Privacy Tools for Data Sharing[1]

Stephen Weis

January 2020

[1] Published on saweis.net

This is an overview about common tools and definitions for anonymizing data or confidentially computing over private data. It is roughly in order of practicality and maturity and is summarized in a table at the end.

Information Reduction Techniques

Minimization refers to collecting the minimal data necessary for a specific purpose. **Redaction** deletes or censors data which has already been collected.

Aggregation reduces a set of individual values to a smaller set of derived values, for example, summing or averaging a list of values into a single number. Training machine learning models or projecting high-dimensional data to lower dimensions are also forms of aggregation.

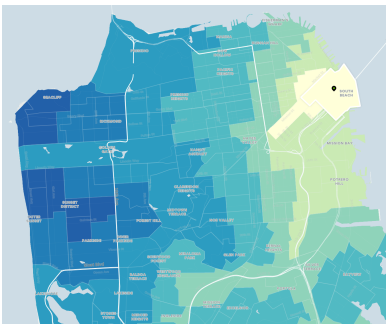

Figure 1: An example map of aggregated geolocation data, binned by neighborhood and time.

Binning or **bucketing** is a common form of aggregation that groups values into buckets or ranges through rounding, truncation, or some other mapping. Examples might include truncating GPS coordinates to 3 decimal places or rounding time values to the nearest 15 minute period.

Top-coding and **bottom-coding** are both types of binning which group together the tail ends of distributions, where there may be few samples. For instance, grouping ages of "Under 20", "20-45", "46-65", "Over 65" uses both bottom- and top-coding for the respective youngest and oldest groups.

One concern with aggregation are **differencing attacks** which compare two aggregated values for different sets and extract the different individual input values. For example, given the average of Alice and Bob's salary (e.g. $50k) and the average of Alice, Bob, and Carol's salary ($60k), you can learn Carol's salary ($80k). Data sets which differ in individual values are called **neighboring data**.

k-Anonymity and Extensions

Implicit in binning values for privacy is the notion of *safety in numbers* – that grouping an individual with others will hide that person's identity. The notion of ***k*-anonymity** [34] is that at least k people will be in any given bin or group. Aggregated data, APIs, or visualization tools often have **minimum thresholds** before returning data, so are

k-anonymous in practice. Google often uses k values of 1000 in their advertising and analytics products. Facebook has used k values of 20 for advertising products. Some government agencies use minimum thresholds of 3 before showing data.

The privacy provided by k-anonymity depends on the underlying data set. One problem is **homogenous data**, where every member of a bin has a common sensitive trait. In Figure 2, we know all 30-40 year olds were diagnosed with cancer.

A second problem is deanonymizing people by joining **background knowledge** to a k-anonymous data set. For example, again in Figure 2, one may know the incidence rate of specific diseases across different nationalities. That could be used to infer the nationality of a given record, which had been redacted.

L**-diversity** [26] is an extension of k-anonymity that addresses the homogeneity issue by ensuring that sensitive traits among bins have at least l distinct values. T**-closeness** [25] is a further refinement that ensures that those distinct values are distributed similarly to the population that data are drawn from. Neither l-diversity nor t-closeness are widely used in practice, though Google offers an l-diversity measurement in their data loss prevention API [20].

	Non-Sensitive			Sensitive
	Zip Code	Age	Nationality	Condition
1	130**	< 30	$*$	Heart Disease
2	130**	< 30	$*$	Heart Disease
3	130**	< 30	$*$	Viral Infection
4	130**	< 30	$*$	Viral Infection
5	1485*	≥ 40	$*$	Cancer
6	1485*	≥ 40	$*$	Heart Disease
7	1485*	≥ 40	$*$	Viral Infection
8	1485*	≥ 40	$*$	Viral Infection
9	130**	$3*$	$*$	Cancer
10	130**	$3*$	$*$	Cancer
11	130**	$3*$	$*$	Cancer
12	130**	$3*$	$*$	Cancer

Figure 2: A 4-anonymous table. Zip codes have been truncated, ages have been top- and bottom-coded, and nationalities have been redacted so that each (zip code, age range) has 4 records associated with it. Source: [26]

Pseudonymization and Tokenization

Pseudonymization or **tokenization** mean replacing sensitive values like names, credit card numbers, or phone numbers with surrogate or token values. These tokens retain their original relationship in the data set and are often used as database lookup keys.

Tokenization often uses a **lookup table with random values** or universally unique identifiers (UUIDs) mapping to original values. This has the strongest security guarantee, but requires storing an entire mapping of the original data to token values.

Pseudorandom functions (PRFs) are functions that take a secret key and arbitrary data as input, and output fixed-length values indistinguishable from randomness. Rather than storing an entire table of random tokens, one can use a PRF to tokenize data with only a secret key. HMACs like HMAC-SHA256 are often used as PRFs in practice.

Format preserving encryption [3] (FPE) entails encrypting sensitive fields such that the output ciphertext has the same format as the input. For example, an format-preserving encrypted credit card could have 16 decimal digits. FPE allows ciphertexts to be stored in existing data fields, like database columns, which may not be easily changed. FPE is also advantageous because, unlike PRFs, the original input data can be decrypted with the secret key. AES FFX mode [14] is one recent standard for FPE encryption.

Historically, **one-way functions** have often been misused for pseudonymization. Many examples of re-identification have occurred when someone naively used a hash function like MD5 or SHA-1 and stored the digests without salting the input. Hash functions are not keyed like PRFs. Rather, anyone can compute them.

The risk of unkeyed hash functions is if the input values have low entropy or are from a known set. That allows someone to conduct a **dictionary attack** where they simple try hashing many values or use precomputed known hash values available online. Developers often try to avoid this by including an ad hoc salt value. In general, simply hashing should be avoided unless the input is known to be high-entropy.

Differential Privacy

Differential privacy defines the amount of information about an individual which could be leaked from a dataset [13]. Differential privacy is used by the US Census [17], Apple [1], Google [22], Facebook [29], and Uber [35] to protect research and user data.

Fundamentally, achieving differential privacy means adding some form of noise or randomization while sacrificing accuracy. In its most basic form, differential privacy specifies a numerical bound, ϵ, on how much an algorithm's output can change between two neighboring datasets that only differ in a single element. Part of the value of differential privacy is that ϵ-private systems can be composed with easily understood properties. This lets one set a **privacy budget** and allow queries until that budget is exhausted.

A **privacy mechanism** is an implementation that achieves a target ϵ-privacy. A simple mechanism is **randomized response**, which was designed to allow people to answer surveys about sensitive topics. The idea is that you flip a coin *Heads* to either answer honestly (e.g. *"Did you cheat on your taxes?"*) or *Tails* to commit to the answer with negative connotations (*"Yes"*). This gives the survey taker plausible deniability that they simply flipped a *Tails*. It is also easy to subtract the estimated noise and approximate the true answer rate.

Another privacy mechanism is the **Laplace Mechanism**, which adds randomly sampled noise from the Laplace distribution, shown in Figure 3. This is a simple mechanism that is tunable to achieve any chosen level of ϵ-privacy and does not depend on the domain of the input data.

Local differential privacy refers to applying a privacy mechanism locally before sending data to a centralized server. This is relevant for collecting telemetry or data from client-side devices or browsers. Both Apple [2] and Google [15] developed locally private systems for

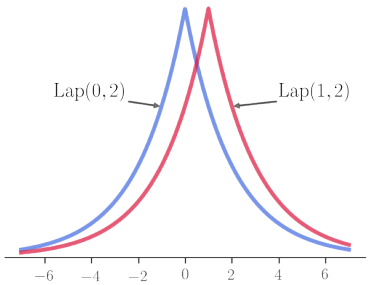

Figure 3: Example of a Laplace distributions offering .5-differential privacy for a function with sensitivity 1.

data collection.

Differential privacy academic literature tends to be oriented to one-shot releases of data, like the US Census, or a fixed set of database queries. In practice, these mechanisms may be difficult to apply to fast-changing databases, data with adversarial inputs, or unbounded numbers of queries.

Synthetic Data

Synthetic data involves taking real data and generating fake data sets that preserve statistical properties. One example generated synthetic commuter patterns from private US Census data [27].

Machine learning models trained on real data points can often be used to also generate sets of realistic-looking synthetic data. There are risks that unintentionally memorized data [8] embedded in machine learning models could be output in synthetic data sets. Because of this risk, differential privacy is often coupled with synthetic data generation to ensure that output is private. NIST recently ran a contest for generating such differentially private synthetic data sets [32].

Secure Enclaves

Secure enclaves are a CPU technology that provide a safe place to run code and perform computations on an otherwise unsafe platform. Intel's Software Guard Extension (SGX) [24] is one example of a more widely available enclave technology. In the context of privacy, enclaves allow one party (e.g. a regulator) to compute over another party's private data (e.g. a service provider) without learning any of the service provider's data. Furthermore, the service provider would not be able to know what a regulator is even searching for.

SGX is the basis for multiple privacy-preserving machine learning schemes [10, 36, 23]. The biggest barriers to adoption at this time is the availability of SGX in deployed servers and the lack of experience in enclave development by most parties. SGX is available on Microsoft Azure under the name Confidential Computing [30]. Microsoft is also making software tools available a part of the new Confidential Computing Consortium. Google also developed a privacy analytics tool, Prochlo [5], based on SGX.

Mix Networks

Mix networks [9] are protocols between mutually distrustful parties to shuffle values and unlink values from inputer's identities. Tor is the most well known mix net used practice and is intended to unlink

someone's source IP address from destination IP addresses [12]. Mix nets also are often proposed in voting schemes to anonymously shuffle votes. "Tumbler" services that mix cryptocurrency transactions to obscure the origins of funds are another form of mixnet.

In the scope of data sharing, mix networks can be used as a tool to allow third parties to unlink individual identities from data sets. For example, a concept of a **mix zone** was proposed as a way of achieving *k*-anonymity of transit trajectories [16]. A mix zone would be a geographic, high-traffic area where vehicle trajectories would not be logged. Upon leaving a mix zone, vehicles would be assigned new identities. If at least *k* vehicles are in the mix zone at all times, each vehicle crossing through would be *k*-anonymous. An example mix zone is illustrated in Figure 4.

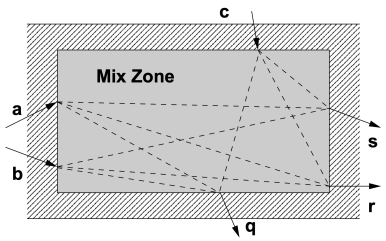

Figure 4: An illustration of a mix zone. Three trajectories, **a**, **b**, and **c**, enter the zone and three trajectories **q**, **r**, and **s** exit it. The paths from ingress trajectories to egress trajectories is hidden and in this example gives 3-anonymity. Source: [16]

Advanced Cryptography

Cryptography offers multiple technologies for privately computing over data, including **zero-knowledge proofs**, secure **multiparty computation**, and **homomorphic encryption**. There is a large body of work on all of these technologies going back to at least 1982 with Yao's notion of Garbled Circuits [37].

Zero-knowledge proofs have had the most recent adoption and development by digital currencies and will be discussed in the following section.

Until recently, real world applications of multiparty computation (MPC) were limited to niche use cases like beet auctions [6]. However, the combination of protocol performance improvements, instruction-level cryptographic primitives, and wider use of machine learning have led to more MPC proposals for privacy-preserving machine learning, e.g. [7].

Homomorphic encryption comes in two flavors: partial and fully. **Partially homomorphic encryption** has been supported for decades by many cryptosystems including RSA, ElGamal, and, most commonly, Paillier's [33]. Recently, Google applied partially homomorphic encryption for **Private Set Intersection** (PSI) [21].

Google is likely using PSI to privately join advertising data with external data from credit card companies [11]. The application is to attribute which ad impressions ultimately result in purchases, without either party sharing its respective ad impression data or purchase data. Private set intersection has a narrow scope, but is useful for summing common values between two parties' data sets without revealing their respective private data.

Fully homomorphic encryption (FHE) is a more powerful construction with the promise of being able to compute arbitrary cir-

cuits over encrypted data. It wasn't until 2009 [18] that the first FHE scheme was implemented; albeit one trillion times slower than a native computation. After a decade of development, FHE performance has improved by orders of magnitude, but is still too slow for general use. FHE is not commonly used in practice, so standards for tools or protocols haven't emerged yet. However, companies are working on building libraries, for example Microsoft Research released a FHE library called SEAL [31].

Verifiable Data & Verifiable Computation

An underlying motivation for data sharing is that one party may not trust another to honestly report aggregate data. For example, regulatory agencies conducting oversight may not trust aggregate data from service providers. **Verifiable data structures** allow parties to verify whether an element is a member of a set and, in some cases, whether it is a non-member.

A Merkle Tree [28] is a classic verifiable data structure based on hash functions and the underlying data structure of most blockchains.

Google's Trillian [19] verifiable data structures are used by Google's Certificate Transparency (CT) project allow browsers to verify whether a TLS certificate is a member of a known valid set. CT functions as a publicly verifiable, immutable log similar to a blockchain, except for being permissioned.

Verifiable computation coupled with a commitment log would allow a private data owner to prove to another party that aggregate values were correctly computed over private data. **Zero knowledge proofs** are a building block to prove knowledge of some value without revealing what it is.

One zero knowledge technology with practical adoption are zk-SNARKs, which stands for *Zero Knowledge, Non-interactive Succinct ARguments of Knowledge* [4]. SNARKs are used by anonymous digital currencies to prove that transactions reconcile without revealing the participants or transaction amount. That is, SNARKs can prove a transaction isn't creating or destroying money without revealing who is getting paid.

The combination of a verifiable data structure and zero knowledge proof system could make it unnecessary to share raw data in the first place. A publishing party would commit to a log of ciphertexts or oblivious commitments before sharing aggregated data. Then the publisher could prove that the inputs to the data aggregation correspond to the committed values. An analog of this are verifiable voting schemes where secret commitments to votes are published, privately tallied, then available for audit to third parties.

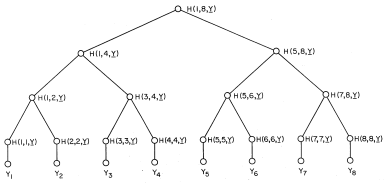

Figure 5: The original Merkle tree from his 1979 patent application.

Summary Table

This table summarizes the approaches discussed in this overview. *Lossy* means that shared data does not contain the entire original data set or does not contain individual records. *Complexity* is generally how complex a technology is and *maturity* is a general sense of how much the approach has been adopted. **Bold** values are desirable.

Approach	Lossy?	Complexity	Maturity
Minimization	N/A	**Low**	**High**
Redaction	Yes	**Low**	**High**
Aggregation	Yes	**Low**	**High**
Binning	Yes	**Low**	**High**
k-anonymity	Yes	Medium	Medium
Tokenization	**No**	**Low**	High
Format-Preserving Encryption	**No**	Medium	Medium
Differential Privacy	Yes	Medium	Medium
Synthetic Data	Yes	Medium	**High**
Secure Enclaves	**No**	High	Medium
Mix Networks	**No**	High	Low
Multiparty Computation	**No**	High	Low
Partially Homomorphic Encryption	**No**	Medium	Medium
Fully Homomorphic Encryption	**No**	High	Low
Verifiable Computation	**No**	High	Low

References

[1] Apple. Differential privacy. `https://www.apple.com/privacy/docs/Differential_Privacy_Overview.pdf`, 2017.

[2] Apple Differential Privacy Team. Learning with privacy at scale. `https://machinelearning.apple.com/docs/learning-with-privacy-at-scale/appledifferentialprivacysystem.pdf`, 2017.

[3] Bellare, M., Ristenpart, T., Rogaway, P., and Stegers, T. Format-preserving encryption. In *Selected Areas in Cryptography (SAC)* (2009), pp. 295–312.

[4] Ben-Sasson, E., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., and Virza, M. Zerocash: Decentralized anonymous payments from bitcoin. In *IEEE Symposium on Security and Privacy* (2014), pp. 459–474.

[5] BITTAU, A., ERLINGSSON, Ú., MANIATIS, P., MIRONOV, I., RAGHUNATHAN, A., LIE, D., RUDOMINER, M., KODE, U., TINNES, J., AND SEEFELD, B. Prochlo: Strong privacy for analytics in the crowd. In *Symposium on Operating Systems Principles (SOSP)* (2017), pp. 441–459.

[6] BOGETOFT, P., CHRISTENSEN, D. L., DAMGÅRD, I., GEISLER, M., JAKOBSEN, T. P., KRØIGAARD, M., NIELSEN, J. D., NIELSEN, J. B., NIELSEN, K., PAGTER, J., SCHWARTZBACH, M. I., AND TOFT, T. Secure multiparty computation goes live. In *Financial Cryptography and Data Security FC* (2009), pp. 325–343.

[7] BONAWITZ, K., IVANOV, V., KREUTER, B., MARCEDONE, A., MCMAHAN, H. B., PATEL, S., RAMAGE, D., SEGAL, A., AND SETH, K. Practical secure aggregation for privacy-preserving machine learning. In *ACM Conference on Computer and Communications Security CCS* (2017), pp. 1175–1191.

[8] CARLINI, N., LIU, C., KOS, J., ERLINGSSON, Ú., AND SONG, D. The secret sharer: Measuring unintended neural network memorization & extracting secrets. *CoRR abs/1802.08232* (2018).

[9] CHAUM, D. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM 24*, 2 (1981), 84–88.

[10] CHENG, R., ZHANG, F., KOS, J., HE, W., HYNES, N., JOHNSON, N. M., JUELS, A., MILLER, A., AND SONG, D. Ekiden: A platform for confidentiality-preserving, trustworthy, and performant smart contracts. In *IEEE European Symposium on Security and Privacy, EuroS&P 2019, Stockholm, Sweden, June 17-19, 2019* (2019), pp. 185–200.

[11] DARROCH, G. Google and mastercard cut a secret ad deal to track retail sales. *Bloomberg Technology* (August 2018).

[12] DINGLEDINE, R., MATHEWSON, N., AND SYVERSON, P. F. Tor: The second-generation onion router. In *USENIX Security Symposium* (2004), pp. 303–320.

[13] DWORK, C., AND ROTH, A. The algorithmic foundations of differential privacy. *Theoretical Computer Science 9*, 3-4 (2014), 211–407.

[14] DWORKIN, M. Recommendation for block cipher modes of operation: methods for format-preserving encryption. *NIST Special Publication 800* (2016), 38G.

[15] ERLINGSSON, Ú., PIHUR, V., AND KOROLOVA, A. Rappor: Randomized aggregatable privacy-preserving ordinal response. In

ACM Conference on Computer and Communications Security (CCS) (2014), pp. 1054–1067.

[16] FREUDIGER, J., RAYA, M., FÉLEGYHÁZI, M., PAPADIMITRATOS, P., AND HUBAUX, J.-P. Mix-zones for location privacy in vehicular networks. In *ACM Workshop on Wireless Networking for Intelligent Transportation Systems (WiN-ITS)* (2007).

[17] GARFINKEL, S. L. Deploying differential privacy for the 2020 census of population and housing. https://www.census.gov/content/dam/Census/newsroom/press-kits/2019/jsm/presentation-deploying-differential-privacy-for-the-2020-census-of-pop-and-housing.pdf, July 2019.

[18] GENTRY, C. Fully homomorphic encryption using ideal lattices. In *ACM Symposium on Theory of Computing,STOC* (2009), pp. 169–178.

[19] GOOGLE. Trillian: A transparent, highly scalable and cryptographically verifiable data store. https://opensource.google.com/projects/trillian.

[20] GOOGLE. Computing l-diversity with cloud DLP. https://cloud.google.com/dlp/docs/compute-risk-analysis#compute-l-diversity, 2019.

[21] GOOGLE. Helping organizations do more without collecting more data. https://security.googleblog.com/2019/06/helping-organizations-do-more-without-collecting-more-data.html, 2019.

[22] GUEVARA, M. Enabling developers and organizations to use differential privacy. https://developers.googleblog.com/2019/09/enabling-developers-and-organizations.html, September 2019.

[23] HUNT, T., SONG, C., SHOKRI, R., SHMATIKOV, V., AND WITCHEL, E. Chiron: Privacy-preserving machine learning as a service. *CoRR abs/1803.05961* (2018).

[24] INTEL. Software guard extensions. https://software.intel.com/en-us/sgx, 2019.

[25] LI, N., LI, T., AND VENKATASUBRAMANIAN, S. t-closeness: Privacy beyond k-anonymity and l-diversity. In *International Conference on Data Engineering (ICDE)* (2007), pp. 106–115.

[26] MACHANAVAJJHALA, A., GEHRKE, J., KIFER, D., AND VENKITASUBRAMANIAM, M. l-diversity: Privacy beyond k-anonymity. In *International Conference on Data Engineering (ICDE)* (2006), p. 24.

[27] MACHANAVAJJHALA, A., KIFER, D., ABOWD, J. M., GEHRKE, J., AND VILHUBER, L. Privacy: Theory meets practice on the map. In *International Conference on Data Engineering ICDE* (2008), pp. 277–286.

[28] MERKLE, R. C. Method of providing digital signatures, September 1979. US Patent 4,309,569.

[29] MESSING, S., DEGREGORIO, C., HILLENBRAND, B., KING, G., MAHANTI, S., NAYAK, C., PERSILY, N., STATE, B., AND WILKINS, A. Facebook privacy-protected URLs light table release. `https://socialscience.one/files/partnershipone/files/facebook_urls-light_codebook_v2.0.pdf`, September 2009.

[30] MICROSOFT. Azure confidential computing. `https://azure.microsoft.com/en-us/solutions/confidential-compute/`, 2019.

[31] MICROSOFT RESEARCH. SEAL. `https://github.com/Microsoft/SEAL`, 2019.

[32] NIST. Differential privacy synthetic data challenge. `https://www.nist.gov/communications-technology-laboratory/pscr/funding-opportunities/open-innovation-prize-challenges-1`, 2018.

[33] PAILLIER, P. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology - EUROCRYPT* (1999), pp. 223–238.

[34] SWEENEY, L. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10*, 5 (2002), 557–570.

[35] TEZASIDIS, K. Uber releases open source project for differential privacy. `https://medium.com/uber-security-privacy/differential-privacy-open-source-7892c82c42b6`, July 2017.

[36] TRAMER, F., AND BONEH, D. Slalom: Fast, verifiable and private execution of neural networks in trusted hardware. *arXiv preprint arXiv:1806.03287* (2018).

[37] YAO, A. C. Protocols for secure computations. In *Foundations of Computer Science (FOCS)* (1982), pp. 160–164.