Bugs can lead to privacy incidents. Practitioners should monitor the latest version of any available PET and switch to it when it makes sense to their product. For example, the team responsible for PII scrubbing can monitor and benchmark the system's precision-recall curve better than a downstream client team. Similarly, teams responsible for PETs are best equipped to understand when systems are not behaving to specification.

### 4.3.3 Demonstrate privacy improvements

PETs benefit greatly from openness. Organizations should strive to publish research, discuss their use of PETs, and open source their code when possible. External auditing helps correct for internal blind spots. There is no competitive advantage to be earned from using a PET without consensus acceptance from the broader community, so bringing ideas to the community helps drive the next generation of PETs. Google's approach relies on a long track record of building, innovating, implementing and sharing our PETs in research papers, open source projects [26] [27], and Cloud offerings[28].

[26] github.com/google/differential-privacy
[27] pipelinedp.io
[28] cloud.google.com/security/products/dlp

## 4.4 Be pragmatic

Privacy can be complex. The most privacy-safe solution may not be good for the user if it removes major functionality that users love. An overly restrictive approach can incentivise product teams to create their own solutions, forgoing the collective experience of the privacy community and ultimately hurting privacy.

### 4.4.1 PETs are tools for product experiences

PETs should be thought of as tools that help product teams meet their requirements, not as blockers. Rather than worry about what cannot be done, PETs help what can be, and enable the best possible product experience.

### 4.4.2 When possible, don't create your own PETs

If others have demonstrated the positive impact of PETs for a given use case, product teams should consider reusing the same approaches for their product. In the same way that one should not roll out their own cryptographic solutions, one should not roll out their own PETs when there are available ones to rely on.

## 4.5 Establish guidelines

To ensure consistent protection, it is vital to establish clear guidelines for selecting and configuring PETs[29],

### 4.5.1 Make it easy to repeat PETs

When possible, treat PETs as independent horizontal infrastructure. This can mean integrating PETs into commonly used infrastructure for easy access or defining combinations of PETs and configurations for common use cases. These "well-lit paths" can make deployment easier and more predictable. For example, a well-lit path for anonymizing training data might specify thresholding (with a defined context-specific threshold), differential privacy (with a specific privacy budget), and data filtering.

### 4.5.2 Share innovations when possible

Bringing innovations to the community helps drive research and development, inspiring the next generation of PETs. To that end, Google is on a mission to democratize access to our PETs by releasing open source versions. For example, we released the first open source version of

[29] csrc.nist.gov/pubs/sp/800/226/final

our foundational differential privacy libraries[30], a first-of-its-kind Fully Homomorphic Encryption (FHE) transpiler[31], and our work on Federated Learning and secure multi-party computation.

# 5. Conclusion

We have discussed common privacy risks involved in the AI lifecycle and argued that deciding which privacy solution to adopt can be difficult. We challenged the view that utility and privacy must be pitted against each other, proposing instead an approach where product owners make a thoughtful choice about the kind of product they are trying to build. Rather than thinking of privacy as merely a compliance effort, we proposed thinking of it as a creative effort that enables novel user experiences. To support our approach, we introduced a principled framework for selecting privacy solutions rooted in an analysis of cost, utility, and privacy. Finally, we provided recommendations for AI product owners: incorporate privacy early, consider different options, demonstrate benefits, be pragmatic and share results. We highlighted that with the right investment, organizations can differentiate themselves while protecting user privacy.

We hope that our work inspires others to join us in our quest to safely develop AI models with privacy and advance the security of AI systems with CoSAI[32] by encouraging the adoption of PETs[33], creating innovative privacy solutions, and using them in cutting-edge AI products. Looking ahead, we will continue to prototype novel PETs and share the results of our investigations with the world.

# Acknowledgments

We are grateful to the many individuals who helped make this paper possible through their feedback and suggestions: Katherine Lee (for her sustained work on this paper while at Google), Kara Olive, Sergei Vassilvitskii, Yurii Sushko, Daphne Ippolito, Sheri Ringland, Mariana Raykova, Amin Charaniya, Andrew Over, David LaBianca, Bryant Gipson, Ari Levenfeld, Sarah Holland, Ryan Woo, Kristie Chon Flynn, Four Flynn, Dave Kleidermacher, Evan Kotsovinos, Jeffrey Korn, Amanda Walker, Subhash Sankuratripati, Priya Jhakra, Austin Tarango, Ben Kamber, John Stone, Lanah Kammourieh Donnelly, Stacy Karol, Kimberly Samra, Karl Ryan, Brice Daniels and Royal Hansen.

[30] https://opensource.googleblog.com/2020/06/expanding-our-differential-privacy.html
[31] https://github.com/google/fully-homomorphic-encryption
[32] https://www.coalitionforsecureai.org/
[33] https://services.google.com/fh/files/misc/pets_whitepaper.pdf