DOI:10.1145/2660766

**Preparing data for public release requires significant attention to fundamental principles of privacy.**

BY ASHWIN MACHANAVAJJHALA AND DANIEL KIFER

# Designing Statistical Privacy for Your Data

IN 2006, AOL RELEASED a file containing search queries posed by many of its users. The user names were replaced with random hashes, though the query text was not modified. It turns out some users had queried their own names, or "vanity queries," and nearby locations like local businesses. As a result, it was not difficult for reporters to find and interview an AOL user[1] then learn personal details about her (such as age and medical history) from the rest of her queries.

Could AOL have protected all its users by also replacing each word in the search queries with a random hash? Probably not; Kumar et al.[27] showed that word co-occurrence patterns would provide clues about which hashes correspond to which words, thus allowing an attacker to partially reconstruct the original queries. Such privacy concerns are not unique to Web-search data. Businesses, government agencies, and research groups routinely collect data about individuals and need to release some form of it for a variety of reasons (such as meeting legal requirements, satisfying business obligations, and encouraging reproducible scientific research). However, they must also protect sensitive information, including identities, facts about individuals, trade secrets, and other application-specific considerations, in the raw data. The privacy challenge is that sensitive information can be inferred in many ways from the data releases. Homer et al.[20] showed participants in genomic research studies may be identified from publication of aggregated research results. Greveler et al.[17] showed smart meter readings can be used to identify the TV shows and movies being watched in a target household. Coull et al.[6] showed webpages viewed by users can be deduced from metadata about network flows, even when server IP addresses are replaced with pseudonyms. And Goljan and Fridrich[16] showed how cameras can be identified from noise in the images they produce.

Naive aggregation and perturbation of the raw data often leave exposed channels for making inferences about sensitive information;[6,20,32,35] for instance, simply perturbing energy readings from a smart meter independently does not hide trends in energy use. "Privacy mechanisms," or algorithms that transform the data to ensure privacy, must be designed carefully according to guidelines set by a privacy definition. If a privacy definition is chosen wisely by the data curator, the sensitive information will be protected.

## » key insights

- **Data snoopers are highly motivated to publicize or take advantage of private information they can deduce from public data.**
- **History shows simple data anonymization and perturbation methods frequently leak sensitive information.**
- **Focusing on privacy design principles can help mitigate this risk.**

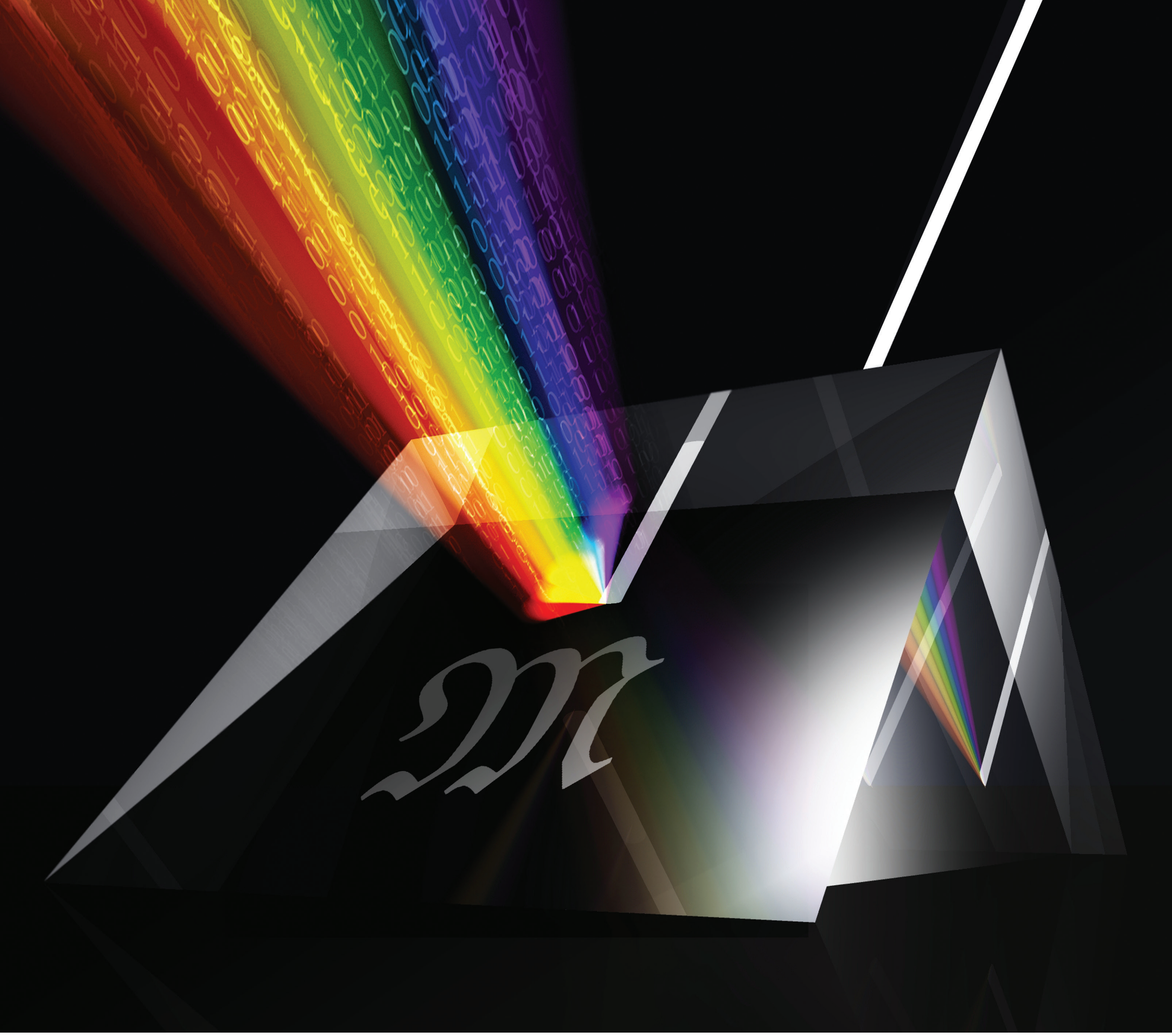


IMAGE BY ANDRIJ BORYS ASSOCIATES/SHUTTERSTOCK

Unfortunately, privacy definitions are not one-size-fits-all. Each application could have its own unique privacy requirements. Working independently, researchers from disparate fields rediscover similar privacy technologies, along with their weaknesses, new fixes, and other vulnerabilities. Our goal here is to synthesize some of the latest findings in the science of data privacy in order to explain considerations and best practices important for the design of robust privacy definitions for new applications. We begin by describing best practices, then explain how they lead to a generic template for privacy definitions, explore various semantic privacy guarantees achievable with this template, and end with an example of a recent privacy definition based on the template and apply it to privacy-preserving *k*-means clustering.

## Desiderata of Privacy Definitions

When data is collected, the curator, with the aid of a privacy definition, puts it in a form that is safe to release. A privacy definition is a specification for the behavior of randomized and deterministic algorithms. Algorithms that satisfy the spec are called privacy mechanisms. The curator first chooses a privacy definition, then a privacy mechanism $\mathfrak{M}$ satisfying the definition. The curator will run $\mathfrak{M}$ on the sensitive data, then grant external users access to the output of $\mathfrak{M}$, or the "sanitized output."

There is a long history of proposed privacy definitions, new vulnerabilities discovered, and amended privacy definitions developed only to be broken once again. As privacy concerns spread, parallel copies of this process are spawned in many research areas. Fortunately, current research has identified many best practices for engineering robust privacy protections for sensitive data. Although they can be formalized in a mathematically rigorous way, we present them at a more intuitive level, leveraging the following privacy definitions as sources of examples.

*Definition 1* ($\in$-differential privacy[9,11]). An algorithm $\mathfrak{M}$ satisfies $\epsilon$-differential privacy if for each of its possible outputs $\omega$ and for every pair of databases $D_1$, $D_2$ that differ on the addition or removal of a single record, $P(\mathfrak{M}(D_1) = \omega) \leq e^{\epsilon} P(\mathfrak{M}(D_2) = \omega)$.

Intuitively, $\epsilon$-differential privacy guarantees that adding or removing a record from the data will have little effect on the output of a privacy mechanism $\mathfrak{M}$. For small $\epsilon$, it means $\mathfrak{M}$ will probably produce the same sanitized output regardless of whether or not Bob's record is in the data.

How should a data curator choose $\epsilon$? Consider a highly targeted query about an individual (such as asking if Bob's record is in the data). For $\in$-differential privacy, the most revealing privacy mechanism answers truthfully with probability $e^{\epsilon}/(1 + e^{\epsilon})$ and falsely with probability $1/(1 + e^{\epsilon})$.[24] When $\epsilon$ is close to 0, both these probabilities are close to ½, and little information is provided; the mechanism is almost as likely to lie as respond truthfully; for example, when $\epsilon = 0.1$, the true answer probability is ≈ 0.525, and when $\epsilon = 0.01$, the probability is ≈ 0.502. We recommend choosing $\epsilon$ based on how close the curator wants this value to be to ½.

The Laplace mechanism is a popular mechanism for $\epsilon$-differential privacy. Let *f* be a function that computes a vector of query answers on the data. To each query answer, the Laplace mechanism adds an independent Laplace random variable with mean 0 and standard deviation $\sqrt{2}S(f)/\epsilon$, where *S*(*f*) is the global sensitivity of $f$– the largest possible change in *f* due to the addition of one record, or the maximum of $||f(D_1) - f(D_2)||_1$ over pairs of databases $D_1$, $D_2$ that differ in one record. Intuitively, the noise masks the influence of any single record on the result of *f*. Now consider:

*Definition 2* (*k*-anonymity.[34,35]) Given a set *Q* of attributes, known as the quasi-identifier, a table is *k*-anonymous if every record in it has the same quasi-identifier values as *k*–1 other records. An algorithm satisfies *k*-anonymity if it outputs only *k*-anonymous tables.

*K*-anonymity defends against one type of attack called a "linkage attack"—joining an external dataset that associates an identity (such as name) with the quasi-identifier (such as ZIP code and age) to a *k*-anonymous table containing this publicly available quasi-identifier. Its goal is to prevent the matching of an identity to a single tuple in the *k*-anonymous table; clearly, there will always be at least *k* candidates in the join result. *K*-anonymity mechanisms usually operate by coarsening attributes (such as dropping digits from ZIP codes and changing ages to age ranges); see Figure 1 for two examples of *k*-anonymous tables.

**Figure 1. Examples of *k*-anonymity: (a) 4-anonymous table; (b) 3-anonymous table.**

| Zip Code | Age | Disease |
|---|---|---|
| 130** | 25–30 | None |
| 130** | 25–30 | Stroke |
| 130** | 25–30 | Flu |
| 130** | 25–30 | Cancer |
| | | |
| 902** | 60–70 | Flu |
| 902** | 60–70 | Stroke |
| 902** | 60–70 | Flu |
| 902** | 60–70 | Cancer |

(a)

| Zip Code | Age | Disease |
|---|---|---|
| 130** | < 40 | Cold |
| 130** | < 40 | Stroke |
| 130** | < 40 | Rash |
| | | |
| 1485* | ≥ 40 | Cancer |
| 1485* | ≥ 40 | Flu |
| 1485* | ≥ 40 | Cancer |

(b)

## Security Without Obscurity

The process of sanitizing sensitive data through a privacy mechanism $\mathfrak{M}$ must follow Kerckhoffs's principle[21] and ensure privacy even against adversaries who might know the details of $\mathfrak{M}$, except for the specific random bits it may have used. Better yet, the mechanism $\mathfrak{M}$ must be revealed along with the sanitized output.

The reasons for making the mechanism $\mathfrak{M}$ publicly known are twofold: First, history shows "security through obscurity" is unreliable in many applications; and, second, the output of $\mathfrak{M}$ must be useful. This sanitized output often takes the form of a dataset or statistical model and could be intended to support scientific research. In such a case, statistical validity is an important concern, and statistically valid conclusions can be drawn only when scientists know precisely what transformations were applied to the base sensitive data.

Likewise, privacy-mechanism designers should always assume attackers are smarter than they are. Just because the designer of a privacy mechanism cannot deduce sensitive information from the output of a piece of software, an adversary will also fail. A well-engineered privacy definition will overcome these disadvantages, protecting sensitive information from clever attackers who know how $\mathfrak{M}$ operates. We explain how in subsequent sections.

**Post-processing.** A privacy definition determines the mechanisms that data curators can trust to not leak sensitive information. Let $\mathfrak{M}$ be one such mechanism, and suppose $\mathcal{A}$ is some algorithm that can be applied to the output of $\mathfrak{M}$; for example, suppose $\mathfrak{M}$ creates synthetic data from its inputs, and $\mathcal{A}$ builds a decision tree. Let the notation $\mathcal{A} \circ \mathfrak{M}$ denote the composite algorithm that first applies $\mathfrak{M}$ to the sensitive data and then runs $\mathcal{A}$ on the sanitized output of $\mathfrak{M}$.

If $\mathfrak{M}$ is trusted, should this composite algorithm $\mathcal{A} \circ \mathfrak{M}$ also be trusted? Intuitively, the answer is yes. It would be very strange if a data curator released privacy-preserving synthetic data but then claimed building statistical models from this data is a violation of privacy.

A privacy definition is closed under post-processing if $\mathcal{A} \circ \mathfrak{M}$ satisfies the constraints defining the privacy definition whenever $\mathfrak{M}$ does. Differential privacy[11] satisfies this property, but *k*-anonymity does not.[23] Closure under post-processing has two important consequences: First, it ensures compatibility between a privacy definition and Kerckhoffs's principle; for example, some algorithms that satisfy *k*-anonymity are susceptible to a so-called minimality attack.[13,36] For each such *k*-anonymity mechanism $\mathfrak{M}$, it is possible to craft a post-processing algorithm $\mathcal{A}$ that takes the output of $\mathfrak{M}$, undoes some of its data transformations, and outputs a new dataset having records with possibly unique quasi-identifier values that are vulnerable to linkage attacks with external data. That is, the composite algorithm $\mathcal{A} \circ \mathfrak{M}$ does not satisfy the same conditions as $\mathfrak{M}$, or *k*-anonymity, and often reveals sensitive records.

By contrast, suppose an $\epsilon$-differentially private algorithm $\mathfrak{M}$ is applied to the data $D$, and the result $\mathfrak{M}(D)$ is published. Given knowledge of $\mathfrak{M}$, a clever adversary can design an attack algorithm $\mathcal{A}$ and run it on the published data to obtain the result $\mathcal{A}(\mathfrak{M}(D))$. Note $\mathcal{A}(\mathfrak{M}(D))$ is the result of applying the composite algorithm $\mathcal{A} \circ \mathfrak{M}$ to the data $D$. Since $\epsilon$-differential privacy is closed under post-processing, the composite algorithm $\mathcal{A} \circ \mathfrak{M}$ still satisfies $\epsilon$-differential privacy and hence has the same semantics; the output of $\mathcal{A} \circ \mathfrak{M}$ is barely affected by the presence or absence of Bob's (or any other individual's) record in the database.

The second important consequence of closure under post-processing is how a data curator must express privacy definitions. Consider $k$-anonymity and $\epsilon$-differential privacy. By analogy to database-query languages, the definition of $k$-anonymity is declarative; that is, it specifies what we want from the output but not how to produce this output. On the other hand, differential privacy is more procedural, specifying constraints on the input/output behaviors of algorithms through constraints on probabilities (such as $P(\mathfrak{M}(D) = \omega)$). This is no coincidence; in order to achieve closure under post-processing, it is necessary that the privacy definition impose conditions on the probabilities (even when $\mathfrak{M}$ is deterministic) rather than on the syntactic form of the outputs.[22]

**Composition.** We introduce the concept of composition with an example. Suppose the 4-anonymous table in Figure 1 was generated from data from Hospital A, while the 3-anonymous table in Figure 1 was generated by Hospital B. Suppose Alice knows her neighbor Bob was treated by both hospitals for the same condition. What can Alice infer about, say, Bob's private records? Bob corresponds to an anonymized record in each table. By matching ZIP code, age, and disease, Alice can deduce that Bob must have had a stroke. Each anonymized table individually might have afforded Bob some privacy, but the combination of the two tables together resulted in a privacy breach. The degradation in privacy that results from combining multiple sanitized outputs is known as "composition."[14]

*Self-composition.* "Self-composition" refers to the scenario where the sanitized outputs are all produced from privacy mechanisms that satisfy the same privacy definition. Fundamental limits on a privacy definition's ability to withstand composition are part of a growing literature inspired by the results of Dinur and Nissim[7] who showed that the vast majority of records in a database of size $n$ can be reconstructed when $n \log(n)^2$ statistical queries are answered, even if each answer has been arbitrarily altered to have up to $o(\sqrt{n})$ error; that is, a distortion that is less than the natural variation of query answers that an adversary would get from collecting a sample of size $n$ from a much larger population.

**The privacy challenge is that sensitive information can be inferred in many ways from the data releases.**

Despite such negative results that limit the number of times a private database can be queried safely, there can be a graceful degradation of privacy protections, as in the case of $\epsilon$-differential privacy. If $\mathfrak{M}_1, \mathfrak{M}_2, \ldots, \mathfrak{M}_k$ are algorithms such that each $\mathfrak{M}_i$ satisfies $\epsilon_i$-differential privacy, then the combination of their sensitive outputs satisfies $\epsilon$-differential privacy with $\epsilon = \epsilon + \ldots + \epsilon$;[30] more formally, this privacy level is achieved by the algorithm $\mathfrak{M}$ running mechanisms $\mathfrak{M}_1, \mathfrak{M}_2, \ldots, \mathfrak{M}_k$ on the input data and releases all their outputs. The end result thus does not reveal any record deterministically while still satisfying differential privacy but with a linear degradation in the privacy parameter.

Self-composition has another practical benefit—simplifying the design of privacy-preserving algorithms. Complicated mechanisms can be built modularly from simpler mechanisms in the same way software is built from functions. By controlling the information leakage of each component individually, a privacy-mechanism designer can control the information leakage of the entire system. In the case of $\epsilon$-differential privacy, the privacy parameter $\epsilon$ of the final mechanism is at most the sum of the privacy parameters of its components.[4,30]

*Composition with other mechanisms.* The data curator must also consider the effect on privacy when the mechanisms do not satisfy the same privacy definition. As an example,[24,26] consider a database where each record takes one of $k$ values. Let $x_1, x_2, \ldots, x_k$ denote the number of times each of these values appears in the database; they are histogram counts. Let $\mathfrak{M}_1$ be a mechanism that releases the sums $x_1 + x_2, x_2 + x_3, \ldots, x_{k-1} + x_k$. Note $\mathfrak{M}_1$ does not satisfy $\epsilon$-differential privacy. Moreover, the knowledge of any one count $x_i$, combined with the output of $\mathfrak{M}_1$, would reveal all the original counts. Now consider a mechanism $\mathfrak{M}_2$ that adds noise drawn from a Laplace distribution, with variance $2/\epsilon^2$, independently, to each histogram count, so its output consists of $k$ noisy counts $\tilde{x}_1, \ldots, \tilde{x}_2$. Mechanism $\mathfrak{M}_2$ does satisfy $\epsilon$-differential privacy;[9] it is the Laplace mechanism mentioned earlier.

What is the effect of the combined release of the sanitized outputs of $\mathfrak{M}_1$ and $\mathfrak{M}_2$? From $\tilde{x}_1$, we have a noisy estimate of $x_1$. From the quantity $x_1 + x_2$ and the noisy value $\tilde{x}_2$, we can obtain another independent estimate of $x_1$. Combining $x_1 + x_2$, $x_2 + x_3$, and $\tilde{x}_3$ we get yet another estimate. Overall, there are $k$ independent noisy estimates of $x_1$ that can be averaged together to get a final estimate with variance $2/(k\epsilon^2)$, which is $k$ times lower than what we could get from $\mathfrak{M}_2$ alone. This example illustrates why there is a recent push for creating flexible privacy definitions that can account for prior releases of information (such as the output of $\mathfrak{M}_1$) to control the overall inference.[2,15,25]

**Convexity.** Consider a privacy definition satisfied by two mechanisms, $\mathfrak{M}_1$ and $\mathfrak{M}_2$. We can create an algorithm $\mathfrak{M}^{(p)}$, or their "convex combination," that randomly chooses among them; with probability $p$ it applies $\mathfrak{M}_1$ to its input and with probability $1-p$ it applies $\mathfrak{M}_2$. Why consider mechanisms like $\mathfrak{M}^{(p)}$? Convex combinations like $\mathfrak{M}^{(p)}$ could provide better worst-case error guarantees for some queries than either $\mathfrak{M}_1$ or $\mathfrak{M}_2$ for reasons similar to why mixed strategies may be preferred over pure strategies in game theory.

Now, should we trust $\mathfrak{M}^{(p)}$ to protect privacy? It is reasonable to do so because the only thing $\mathfrak{M}^{(p)}$ does is add additional randomness into the system.[22,23] We say a privacy definition is convex if every convex combination of its mechanisms also happens to satisfy that privacy definition. Convex privacy definitions have useful semantic properties we discuss in more detail in the next section.

**Minimizing probabilistic failure.** Consider a private record that can be expressed in one bit; that is, 1 if Bob

A naive implementation of a privacy-preserving algorithm may not satisfy the requirements of a chosen privacy definition.

has cancer and 0 otherwise. We add noise from a standard Gaussian distribution and release the result, which happens to be 10. If Bob's bit is 1, then we are 13,000 times more likely to observe a noisy value of 10 than if Bob's bit is 0. We have thus almost certainly discovered the value of Bob's bit.

One can argue that observing a noisy value this large is so unlikely (regardless of the value of Bob's bit) that such a privacy breach is very rare and hence can be ignored. Such reasoning has led to relaxations of privacy definitions that allow guarantees to fail with a small probability $\delta$.; one example is the relaxation $(\epsilon, \delta)$-differential privacy, which can produce more accurate data-mining results.

*Definition 3* $(\epsilon, \delta)$-differential privacy.[10,11] Let $\mathfrak{M}$ be an algorithm and $\mathcal{S}$ be its set of possible outputs. $\mathfrak{M}$ satisfies $(\epsilon, \delta)$-differential privacy if for all subsets $\mathcal{B} \subset \mathcal{S}$ and for all pairs of databases $D_1$, $D_2$ differing on the value of a single record, $P(\mathfrak{M}(D_1) \in \mathcal{B}) \le e^{\epsilon} P(\mathfrak{M}(D_2) \in \mathcal{B}) + \delta$.

The decision whether to always provide guarantees or allow privacy protections to fail with a small probability is application-specific and depends on the stakes involved. It is a privacy/utility trade-off having consequences with different levels of subtlety. For instance, let $\mathfrak{M}$ be the algorithm that outputs $\perp$ with probability $1-\delta$. and outputs the input dataset with probability $\delta$. This $\mathfrak{M}$ satisfies the more relaxed conditions of $(\epsilon, \delta)$-differential privacy. Similarly, consider an algorithm $\mathfrak{M}^*$ that returns the record of a randomly selected individual from the input dataset. If the number of records is $N$ and if $N > 1/\delta$ then $\mathfrak{M}^*$ satisfies $(\epsilon, \delta)$-differential privacy yet always violates the privacy of some individual.

Worth noting is that $\epsilon$-differential privacy and the relaxed $(\epsilon, \delta)$-differential privacy both offer the same high-level guarantee—the output distribution of a mechanism is barely affected by the value of any individual record. Still, privacy relaxation may consistently cause privacy violations, while the former will not. Reasoning about attackers can help data curators set parameter values that limit such information leakages[14] and (as we discuss later) provide new perspectives on achievable guarantees.

**Implementation concerns.** As with many aspects of security, moving from theory to practice requires great care. In particular, a naive implementation of a privacy-preserving algorithm may not ensure privacy even though the algorithm is theoretically proven to satisfy the requirements of a chosen privacy definition. One problem arises from side-channels. Consider a program that processes a sensitive database and behaves as follows: If Bob's record is in the database, it produces the output 1 after one day; if Bob's record is not in the database, it outputs 1 right away. The output is the same, no matter what the database is. But by observing the time taken for a result to be output, we learn something about the database.[18]

Another concern arises when the theoretical algorithms base their security properties on exact computation that may be beyond the limits of digital computers.[5,31] The most common example is the addition of noise from continuous distributions (such as Gaussian and Laplace). For most floating-point implementations, an analysis of the bit patterns yields additional information about the input data.[31]

Finally, many privacy mechanisms rely on a random number generator. A provably secure implementation of a privacy-preserving algorithm must be tailored to the quality of the randomness of the bits.[8]

## A Generic Recipe

Privacy definitions that are convex, closed under post-processing, and require protections for all outputs of a mechanism $\mathfrak{M}$ all have a similar format and can be written in terms of linear constraints,[28] as in the following generic template:

*Definition 4* (a generic privacy definition). Let $D_1, D_2, \ldots$ be the collection of possible input datasets. For some fixed constants $S_1^{(1)}, S_1^{(2)}, \ldots, S_2^{(1)}, S_2^{(2)}, \ldots, S_3^{(1)}, S_3^{(2)}, \ldots$ an algorithm $\mathfrak{M}$ must satisfy the following conditions for every possible output the algorithm $\omega$ can produce:

$$\sum_j S_j^{(1)} P(\mathfrak{M}(D_j)=\omega) \leq 0; \quad \sum_j S_j^{(2)} P(\mathfrak{M}(D_j)=\omega) \leq 0$$
$$\sum_j S_j^{(3)} P(\mathfrak{M}(D_j)=\omega) \leq 0; \qquad \cdots \text{ etc.}$$

To evaluate a proposed privacy definition, a good sanity check for current best practices is thus to verify whether or not the algorithm $\mathfrak{M}$ can be expressed as linear constraints on the probabilistic behavior of algorithms, as in Definition 4; for example, *k*-anonymity does not fit this template,[28] but with $\epsilon$-differential privacy, there is a linear constraint $P(\mathfrak{M}(D_{j1}) = \omega) - e^{\epsilon} P(\mathfrak{M}(D_{j2}) = \omega) \leq 0$ for every pair of datasets $D_{j1}$, $D_{j2}$ that differ on the presence of one rec-ord. We next discuss some semantic guarantees achievable through this template.

**Good and bad disclosures.** Even when published data allows an analyst to make better inferences about Bob, Bob's privacy has not necessarily been violated by this data. Consider Bob's nosy but uninformed neighbor Charley, who knows Bob is a life-long chain-smoker and thinks cancer is unrelated to smoking. After seeing data from a smoking study, Charley learns smoking causes cancer and now believes that Bob is very likely to suffer from it. This inference may be considered benign (or unavoidable) because it is based on a fact of nature.

Now consider a more nuanced situation where Bob participates in the aforementioned smoking study, the data is processed by $\mathfrak{M}$, and the result $\omega$ (which shows that smoking causes cancer) is published. Charley's beliefs about Bob can change as a result of the combination of two factors: by him learning that smoking causes cancer, and since Bob's record may have affected the output of the algorithm. This latter factor poses the privacy risks. There are two approaches to isolate and measure whether Charley's change in belief is due to Bob's record and not due to his knowledge of some law of nature—"counterfactuals"[12,25,33] and "simulatability."[2,15,29]

**Privacy via counterfactuals.** The first approach[12,25,33] based on counterfactual reasoning is rooted in the idea that learning the true distribution underlying the private database is acceptable, but learning how a specific individual's data deviates from this distribution is a privacy breach.

Consider pairs of alternatives (such as "Bob has cancer" and "Bob is healthy"). If the true data-generating distribution $\theta$ is known, we could use it to understand how each alternative affects the output of $\mathfrak{M}$ (taking into account uncertainty about the data) by considering the probabilities $P_\theta(\mathfrak{M}$ outputs $\omega$ | Bob has cancer) and $P_\theta(\mathfrak{M}$ outputs $\omega$ | Bob is healthy). Their ratio is known as the "odds ratio." It is the multiplicative factor that converts the initial odds of Bob having cancer (before seeing $\omega$) into the updated odds (after seeing $\omega$). When the odds ratio is close to 1, there is little change in the odds, and Bob's privacy is protected.

Why does this work? If the reasoning is done using the true distribution, then we have bypassed the change in beliefs due to learning about laws of nature. After seeing $\omega$, the change in Charley's beliefs depends only on the extent to which $\omega$ is influenced by Bob (such as it was computed using Bob's record).

What if the true distribution is unknown? To handle this scenario, the data curator can specify a set $\Xi$ of plausible distributions and ensure reasoning with any of them is harmless; the corresponding odds ratios are all close to 1. A counterfactual-based privacy definition would thus enforce constraints like $P_\theta(\mathfrak{M}$ outputs $\omega$ | Bob has cancer) $\leq e^{\epsilon} P_\theta(\mathfrak{M}$ outputs $\omega$ | Bob is healthy) for all possible $\omega$, for various pairs of alternatives and distributions $\theta$. When written mathematically, these conditions turn into linear constraints, as in the generic template (Definition 4).

**Privacy via simulatability.** The second approach,[2,15,29] based on simulatability, is motivated by the idea that learning statistics about a large population of individuals is acceptable, but learning how an individual differs from the population is a privacy breach. The main idea is to compare the behavior of an algorithm $\mathfrak{M}$ with input $D$ to another algorithm, often called a "simulator," $\mathcal{S}$ with a safer input $D'$; for example, $D'$ could be a dataset that is obtained by removing Bob's record from $D$. If the distribution of outputs of $\mathfrak{M}$ and $\mathcal{S}$ are similar, then an attacker is essentially clueless about whether $\omega$ was produced by running $\mathfrak{M}$ on $D$ or by running $\mathcal{S}$ on $D'$. Now $\mathcal{S}$ does not know anything about Bob's record except what it can predict from the rest of the records in $D'$ (such as a link between smoking and cancer). Bob's record is thus protected. Similarly, Alice's privacy can be tested by considering different alterations where Alice's record is removed instead of Bob's record. If

$\mathcal{S}$ can approximately simulate the behavior of $\mathfrak{M}$ no matter what the true data $D$ is and no matter what alteration was performed, then every individual record is protected.

Privacy definitions based on simulatability are generally more complex than those based on counterfactuals. To check whether $\mathfrak{M}$ satisfies the definitions, it is often necessary to find the appropriate simulator $\mathcal{S}$. However, in some cases, the privacy definitions can also be expressed using linear constraints, as in the generic privacy definition template.

**Counterfactuals vs. simulatability.** The differences between counterfactual and simulatability approaches depend on the nature of the data that must be protected. When the data records are independent of each other, properties of the data-generating distribution and properties of a population are essentially the same (due to the law of large numbers), in which case both approaches provide similar protection.

*Data correlations.* A difference arises when there is correlation between individuals. First, we consider a scenario when counterfactuals would be more appropriate. Suppose a database contains records about Bob and his relatives. Even if Bob's record is removed, Bob's susceptibility to various diseases can be predicted from the rest of the data because it contains his family's medical history. The general goal of privacy definitions based on simulatability is not to hide this inference but to hide how Bob's actual record differs from this prediction. On the other hand, if we include probabilistic models of how diseases are passed through genetics, then privacy definitions based on counterfactuals will try to prevent predictions about Bob and his family. Intuitively, this happens because the actual family medical history is not a property of the data-generating distribution but of a sample from that distribution. Since the family medical history is correlated with Bob's record, it would allow better predictions about how Bob deviates from the data-generating distribution; hence, it must be protected as well.

Next, we examine a situation where simulatability-based privacy definitions are more appropriate. Consider a social network where many profiles of individuals are public. Private information about individuals is often predictable directly from the public profiles of their friends and contacts.[37] Even if Bob's profile is private, it is easy to collect information that is correlated with Bob. Here, privacy definitions based on simulatability are applicable, allowing data curators to process the social network data with algorithms $\mathfrak{M}$ that create outputs from which it is difficult to tell if Bob's record was used in the computation.

*Data constraints.* One difficulty in designing privacy definitions is accounting for public knowledge of constraints the input database must satisfy. Constraints may correlate the values of different records, arising due to, say, functional dependencies across attributes or prior exact releases of histograms. Correlations arising from constraints provide inference channels attackers could use to learn sensitive information. A privacy definition must thus account for them; for example, while Census data records must be treated confidentially, certain coarse population statistics must, by law, be released exactly in order to determine the number of Congressional Representatives for each state. More generally, if a histogram $H$ of the data has been released exactly, how can a data curator choose a privacy definition, and hence constraints on $\mathfrak{M}$ to account for the information in the histogram so any subsequent data release via $\mathfrak{M}$ is able to ensure privacy? Complete solutions to this problem are open but appear to be easier for approaches based on counterfactuals if we use data-generating distributions that are conditioned on the histogram, or $P(D|H)$.[25] For approaches based on simulatability, there is more of a challenge since data-alteration techniques consistent with previously released information must be developed; recall, they provide the guarantee that an attacker would not be able to reliably determine whether the original dataset or altered dataset was used in the computation. It is important to note, too, that constraints on the input data, and especially those arising from prior releases, can be exploited for better utility.

*Interpretations of differential privacy.* These two approaches for defining semantics for privacy definitions also provide two ways of interpreting $\epsilon$-differential privacy. The simulatability argument shows an algorithm satisfying $\epsilon$-differential privacy provides the following protection: an attacker cannot detect whether $\mathfrak{M}$ was run on the original data or on altered data from which any given record was removed.[2,14] This is true no matter how knowledgeable the attacker is, as long as the data alteration is consistent with what is known about the data; if not, additional leakage can occur, as explained in the earlier discussion on composition with other mechanisms. From a different perspective, the counterfactual argument shows an algorithm $\mathfrak{M}$ satisfying $\epsilon$-differential privacy prevents an attacker from learning how an individual differs from the data-generating distribution precisely when all records are independent.[25]

## Example: Blowfish

We illustrate this discussion with Blowfish,[19] a new class of privacy definitions that follows the generic privacy template in Definition 4. Like differential privacy, Blowfish definitions satisfy a number of desirable properties we outlined earlier, including Kerckhoffs's principle, self-composition, convexity, and closure under post-processing. The privacy goals of Blowfish definitions have both a counterfactual and a simulatability interpretation. In addition to satisfying these properties, Blowfish definitions improve on differential privacy by including a generalized and formal specification of what properties of an individual in the data are kept private and by accounting for external knowledge about constraints in the data. Blowfish thus captures part of the privacy design space. In the rest of this section, we describe how data owners can use Blowfish to customize privacy protections for their applications.

Blowfish definitions take two parameters: privacy $\epsilon$ (similar to differential privacy) and policy $P = (\mathcal{T}, \mathcal{G}, \mathcal{I}_{\mathcal{Q}})$ allowing data curators to customize privacy guarantees. Here, $\mathcal{T}$ is the set of possible record values, $\mathcal{Q}$ is a set of publicly known constraints on the data, and $\mathcal{I}_{\mathcal{Q}}$ is the set of all possible datasets consistent with $\mathcal{Q}$. Specifying $\mathcal{I}_{\mathcal{Q}}$ allows a data curator to create privacy definitions that can compose with prior deterministic data releases, thus avoiding some of the difficulties discussed earlier in the section on desiderata. To simplify the discussion, we set $\mathcal{Q}$ to be the single constraint that the dataset has $n$ records, in which case $\mathcal{I}_{\mathcal{Q}} = \mathcal{T}^n$; for more complicated constraints, see He[19] on Blowfish and Kifer and Machanavajjhala[25] on Pufferfish frameworks.

The final component of the policy is $G = (\mathcal{T}, E)$, or the "discriminative secret graph." The vertices in $G$ are the possible values a record can take. Every edge $(x, y) \in E$ describes a privacy goal with both counterfactual and simulatability interpretations. From the simulatability viewpoint, changing a single record from $x$ to $y$ (or vice versa) will not cause a significant change in the probability of any output. From the counterfactual viewpoint, if records are independent, an attacker could estimate the odds of a new record having value $x$ vs. $y$, but estimated odds about any individual in the data would not differ significantly from this value. Using this graph $G$, we define the concept of neighboring databases, then formally define the Blowfish framework:

*Definition 5* ($G$-Neighbors). Let $P = (\mathcal{T}, G, \mathcal{T}^n)$ be a discriminative secret graph. Two datasets $D_1, D_2 \in \mathcal{T}^n$ are called $G$-neighbors if for some edge $(x, y) \in E$ and some dataset $D \in \mathcal{T}^{n-1}$, $D_1 = D \cup \{x\}$ and $D_2 = D \cup \{y\}$.

*Definition 6* (($\epsilon$, $P$)-Blowfish Privacy). Let $P = (\mathcal{T}, G, \mathcal{T}^n)$ be a policy. An algorithm $\mathfrak{M}$ satisfies ($\epsilon$, $P$)-Blowfish privacy if for all outputs $\omega$ of the algorithm $\mathfrak{M}$ and all $G$-neighbors $D_1$, $D_2$ we have $P(\mathfrak{M}(D_1)=\omega) \leq e^{\lfloor d(x,y)/10 \rfloor \epsilon} P(\mathfrak{M}(D_2)=\omega)$.

This privacy definition clearly matches the generic template of Definition 4. We now examine some policies and their applications.

*Full domain.* Consider a policy $P_K = (\mathcal{T}, G, \mathcal{T}^n)$ where $K$ is a complete graph, and every pair of values in the domain $\mathcal{T}$ are connected. The result is that two datasets are neighbors if they differ (arbitrarily) in any one record. ($\epsilon$, $P_K$)-Blowfish privacy is equivalent to a popular variant of differential privacy[11] that requires $P(\mathfrak{M}(D_1)=\omega) \leq e^{\lfloor d(x,y)/10 \rfloor \epsilon} P(\mathfrak{M}(D_2)=\omega)$ for all $\omega$ and for all pairs of datasets $D_1, D_2$ that differ (arbitrarily) in the value (rather than presence/absence) of one record.

*Partitioned.* Let us partition the domain $\mathcal{T}$ into $p$ mutually exclusive subsets, with $\mathcal{P} = \{P_1, \ldots, P_p\}$. Consider a graph $G^{\mathcal{P}} = (\mathcal{T}, E)$, where any two values $x$, $y$ are connected by an edge if and only if $x$ and $y$ appear in the same partition. Each connected component of $G^{\mathcal{P}}$ is thus a clique corresponding to one of the $P_i$. Now, two datasets $D_1$ and $D_2$ are neighbors if $D_2$ can be obtained from $D_1$ by replacing the value of one record with a new value belonging to the same partition. For example, let $\mathcal{T}$ be the set of all disease outcomes, partitioned into three subsets: healthy cases, communicable diseases, and non-communicable diseases. Let us use the graph $G^{\mathcal{P}}$ corresponding to this partition in our Blowfish policy. An algorithm $\mathfrak{M}$ satisfying Definition 6 comes with the guarantee that the probabilities of its outputs do not change substantially if one communicable disease is replaced with another communicable disease or a healthy case with another healthy case, or a simulatability interpretation.

**Learning population statistics is acceptable, but learning how an individual differs from the population is a privacy breach.**

What about replacing a noncommunicable disease with a communicable disease? Can the algorithm's output probabilities be significantly different in such a case? The answer is yes. In fact, this policy allows algorithms to publish the exact status of each individual—healthy, contagious, or noncontagious—and approximate histograms of each disease. However, specific details (such as which person has which contagious disease) are protected. Such behavior may be desirable in certain health-care applications where some facts must be disclosed but further details kept confidential.

*Distance threshold.* Many applications involve a concept of distance between records; for instance, the distance between two age values can be the absolute difference, and the distance between two points on a plane can be the straightline Euclidean distance or the Manhattan distance along a grid. Given a distance metric $d$, one can define a discriminative secret graph $G^{d,\theta}$ in which only nearby points are connected. That is, $(x, y) \in E$ only when $d(x, y) < 0$ for some threshold $q$; for example, if $\mathcal{T}$ is the set of all points on Earth, and $d$ is the orthodromic distance between pairs of points, we can set $\theta = 10$ miles, so valid record locations are connected to other valid record locations that are within 10 miles of each other. In general, if an individual's location $x$ (in dataset $D_1$) was changed to another point y (resulting in a neighboring dataset $D_2$), then an algorithm satisfying Blowfish with this policy will have the guarantee that for all outputs $\omega$

$$P(\mathfrak{M}(D_1)=\omega) \le e^{\lfloor d(x,y)/10 \rfloor \epsilon} P(\mathfrak{M}(D_2)=\omega)$$

An adversary may thus be able to detect the general geographic region of a target individual but unable to infer the location with a resolution better than 10 miles. Such a relaxed notion of privacy is reasonable when dealing with location data; individuals may not want disclosure of their precise locations but be less worried about disclosing their information at a coarser granularity (that may be obtained from other sources). As we show later, data output by mechanisms that satisfy such relaxed notions of privacy permit data mining results with greater accuracy than if data is generated using mechanisms that satisfy the stricter notion of differential privacy.

*Attribute.* Let $\mathcal{T}$ be a multi-attribute domain with $m$ attributes $\mathcal{T} = A_1 \times A_2 \times \ldots, \times A_m$. Consider a graph $G^{attr,c}$ connecting any two values $x$ and $y$ that differ in at most $c$ attribute values. A Blowfish policy with this graph is useful for location traces and genome data. For the former, attributes correspond to locations of an individual at different times. Neighboring datasets thus differ in at most $c$ locations of a person, hiding the specific details about every sequence of $c$ consecutive locations of an individual. In the genome case, an attribute corresponds to a specific position on the genome. Under this policy, an algorithm's output would be insensitive to changes to a block of up to $c$ positions on the genome.

**Answering queries with Blowfish.** Recall that adding Laplace noise with 0 mean and $\sqrt{2}S(f)/\epsilon$ standard deviation to a function $f$ (where $S(f)$ is the sensitivity of $f$) ensures $\epsilon$-differential privacy. Blowfish, with a policy $P = (\mathcal{T}, G, \mathcal{T}^n)$ is also compatible with additive Laplace noise and requires an often smaller standard deviation of $\sqrt{2}S(f,G)/\epsilon$ where $S(f, G)$ is the policy-specific global sensitivity of $f$—the largest difference $||f(D_1)-f(D_2)||_1$ over all datasets $D_1$, $D_2$ that are $G$-neighbors.

Consider a multidimensional record domain $\mathcal{T} = A_1 \times A_2 \times \ldots, \times A_m$ where each attribute is numeric. Let $q_{sum}$ denote the function that sums all the records together; that is, for each attribute, it computes the sum of the values that appear in the data. Let $a_i$ and $b_i$ denote the maximum and minimum values in attribute $A_i$. The global sensitivity $S(q_{sum})$ of $q_{sum}$ is $\sum_{i=1}^{m} \max\{|a_i|, |b_i|\}$. The policy-specific global sensitivity of $q_{sum}$ under Blowfish policies is usually much smaller. In the case of the distance threshold policy $G^{d,\theta}$ with $d$ being the $L_1$ Manhattan distance, $S(q_{sum}, G^{d,\theta})$ is only $\theta$. Consider a single attribute domain Age and further suppose the age values range from 0 to 100. The global sensitivity of $q_{sum}$ is 100. The policy-specific sensitivity of $q_{sum}$ under $G^{L_1,5}$ is only 5. If, instead, the policy used a partition graph $G^{\mathcal{P}}$ that partitions age into ranges (such as {0 – 10, 11 – 20, 21 – 30,…,91 – 100}), then the policy-specific global sensitivity is only 10. Finally, with the attribute policy, $S(q_{sum}, G^{attr,1}) = \max_i(a_i - b_i)$.

***K*-means clustering.** For a specific data-mining result, consider an application of Blowfish to $k$-means clustering.

*Definition 7* ($K$-means clustering). Given a set of $n$ vectors $\{x_1, \ldots, x_n\}$, the $k$-means clustering problem is to divide these $n$ records among $k$ clusters $S = \{S_1, \ldots, S_k\}$, where $k \le n$, so as to minimize the objective function

$$\sum_{i=1}^{k} \sum_{\mathbf{x}_j \in S_i} ||\mathbf{x}_j - \mu_i||_2^2, \qquad (1)$$

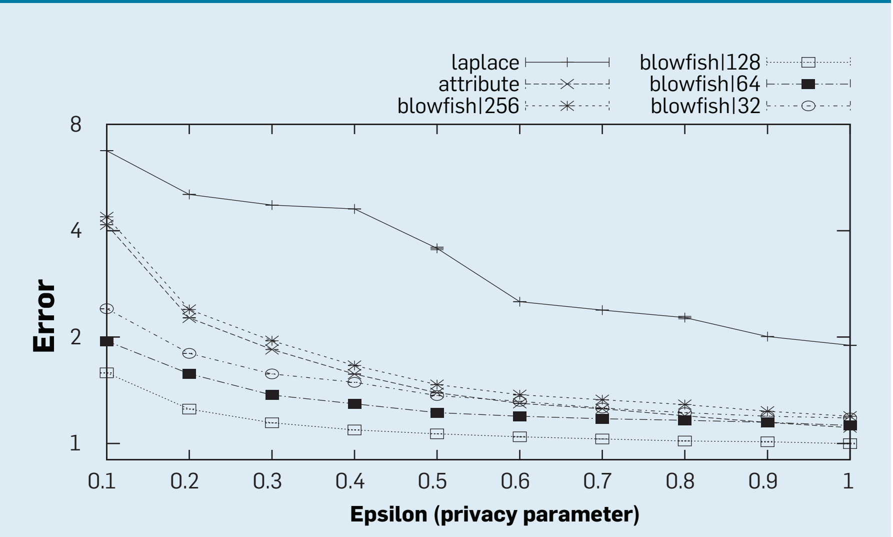
where $\mu_i = \frac{1}{|S_i|}\sum_{\mathbf{x}_j \in S_i} \mathbf{x}_j$ is the mean of cluster $S_i$.

The iterative (non-private) $k$-means clustering algorithm initializes a set of $k$ centroids $\{\mu_1, \mu_2, \ldots, \mu_k\}$, one for each cluster. These centroids are iteratively updated in two steps: assign each $x_j$ to the cluster with the nearest centroid, and set each centroid $\mu_i$ to be the mean of the vectors of its corresponding cluster. The algorithm terminates after a certain number of iterations or when the centroids do not change significantly.

Each iteration (the two steps) are easily modified to satisfy $\epsilon$-differential privacy[4,30] and Blowfish.[19] These steps require access to the answers to two queries: $q_{hist}$, which returns the number of points in each cluster, and $q_{sum}$, or the sum of the points in each cluster. As discussed earlier, $q_{sum}$ can be answered through the Laplace mechanism. Analogously, $q_{hist}$ can be answered with the Laplace mechanism because it has global sensitivity $S(q_{hist}) = 1$ (for differential privacy) and policy-specific global sensitivity $S(f, G) = 2$ for all Blowfish policies discussed here. The policy-specific sensitivity of the $q_{sum}$ query under Blowfish policies is typically much smaller than its global sensitivity so we would thus expect more accurate clustering under the Blowfish privacy definitions.

Figure 2 confirms this improvement in utility. For the clustering task,

**Figure 2. *K*-means under several Blowfish policies.**



we used a small sample of the skin-segmentation dataset,[3] or 1%, which is approximately 2,500 instances, in order to make the problem challenging. Each instance corresponds to the RGB intensities from face images, and each intensity ranges from 0 to 255. The *x*-axis is the privacy parameter $\epsilon$, and on the *y*-axis (note the log scale) we report the error incurred by the privacy-preserving algorithms. We measure the error as the ratio between the squared error (Equation 1) attained by the privacy-preserving algorithms to that achieved by the non-private *k*-means algorithm after 10 iterations that was sufficient for the convergence of the non-private algorithm. The Laplace mechanism for $\epsilon$-differential privacy incurred the most error. Using the $G^{attr,1}$ policy already reduces the error by at least a factor of 1.5. The error is further reduced when using $G^{L_1,\theta}$, for $\theta \in \{256, 128, 64, 32\}$. It is interesting to note the error does not increase monotonically as we increase $\theta$ – $G^{L_1,128}$—an improvement of 3x and 2x over differential privacy for $\epsilon \leq 0.5$ and $\epsilon > 0.5$, respectively. One explanation is that small amounts of error can help avoid local minima while clustering.

## Conclusion

Privacy definitions are formal specifications an algorithm must satisfy to protect sensitive information within data. Our experience shows that designing robust privacy definitions often requires a great deal of subtlety. Our goal is to present some of the major considerations in this design process, along with example privacy definitions and resulting privacy mechanisms. We hope this discussion inspires additional curiosity about the technical nature of privacy.

## Acknowledgment

This work is supported by the National Science Foundation under Grants 1054389 and 1253327.

**References**

1. Barbaro, M. and Zeller, T. A face is exposed for AOL searcher no. 4417749. *The New York Times* (Aug. 9, 2006).
2. Bassily, R., Groce, A., Katz, J., and Smith, A. Coupled-worlds privacy: Exploiting adversarial uncertainty in statistical data privacy. In *Proceedings of the 54th IEEE Annual Symposium on Foundations of Computer Science* (Berkeley, CA, Oct. 27–29). IEEE Computer Society Press, Washington, D.C., 2013, 439–448.
3. Bhatt, R. and Dhall, A. *Skin Segmentation Dataset.* Machine Learning Repository Center for Machine Learning and Intelligent Systems, University of California, Irvine, 2012; https://archive.ics.uci.edu/ml/datasets/Skin+Segmentation/
4. Blum, A., Dwork, C., McSherry, F., and Nissim, K. Practical privacy: The SuLQ framework. In *Proceedings of the 24th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems* (Baltimore, MD, June 13–16). ACM Press, New York, 2005, 128–138.
5. Chaudhuri, K., Monteleoni, C., and Sarwate, A.D. Differentially private empirical risk minimization. *Journal of Machine Learning Research 12* (July 2011), 1069–1109.
6. Coull, S., Collins, M., Wright, C., Monrose, F., and Reiter, M. On Web browsing privacy in anonymized netflows. In *Proceedings of 16th USENIX Security Symposium* (Boston, MA, Aug. 6–10). USENIX Association, Berkeley, CA, 2007, 23:1–23:14.
7. Dinur, I. and Nissim, K. Revealing information while preserving privacy. In *Proceedings of the 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems* (San Diego, CA, June 9–12). ACM Press, New York, 2003, 202–210.
8. Dodis, Y., López-Alt, A., Mironov, I., and Vadhan, S.P. Differential privacy with imperfect randomness. In *Proceedings of the 32nd Annual Cryptology Conference* (Santa Barbara, CA, Aug. 19–23). Springer-Verlag, Berlin, Heidelberg, 2012, 497–516.
9. Dwork, C. Differential privacy. In *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming* (Venice, Italy, July 9–16). Springer-Verlag, Berlin, Heidelberg, 2006, 1–12.
10. Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., and Naor, M. Our data, ourselves: Privacy via distributed noise generation. In *Proceedings of the 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques* (Saint Petersburg, Russia, May 28–June 1). Springer-Verlag, Berlin, Heidelberg, 2006, 486–503.
11. Dwork, C., McSherry, F., Nissim, K., and Smith, A. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Theory of Cryptography Conference* (Columbia University, New York, Mar. 4–7). Springer-Verlag, Berlin, Heidelberg, 2006, 265–284.
12. Evfimievski, A., Gehrke, J., and Srikant, R. Limiting privacy breaches in privacy-preserving data mining. In *Proceedings of the 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems* (San Diego, CA, June 9–12). ACM Press, New York, 2003, 211–222.
13. Fang, C. and Chang, E.-C. Information leakage in optimal anonymized and diversified data. In *Proceedings of the 10th Information Hiding* (Santa Barbara, CA, May 19–21). Springer-Verlag, Berlin, Heidelberg, 2008, 30–44.
14. Ganta, S.R., Kasiviswanathan, S.P., and Smith, A. Composition attacks and auxiliary information in data privacy. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Las Vegas, Aug. 24–27). ACM Press, New York, 2008, 265–273.
15. Gehrke, J., Lui, E., and Pass, R. Towards privacy for social networks: A zero-knowledge-based definition of privacy. In *Proceedings of the Theory of Cryptography Conference* (Providence, RI, Mar. 28–30). Springer-Verlag, Berlin, Heidelberg, 2011, .432-449.
16. Goljan, M. and Fridrich, J. Camera identification from scaled and cropped images. In *Proceedings of Electronic Imaging, Forensics, Security, Steganography, and Watermarking of Multimedia Contents* (Feb. 26, 2008); http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=812538
17. Greveler, U., Justus, B., and Loehr, D. Forensic content detection through power consumption. In *Proceedings of the IEEE International Conference on Communications* (Ottawa, Canada, June 10–15). IEEE Press, Psicataway, NJ, 2012, 6759–6763.
18. Haeberlen, A., Pierce, B.C., and Narayan, A. Differential privacy under fire. In *Proceedings of the 20th USENIX Conference on Security* (San Francisco, CA, Aug. 8–12). USENIX Association, Berkeley, CA, 2011, 33–33.
19. He, X., Machanavajjhala, A., and Ding, B. Blowfish privacy: Tuning privacy-utility trade-offs using policies. In *Proceedings of the ACM SIGMOD/PODS International Conference on Management of Data* (Snowbird, UT, June 22–27). ACM Press, New York, 2014, 1447–1458.
20. Homer, N., Szelinger, S., Redman, M., Duggan, D., Tembe, W., Muehling, J., Pearson, J.V., Stephan, D.A., Nelson, S.F., and Craig, D.W. Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density snp genotyping microarrays. *PLoS Genetics 4*, 8 (Aug. 2008).
21. Kerckhoffs, A. La cryptographie militaire. *Journal des Sciences Militaires 9* (Jan. 1983), 5–83.
22. Kifer, D. and Lin, B.-R. Towards an axiomatization of statistical privacy and utility. In *Proceedings of the 29th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems* (Indianapolis, IN, June 6–11). ACM Press, New York, 2010, 147–158.
23. Kifer, D. and Lin, B.-R. An axiomatic view of statistical privacy and utility. *Journal of Privacy and Confidentiality 4*, 1 (2012), 5–49.
24. Kifer, D. and Machanavajjhala, A. No free lunch in data privacy. In *Proceeding of the ACM SIGMOD/PODS International Conference on Management of Data* (Athens, Greece, June 12–16). ACM Press, New York, 2011, 193–204.
25. Kifer, D. and Machanavajjhala, A. A rigorous and customizable framework for privacy. In *Proceedings of the 31st Symposium on Principles of Database Systems* (Scottsdale, AZ, May 20–24). ACM Press, New York, 2012, 77–88.
26. Kifer, D. and Machanavajjhala, A. Pufferfish: A framework for mathematical privacy definitions. In *Transactions on Database Systems 39*, 1 (Jan. 2014), 3:1–3:36.
27. Kumar, R. Novak, J., Pang, B., and Tomkins, A. On anonymizing query logs via token-based hashing. In *Proceedings of the 16th International World Wide Web Conference* (Banff, Alberta, Canada, May 8–12). ACM Press, New York, 2007, 629–638.
28. Lin, B.-R. and Kifer, D. Towards a systematic analysis of privacy definitions. *Journal of Privacy and Confidentiality 5*, 2 (2014), 57–109.
29. Machanavajjhala, A., Gehrke, J., and M. Götz. Data publishing against realistic adversaries. In *Proceedings of the 35th International Conference on Very Large Data Bases* (Lyon, France, Aug. 24–28, 2009), 790–801.
30. McSherry, F.D. Privacy integrated queries: An extensible platform for privacy-preserving data analysis. In *Proceedings of ACM SIGMOD/PODS International Conference on Management of Data* (Providence, RI, June 29–July 2). ACM Press, New York, 2009, 19–30.
31. Mironov, I. On significance of the least significant bits for differential privacy. In *Proceedings of the 19th ACM Conference on Computer and Communications Security* (Raleigh, NC, Oct. 16–18). ACM Press, New York, 2012, 650–661.
32. Narayanan, A. and Shmatikov, V. Robust de-anonymization of large sparse datasets. In *Proceedings of the 2008 IEEE Symposium on Security and Privacy* (Oakland, CA). IEEE Computer Society Press, Washington, D.C., 2008, 111–125.
33. Rastogi, V., Hay, M., Miklau, G., and Suciu, D. Relationship privacy: Output perturbation for queries with joins. In *Proceedings of the 28th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems* (Providence, RI, June 29–July 2). ACM Press, New York, 2009, 107–116.
34. Samarati, P. Protecting respondents' identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering 13*, 6 (Nov. 2001), 1010–1027.
35. Sweeney, L. *K*-anonymity: A model for protecting privacy. *International Journal of Uncertainty Fuzziness and Knowledge-Based Systems 10*, 5 (Oct. 2002), 557–570.
36. Wong, R., Fu, A., Wang, K., and Pei, J. Minimality attack in privacy-preserving data publishing. In *Proceedings of the 33rd International Conference on Very Large Data Bases* (University of Vienna, Austria, Sept. 23–27). VLDB Endowment, 2007, 543–554.
37. Zheleva, E. and Getoor, L. To join or not to join: The illusion of privacy in social networks with mixed public and private user profiles. In *Proceedings of the 18th International World Wide Web Conference* (Madrid, Spain, Apr. 20–24). ACM Press, New York, 2009, 531–540.

**Ashwin Machanavajjhala** (ashwin@cs.duke.edu) is an assistant professor in the Department of Computer Science at Duke University, Durham, NC.

**Daniel Kifer** (dkifer@cse.psu.edu) is an associate professor in the Department of Computer Science & Engineering at Penn State University, University Park, PA.