Beyond Cryptography: Deniable Privacy for Secure Data Aggregation

by

Eric J. Pence

S.B., Electrical Engineering and Computer Science,
Massachusetts Institute of Technology, 2021

Submitted to the Department of Electrical Engineering and Computer Science

in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2022

Author .....
Department of Electrical Engineering and Computer Science
May 20, 2022

Certified by .....
Daniel Weitzner
3Com Founders Principal Research Scientist, MIT Computer Science and Artificial Intelligence Research Lab (CSAIL)
Thesis Supervisor

Certified by .....
Taylor Reynolds
Technology Policy Director, MIT Internet Policy Research Initiative (IPRI)
Thesis Supervisor

Accepted by .....
Katrina LaCurts
Chair, Master of Engineering Thesis Committee

Beyond Cryptography: Deniable Privacy for Secure Data Aggregation

by

Eric J. Pence

Submitted to the Department of Electrical Engineering and Computer Science
on May 20, 2022, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

We assess the privacy properties of the count function, an essential data aggregation primitive, in the context of a real-world secure data aggregation platform called SCRAM (Secure Cyber Risk Aggregation and Measurement). Subject to the constraints of few data contributors and a limited tolerance for noise in the output of the count function, we seek an alternative to differential privacy, and we develop a new privacy-preserving mechanism called deniable privacy. We show that deniable privacy provides the proper balance between accuracy and privacy in the case of SCRAM, and we demonstrate that the utility of deniable privacy extends broadly to other data aggregation applications.

Thesis Supervisor: Daniel Weitzner
Title: 3Com Founders Principal Research Scientist, MIT Computer Science and Artificial Intelligence Research Lab (CSAIL)

Thesis Supervisor: Taylor Reynolds
Title: Technology Policy Director, MIT Internet Policy Research Initiative (IPRI)

Acknowledgments

I would like to thank Daniel Weitzner and Taylor Reynolds for their constant support, guidance, and feedback throughout my research. It was a joy to work with both Daniel and Taylor, and I am so lucky to have had the opportunity to conduct research within MIT's Internet Policy Research Initiative (IPRI). In fact, the whole leadership team of IPRI deserves recognition, as every faculty member I encountered was tremendously supportive. I would also like to express my deepest gratitude to Leo de Castro, who inspired many of the contributions offered by this paper. Leo always knew the next step to take when our research hit a dead end. He is both a perceptive mentor and skilled academic.

Contents

List of Figures

Chapter 1

Introduction

Aggregation and sharing of data have been key drivers behind societal advancement, technological and otherwise. Finance, agriculture, education [6], and pandemic response efforts [4, 18] have all benefited from wide-scale data aggregation. Given the proliferation of such tools in recent years, it is safe to assume that they are here to stay. Unfortunately, the privacy properties of secure data aggregation functions are not well understood, especially when the number of data points and the number of data contributors is small (i.e. less than 100).

Assessing the landscape of tools for privately releasing aggregate statistics, we believe this field has tended toward two extremes. On one end, there is Multiparty Computation-style privacy, which is essentially no output privacy at all. The Multiparty Computation (MPC) definition makes no guarantees on what an output itself may reveal about the inputs that produced it [12]. The advantage of the MPC-style privacy definition is that it offers maximal utility, since there is no distortion of the data. The disadvantage, of course, is the lack of privacy guarantees. On the other end, there is the differential privacy [15], which by definition limits how much information is revealed by a function's output and holds true even in the face of worst-case adversarial knowledge [15]. The advantage of this approach is the strong and widely accepted privacy guarantees it offers [1, 4]. The disadvantage of differential privacy is two-fold. First, it requires the data to be distorted by the addition of noise, and this noise diminishes the utility of the data. Second, the parameters of ϵ and δ in its def-

inition are not very intuitive. It is not clear how much privacy is afforded by specific values of ϵ and δ, which is why the National Institute of Standards and Technology has issued broad guidance on setting the parameters for differential privacy [7]. At both of these extremes, insufficient attention has been paid to offering privacy when the number of data contributors is small, and not enough effort has been placed on reducing the amount of noise used to provide privacy. Our work attempts to highlight this gap and address it for a specific data aggregation function, the count function.

1.1 Major Contributions

In this work, we provide an empirical assessment of the privacy guarantees inherent to one specific data aggregation function: the count function, as defined in Section 3.3.3. This empirical assessment first explains the risks and protections afforded by the count function when no noise is added to its output. This is what we refer to as the noiseless approach to privacy, which is based on the work of Bhaskar et al. [10]. The noiseless approach to privacy relies on randomness embedded in the data itself to provide privacy. Our analysis reveals the fact that, for the count function, this noiseless approach to privacy does not offer protections when the number of data providers to the count function is less than 31, assuming a binomial distribution over the output data and privacy parameter $\delta = 10^{-9}$. This leads us to design a new privacy mechanism: deniable privacy. As shown in Chapter 3, deniable privacy allows us to execute the count function privately, even when the number of data providers, or inputs, to the count function is less than 31. Further, our analysis of noiseless privacy and deniable privacy provides an intuitive explanation for the relationships between the privacy parameters ϵ and δ and the number of data contributors n, as well as the distribution $\mathcal{D}$ of the input data.

In sum, our major contributions are three-fold. First, we conduct a baseline privacy assessment of the count function, as applied to a specific secure data aggregation platform. We show the concrete values for a variety of parameters, including ϵ and δ, that are required to achieve noiseless privacy. Second, we offer a new notion of pri-

vacy, specific to the count function, called deniable privacy. Deniable privacy adds the minimum amount of noise necessary to hide outputs that are inherently non-private, and in doing so it offers stronger privacy guarantees for smaller n, as compared with the noiseless approach to privacy. Finally, we generalize the notion of deniable privacy to produce k-deniable privacy, which further improves the privacy guarantees afforded by the count function at the expense of additional noise.

Chapter 2

Background

2.1 Secure Data Aggregation

Before diving into the details of Multiparty Computation (MPC) and the platform called Secure Cyber Risk Aggregation and Measurement (SCRAM) that motivated this work, it is critical to note that our analysis and findings pertaining to the count function are applicable to secure data aggregation platforms generally. Though we discuss MPC and SCRAM in detail, we do so only in an effort to demonstrate the utility of our findings in a real-world application. Throughout this work, keep in mind that the privacy insights we derive from assessing the secure data aggregation functions implemented by SCRAM apply broadly to most, if not all, secure data aggregation protocols.

2.1.1 Multiparty Computation (MPC)

One tool that has aided in the success of data aggregation in recent year is Multiparty Computation (MPC) [19]. Though MPC has existed for decades, it is now becoming a popular and reliable replacement for trusted third party data aggregators [13]. Even so, for the privacy-conscious data owner, MPC alone is not always a satisfying solution to data aggregation needs.

To understand why this is the case, it is essential to start by understanding what

MPC protects against, and what it does not. MPC, as first defined in the two-party setting by Yao [23], is a tool that facilitates anonymous and secure data aggregation. Its increasingly popularity is shown by the work of Boston University's Accessible and Scalable Secure Multi-Party Computation group [19]. While MPC is provably secure [17], it fails to address a critical dilemma in data aggregation: the output of its aggregation function may leak information that the parties intended to keep private, especially if this output can be can combined with information external to the computation. We often refer to this type of extra information as auxiliary information.

Crucially, this is not a failure of MPC. The definition of MPC guarantees that the only new information revealed by the MPC protocol itself is the output of the protocol [12]. If the output of a secure multiparty aggregation function f is some dataset Y, the definition claims that Y is the only new information revealed, which is a very powerful property provided the output Y of aggregation function f does not leak much, if any, additional information. However, the MPC definition makes no claim about the privacy properties of output Y, including what Y implies about the inputs to f. So, even though MPC is often used as a replacement for a trusted third-party data aggregator [13], data owners cannot be sure the functions being computed will produce privacy-preserving outputs. This results in data owners who are hesitant or outright unwilling to share their data despite the potential MPC has to unlock collectively beneficial insights, such as industry-wide trends in cybersecurity [13], that would otherwise be unknown and untapped.

Popular literature might suggest that differential privacy is the best solution to ensuring a securely aggregated dataset preserves privacy. In fact, differential privacy is widely deployed because it is both powerful and versatile; it makes no assumptions about the distribution over the input data and it holds even in the face of worst-case adversarial knowledge [15]. However, differential privacy relies on the addition of noise to the output of the aggregation function [15], and there are situations in which noisy data is unacceptable. Dwork [14] points out one instance where a noisy output is unacceptable: price setting in auctions. While each bidder may have an interest in keeping their exact bid private, noise can unnaturally inflate or deflate the true

selling price [14]. Another situation where a noisy output is unacceptable is financial auditing. Take, for example, a financial auditor who needs to know the exact revenues and expenditures, including salaries, of a company [10]. Such an individual will not tolerate a noisy dataset, though the company may have in interest in keeping certain details about their revenues and expenditures private.

Clearly, there is a need for secure data aggregation functions that offer privacy guarantees while faithfully representing the data over which they compute. Ideally, this faithful representation would involve no noise at all. This is the focus of our work, as existing literature on secure data aggregation does not sufficiently address privacy, or its relationship with utility, especially for small datasets.

2.1.2 Secure Cyber Risk Aggregation and Measurement

To ground our discussion of MPC and the broader family of secure data aggregation protocols, we rely on the platform named Secure Cyber Risk Aggregation and Measurement (SCRAM) [13] as a case study.

SCRAM is an MPC platform that computes aggregate statistics over encrypted data. Specifically, SCRAM handles cybersecurity-related data. Despite this specific use case, SCRAM's foundational MPC infrastructure can handle data of any kind. This further ensures the lessons we derive from SCRAM apply more broadly to secure data aggregation. As with any MPC application, the SCRAM server never sees the plain-text inputs of the participants. Instead, it executes an aggregation function, agreed upon by the participants ahead of the computation, over the encrypted data and returns an encrypted result. It is then up to the participants to jointly decrypt this result into plain-text.

SCRAM's computation process is secure in accordance with the definition of MPC [13]. But, as we pointed out in Chapter 1, this does not mean the outputs released by SCRAM are private. For this reason, we must zoom in on the particular functions deployed by SCRAM before we begin the privacy analysis of Chapter 3.

SCRAM Functions

To begin any assessment of the privacy afforded by secure data aggregation, we find that it is essential to define and analyze the intended aggregate output and the particular functions used to compute this output. In the case of SCRAM, two key metrics, both intended to inform future cybersecurity investment, are computed:

1. Implementation level of cybersecurity controls

2. Dollar losses per cybersecurity control

A cybersecurity control, as defined in [13], is one of 171 defensive measures specified by the Center for Internet Security (CIS). So, the implementation level metric gives an individual firm perspective on how their defensive posture compares to other firms. The dollar losses metric gives perspective on which CIS controls are most important to implement from an economic value perspective, as a firm is inclined to invest more in defenses that protect against larger losses.

The implementation level metric is computed via a count function that follows Definition 5 in Section 3.3.3. The desire among existing SCRAM computation participants to compute this metric privately motivates our analysis in Chapter 3. In fact, it is the focus of our work. We save analysis of the second metric, computed via a sum function, for future work, as discussed in Chapter 4. Individually, these two functions are key building blocks of many secure data aggregation computations, which is what makes our analysis and conclusions broadly applicable.

SCRAM Privacy Constraints

Notably, we have executed many aggregation operations in the real-world with SCRAM [13]. Throughout all of these operations, we have encountered two key privacy constraints on SCRAM:

1. Each computation consists of very few data contributors. In practice, the number of participants n in any given SCRAM computation is often less than 20 ($n < 20$).

2. Noise greatly diminishes the utility of a SCRAM computation because our real-world participants do not want to add noise to their data directly, or to the output of the aggregation function.

Taken together, these constraints pose a great challenge to the privacy guarantees sought by SCRAM. Producing a very large dataset and adding noise to a dataset are generally thought of as reliable techniques to improving privacy, but the use case of SCRAM limits our ability to use on them.

SCRAM Privacy Goal

The last step, before beginning our privacy analysis of the count function, is to define what privacy guarantees a secure data aggregation platform intends to offer. This is necessary to provide us with a clear success metric to measure our privacy guarantees against. Crucially, different platforms will seek different privacy standards, so we focus on the privacy standard sought by our use case of SCRAM.

We define SCRAM privacy as the disassociation of participant identities from their inputs. We do not want an adversary to be able to prove with certainty that a particular participant in the computation input a particular response. Making this explicit, we define $Y = \mathbf{SCRAM}_P(X)$ as a SCRAM computation with a group of n individual data contributors $P = \{p_1, \ldots, p_n\}$, private input $X = \{x_1, \ldots, x_n\}$, and aggregate output Y. We assume each data contributor p_i is responsible for input x_i for $i \in [n]$. We define the disassociation of participant identities from their inputs as the property that the aggregate output Y should not enable a participant p_i for $i \in [n]$ to determine with certainty the input x_j for $j \neq i \in [n]$ of participant x_j.

This privacy definition for SCRAM embraces the fact that an adversary or curious observer, especially one possessing substantial auxiliary information as described in Chapter 1, will be able to make inferences about the input provided by each data contributor provided. Based on the Reiter and Rubin notion of possible innocence [21], the SCRAM privacy definition is not concerned with preventing inference. Instead, it seeks to offer participants deniability. We wish to provide participants of the SCRAM computation the ability to deny any inference of their input by an adversary.

To make the idea of deniability more clear, we briefly describe the concept of possible innocence [21].

Possible Innocence

Reiter and Rubin [21] define possible innocence in terms of web transactions, but the notion extends to privacy in data aggregation. They specify possible innocence as one level of privacy on a spectrum from absolute privacy to provably exposed [21]. The exact definition of possible innocence is as follows:

Definition 1 (Possible Innocence ([21], figure 1))**.** *A sender is possibly innocent if, from the attacker's point of view, there is a nontrivial probability that the real sender is someone else.*

In the case of SCRAM, we replace the sender with the data contributor, and then we set this definition as our standard.

Justifying Possible Innocence for SCRAM

We explain why possible innocence is the right privacy definition for SCRAM by describing the three most potent forms of risk posed by the SCRAM computation: regulatory, reputational and adversarial. We define regulatory risk as the likelihood a participant faces adverse regulatory action [3, 9], possibly in the form of fines, as a result of the output leaking unintended information. We define reputational risk as the likelihood a participant suffers reputational harm as a result of the output leaking unintended information [9]. We define adversarial risk as the likelihood a participant becomes a target of a cyberattack as a result of the output leaking unintended information, such as cyber-vulnerabilities [3].

Considering these three forms of risk, we adopt the notion of possible innocence as SCRAM's privacy standard because we believe a data contributor can avoid the bulk of regulatory, reputational, and adversarial consequences if they are simply able to deny wrong-doing or vulnerability. Of course, possible innocence does not protect

against the suspicion of wrong-doing or vulnerability, but it does offer data contributors the ability to deny suspicion. As described in Section 3.5, this notion of deniability is achieved by our key contribution of deniable privacy.

2.2 Differential Privacy

Having defined our use case of SCRAM and its privacy definition, we now provide an overview of differential privacy. As explained in Chapter 1, differential privacy is often thought of as the best method for providing privacy to aggregate datasets. For this reason, we provide its definition and key characteristics so that we can benchmark the notions of noiseless privacy (Definition 4) and deniable privacy (described in Section 3.5) against differential privacy in Chapter 3.

Differential Privacy, as defined by Dwork and Roth [15], is widely considered the gold standard for privacy because it protects against arbitrary auxiliary information [15]. Its definition is as follows:

Definition 2 ((ϵ, δ)-Differential Privacy ([15], definition 2.4))**.** *A randomized algorithm M with domain $N^{|X|}$ is (ϵ, δ)-Differentially Private if for all $S \subseteq \mathsf{Range}(M)$ and for all $x, y \in N^{|X|}$ such that $||x - y||_1 \leq 1$:*

$$\Pr[M(x) \in S] \leq e^{\epsilon} \cdot \Pr[M(y) \in S] + \delta$$

where the probability space is over the coin flips of the mechanism M.

Importantly, differential privacy achieves the guarantees in Definition 2 by adding noise to the output of mechanism M. This noise can be drawn from a variety of distributions, including the Laplace and Gaussian distributions. While work has been done to improve utility when adding Gaussian noise [8], Dwork and Roth [15] argue it is often best to draw noise from the Laplace distribution as follows:

Definition 3 (Laplace Distribution with Scale $\frac{1}{\epsilon}$ and Mean 0 ([15], definition 3.2))**.**

$$\mathsf{Lap}\left(x \middle| \frac{1}{\epsilon}\right) = \frac{\epsilon}{2} \cdot \exp(-\epsilon \cdot |x|)$$

While differential privacy has been widely adopted, including by Apple for purposes such as improving Emoji suggestions and gather Safari crash data [1] and Google for COVID-19 disease tracking [4], it has drawbacks. One such drawback has been highlight by the U.S. Census Bureau, which claims difficulty in setting the differential privacy parameters [16]. Another drawback, as mentioned in Chapter 1 and highlighted by Dwork [14], is that the noise it adds is unacceptable for various data aggregation tasks. Our work in Chapter 3 calls attention to both of these drawbacks by analyzing the privacy properties of the count function.

The remaining Chapters of this work serve to illustrate how the privacy vulnerabilities of a specific data aggregation function, the count function, can be understood and mitigated. Throughout this work, we call attention to the two major drawbacks of differential privacy, as stated in Section 2.2.

Chapter 3

Privacy Analysis of the Count Function

3.1 Introduction

The purpose of this chapter is two-fold. First, we empirically define the baseline privacy offered by a secure data aggregation function, namely the count function, when no noise or other privacy-preserving mechanisms are added to the function. We find that this approach fails when the number of data contributors n is less than 31, given our application-specific $\delta = 10^{-9}$ (explained in Section 3.4.2). This approach is what we refer to as the noiseless approach to privacy [10]. Then, in order to provide privacy when the number of data contributors to the count function is smaller than 31 ($n < 31$), we define the deniable approach to privacy. This approach is grounded in a new mechanism that both ensures privacy for all values of n and does not suffer from the same utility loss as differential privacy.

Both the noiseless approach to privacy and the deniable approach to privacy are motivated by the two major drawbacks of differential privacy, as explained in Section 2.2. Specifically, the utility loss from differential privacy motivates the noiseless approach [10] to privacy, as described in Section 3.3.2. Further, the lack of intuition behind how to set the parameters of differential privacy motivates the way we describe deniable privacy in Section 3.5. Finally, the lack of clarity in differential

privacy literature on the trade-off between privacy and utility when the number of data points is small, combined with the small n constraint on SCRAM (described in Section 2.1.2), motivates our desire to achieve deniable privacy for small numbers of data contributors.

3.2 Road Map

Recognizing the count function to be a critical data aggregation primitive, as well as a key component of SCRAM, we assess its privacy properties in depth. The structure of this chapter is as follows:

In Section 3.3, we define our notation, provide the definition of noiseless privacy (Definition 4, [10]), and detail how this definition can be applied to the count function. In Section 3.4, we show how to model the count function as a binomial random variable, as done by Bhaskar et al. [10]. We then define the parameters of ϵ and δ in terms of the count function, and show how to set these parameters to achieve noiseless privacy. In Section 3.5, motivated by the failures of noiseless privacy, we define a modified count function, denoted as $\mathbf{Count}_{deniable}()$. We show that this function improves the privacy afforded by the noiseless approach, while adding a very limited amount of noise to the aggregate output. We call this deniable privacy. In Section 3.6, we generalize deniable privacy and introduce a noisier version of $\mathbf{Count}_{deniable}()$ that further improves privacy guarantees.

3.3 Background on Noiseless Privacy

3.3.1 Notation

Binomial Distribution Notation

The binomial distribution is a discrete probability distribution of the sum of n independent and repeated Bernoulli trials. A Bernoulli trial is a random variable that takes the value of 1 with probability p and takes the value of 0 with probability $1-p$.

We denote the Bernoulli distribution as $\textbf{Bernoulli}(p)$, where p is the probability of 1 in any given sample. We denote the binomial distribution as $\mathcal{B}(n, p)$, where n is the number of Bernoulli trials and p is the probability of 1 in any given trial. A binomial random variable is a discrete random variable sampled from the binomial distribution. We denote the probability mass function (PMF) of a binomial random variable $\mathcal{B}(n, p) = k$ for $k \in \{0, \ldots, n\}$ as $\textbf{PMF}(k; \mathcal{B}(n, p))$. The definition of this quantity is as follows:

$$\textbf{PMF}(k; \mathcal{B}(n, p)) = \binom{n}{k} \cdot p^k \cdot (1 - p)^{n-k}$$

Negligible Function Notation

The negligible function is defined as a function $negl : \mathbb{N} \to \mathbb{R}$ such that for every positive polynomial $poly(\cdot)$ there exists and integer $N_{poly} > 0$ such that for all $x > N_{poly}$:

$$|negl(x)| < \frac{1}{poly(x)}$$

We denote a function that is negligible in its input x as $negl(x)$.

3.3.2 Defining Noiseless Privacy

As a response to the diminished utility afforded by the noisy outputs of differentially private mechanisms [22], Bhaskar et al. present noiseless privacy in their paper Noiseless Database Privacy [10]. Its definition is as follows:

Definition 4 ((ϵ, δ)-Noiseless Privacy ([10], definition 4))**.** *Let $f : D^n \to Y$ be a deterministic query function on a database of length n drawn from domain D. Let $\mathcal{D}$ be a distribution on D^n. Let $S_1 \subseteq Y$ and $S_2 \subseteq D$ be two sets such that for all $j \in [n]$, $\Pr_{T \sim \mathcal{D}}[f(T) \in S_1] + \Pr_{T \sim \mathcal{D}}[t_j \in S_2] \leq \delta$, where t_j is the j^{th} entry of T. The function f is said to be (ϵ, δ)-Noiseless Private under distribution $\mathcal{D}$ and some auxiliary information Aux, if there exists S_1, S_2 as defined above such that, for all measurable sets $\mathcal{O} \in Y - S_1$, for all $a, a' \subseteq D - S_2$ and for all $l \in [n]$ the following*

holds:

$$\Pr_{T\sim\mathcal{D}}[f(T) \in \mathcal{O}|t_l = a, Aux] \leq e^{\epsilon} \cdot \Pr_{T\sim\mathcal{D}}[f(T) \in \mathcal{O}|t_l = a', Aux] \tag{3.1}$$

Similar to the definition of (ϵ, δ)-differential privacy (Definition 2 in Section 2.2) (ϵ, δ)-noiseless privacy is also parameterized by ϵ, a quantity to bound the probability ratio of two outcomes, and δ, a failure rate of the privacy definition itself. However, (ϵ, δ)-noiseless privacy is distinctly different from (ϵ, δ)-differential privacy in four critical ways:

1. It can hold for a deterministic function f because it relies on the randomness within the input to f, whereas (ϵ, δ)-differential privacy relies on the randomness of the function itself.

2. It assumes a distribution $\mathcal{D}$ over the inputs, whereas (ϵ, δ)-differential privacy does not assume any distribution.

3. It protects against auxiliary information possessed by an adversary only if that information is specified in the parameter Aux, whereas (ϵ, δ)-differential privacy protects against arbitrary auxiliary information.

4. It explicitly does not add noise to the inputs or outputs of function f, whereas (ϵ, δ)-differential privacy adds noise.

The key idea behind (ϵ, δ)-noiseless privacy in Definition 4 is that the data itself adds entropy to the output of the deterministic function f. Further, Definition 4 does not assume worst-case adversarial knowledge, like differential privacy. Together, the entropy in the data and limit on adversarial knowledge enable Definition 4 to hold without the addition of noise to the output of mechanism f. Equipped with Definition 4, we assess what it reveals about the privacy properties of the count function, specifically.

3.3.3 Defining the Count Function

The count function is simple, yet powerful. The count function is often used to tally the number of individuals who express a property of interest [5, 13]. Notably, the count function is a symmetric function. Bhaskar et al. [10] define a symmetric function as follows:

Definition 5 (Symmetric Function ([10], definition 8))**.** *A function* $f : \{0,1\}^n \to \mathbb{R}$ *is said to be symmetric, if for any input string* $T = \langle t_1 t_2 \dots t_n \rangle \in \{0,1\}^n$ *and for all permutations* σ *from the set of permutations over the set* $\{1, \dots, n\}$ *the following holds* $f(t_1, \dots, t_n) = f(t_\sigma(1), \dots, t_\sigma(n))$

Throughout this analysis we denote the count function as $\mathbf{Count}(T)$, where T is the collection of n input bits t_i for $i \in [n]$. When we assume each input bit t_i to $\mathbf{Count}(T)$ to be a repeated and independent Bernoulli trial, as explained in Section 3.4.1, we denote the random variable that represents the output of $\mathbf{Count}(T)$ as $\mathcal{C}$.

The utility of such a function is best illustrated with an example. In the case of SCRAM, $\mathbf{Count}(T)$ can be used to answer questions like "How many firms participating in this computation deploy Multi-Factor Authentication (MFA) on company-owned devices?" To answer this question, each of the n firms participating in the computation would submit a single bit answer t_i that is 1 if they deploy MFA and 0 if they do not. Together these t_i for $i \in [n]$ would form the input string T, where each t_i comes from a unique data contributor. In accordance with Definition 5, regardless of who inputs which bit t_i, the count computed is simply the sum of all the bits.

Therefore, in addition to being an essential data aggregation primitive, $\mathbf{Count}(T)$ is also a crucial component of SCRAM. For these reasons, we assess its privacy properties in depth.

3.3.4 Privacy of the Count Function

Applying the definition of (ϵ, δ)-noiseless privacy (Definition 4) to $\mathbf{Count}(T)$, Bhaskar et al. [10] show that satisfying the following maximum-likelihood estimator (MLE) is

sufficient for satisfying (ϵ, δ)-noiseless privacy:

Lemma 1 (Count MLE Ratio ([10])). *Let $T \in \{0,1\}^n$ be sampled from the distribution $\mathcal{D}$ where each bit is 1 with probability p. Let c_T be the count of the number of bits in T which are 1 and let a be any element from the set $\{0, \ldots, n\}$. **Count**(T) satisfies (ϵ, δ)-noiseless privacy if for all $i \in [n]$ the following ratio holds with probability $1 - \delta$:*

$$\left| \ln(\Pr_{T \sim \mathcal{D}}[C_T = a | t_i = 0]) - \ln(\Pr_{T \sim \mathcal{D}}[C_T = a | t_i = 1]) \right| \leq \epsilon \tag{3.2}$$

We explain Lemma 1 by first conveying what is represented by Equation 3.2 itself. Intuitively, the difference in logs in Equation 3.2 represents an adversary's ability to predict the output of the count function a, knowing the input bit t_i of a particular data contributor $i \in [n]$. The minuend represents the likelihood an adversary would infer $\mathcal{C} = a$ given knowledge that data contributor $i \in [n]$ provided input bit $t_i = 0$. The subtrahend represents the likelihood an adversary would infer $\mathcal{C} = a$ given knowledge that $t_i = 1$. Simply put, by conditioning the value of the count on the value of t_i, the probabilities themselves answer the question, "What's the likelihood we see a count of a, given this particular input from data contributor i?"

Bounding this difference of logs below by $-\epsilon$ and above by ϵ ensures that, after seeing the true count, an adversary will never be more than e^ϵ times more confident that data contributor $i \in [n]$ gave a particular response, either 0 or 1, instead of the other. In sum, Lemma 1 bounds the performance of an adversary by the performance of the maximum-likelihood estimator (Equation 3.2).

Using the Equation 3.2 from Lemma 1, Bhaskar et al. [10] prove the following theorem:

Theorem 1 ($(\epsilon, negl(n))$-Noiseless Privacy ([10], theorem 5)). *Let T be from the distribution $\mathcal{D}$ where each bit is 1 with probability p (a constant). Let c_T be the count of number of bits in $T \in \{0,1\}^n$ which are 1 and let a be any element from the set $\{0, \ldots, n\}$. For any constant $\epsilon > 0$, c_T is $(\epsilon, negl(n))$-noiseless private.*

The key result of this theorem is the asymptotic guarantee that as $n \to \infty$ every $\epsilon > 0$ will satisfy $(\epsilon, negl(n))$-noiseless privacy, given some function $negl(n)$. However, Theorem 1 does not specify a particular function for $negl(n)$, and is consequently not precise enough to be actionable. By guaranteeing (ϵ, δ)-noiseless privacy relative to n and not in absolute terms, Theorem 1 only offers the promise that as n grows, (ϵ, δ)-noiseless privacy will eventually be achieved. Theorem 1 does not provide any insight into if (ϵ, δ)-noiseless privacy can be achieved when n is small, nor does it explain how a data aggregator should set δ for a given ϵ and a given n in order to satisfy (ϵ, δ)-noiseless privacy under a distribution $\mathcal{D}$. This is where the contributions of this work begin. We show precisely when (ϵ, δ)-noiseless privacy will hold for a specific δ, given exact parameters of ϵ from Definition 4 and n and p from the binomial distribution, $\mathcal{B}(n, p)$.

3.4 (ϵ, δ)-Noiseless Privacy Baseline

3.4.1 Modeling the Count with a Binomial

Following the analysis of Bhaskar et al. [10], we assume the output $\mathcal{C}$ of **Count**() follows a binomial distribution $\mathcal{B}(n, p)$. This assumption enables us to show privacy guarantees via Bayesian analysis, even when we do not add noise to output $\mathcal{C}$ or randomness to the function **Count**().

It is helpful to grasp both mathematically and graphically what it means for the count to be a binomial random variable $\mathcal{B}(n, p)$. Take, for example, a SCRAM computation with 20 participants ($n = 20$) where each participant has a 50% chance of implementing a given cybersecurity control ($p = 0.5$). This translates to $\mathcal{C} = \mathcal{B}(n = 20, p = 0.5)$, as plotted in Figure 3-1.

As can be seen in Figure 3-1, such a representation necessarily implies that most of the probability mass falls around the mean ($n \cdot p = 10$), while very little probability mass is attributed to extreme counts, like $\mathcal{C} = 0$ and $\mathcal{C} = 20$. Practically speaking, this means we expect to see $\mathcal{C} = 10$ and we will rarely see $\mathcal{C} = 0$ or $\mathcal{C} = 20$. Observing

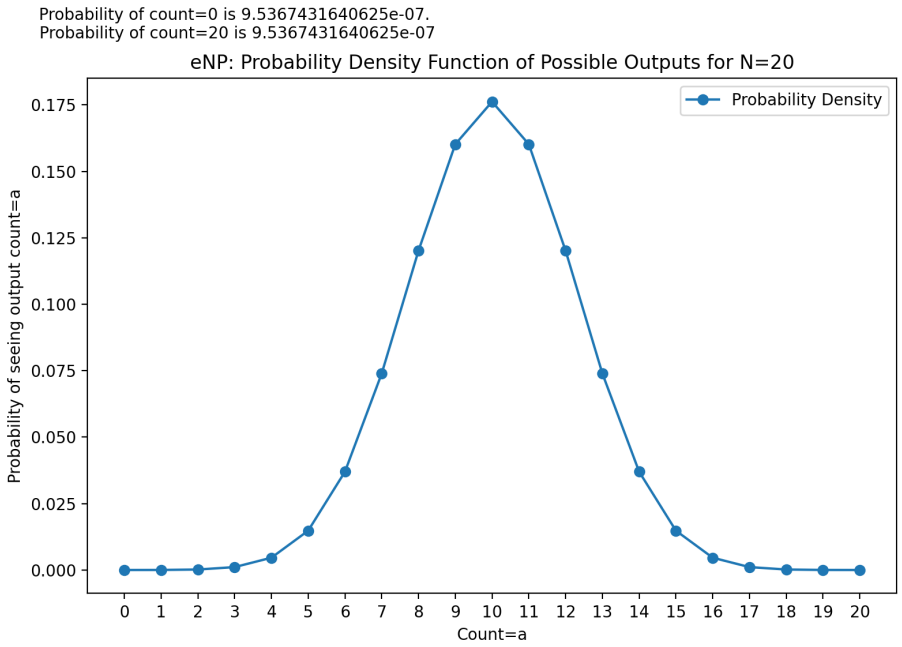

Figure 3-1: Count Represented as a Binomial Random Variable

how little probability mass falls on these extreme cases is essential to the privacy analysis ahead.

Critically, we refer to the two cases of $\mathcal{C} = 0$ and $\mathcal{C} = n$ as failure cases throughout our analysis. We describe them as failure cases because they are the two outputs of the count function for which privacy is necessarily impossible. Upon seeing $\mathcal{C} = 0$ or $\mathcal{C} = n$, the exact input of every data contributor is learned.

Defining (ϵ, δ)-Noiseless Privacy with Binomials

Adopting the binomial random variable representation $\mathcal{C} = \mathcal{B}(n, p)$, we show the following lemma:

Lemma 2 (Count-Binomial Equivalence ([10], proof of Theorem 5))**.**

$$\Pr[\mathcal{C} = a | t_i = 0] = \Pr[\mathcal{B}(n-1, p) = a]$$
$$\Pr[\mathcal{C} = a | t_i = 1] = \Pr[\mathcal{B}(n-1, p) = a-1]$$

where T is from the distribution $\mathcal{D}$ and each bit t_i is 1 with probability p. $\mathcal{C}$ is the count of number of bits in $T \in \{0,1\}^n$ which are 1 and a is any element from the set $\{0,\ldots,n\}$.

As shown in the first equality, when $t_i = 0$, the remaining $n-1$ inputs must account for the entire $\mathcal{C} = a$. However, as shown in the second equality, when $t_i = 1$ the remaining $n-1$ inputs must account for precisely $(\mathcal{C}-1) = (a-1)$. Putting together Lemma 1 and Lemma 2, we show:

Lemma 3 (Bounded Binomial Ratio)**.** *(ϵ, δ)-Noiseless privacy for $\boldsymbol{Count}(T) = \mathcal{C}$ holds if, with probability $1-\delta$:*

$$\Big|\ln(\Pr[\mathcal{B}(n-1,p) = a]) - \ln(\Pr[\mathcal{B}(n-1,p) = a-1])\Big| \leq \epsilon \tag{3.3}$$

where the input $T \in \{0,1\}^n$ is from $\mathcal{D}$ and is comprised of individual bits t_i for all $i \in [n]$. Further, each t_i is 1 with probability p and 0 otherwise. $\mathcal{C}$ is number of bits in $T \in \{0,1\}^n$ which are 1, and a is any element from the set $\{0,\ldots,n\}$, which represents the true value of $\mathcal{C}$.

Both the minuend and the subtrahend in Lemma 3 represent the probability of the count $\mathcal{C}$ being some fixed value a given the input bit t_i of data contributor $i \in [n]$. Since the input bit t_i is known and we assume the remaining input bits to be drawn via independent Bernoulli trials, we can represent the sum of those remaining bits as $\mathcal{B}(n-1,p)$; the number of Bernoulli trials decreases by 1 while the the probability p remains fixed. Then, bounding this ratio with ϵ, except with probability δ, we can achieve the (ϵ, δ)-noiseless privacy standard in Definition 4.

3.4.2 Describing "Good" ϵ and δ Values

Once we establish that (ϵ, δ)-noiseless privacy (Definition 4) can be met for the count function **Count**() according to Lemma 3, we concretely explain what the (ϵ, δ)-noiseless privacy parameters of ϵ and δ represent for **Count**(), specifically.

The parameter ϵ, just as in (ϵ, δ)-differential privacy (Definition 2), sets the multiplicative loss bound between two outcomes, namely the outcome of $\mathcal{C} = a$ for some $a \in \{0, \ldots, n\}$ given that data contributor $i \in [n]$ input a bit of 0 and the outcome of $\mathcal{C} = a$ given that same data contributor i input 1. Throughout our analysis, we assume ϵ values 0 to 5 are "strong," in accordance with guidance from the National Institute of Standards and Technology (NIST) on setting (ϵ, δ)-differential privacy parameters [7]. We refer often to these "strong" ϵ values, and, when we do so, we are referencing NIST's guidance.

NIST advocates for $0 < \epsilon < 5$ as offering "strong privacy" because Apple's differential privacy system [2], the U.S. Census Bureau's redistricting data [11], and Google's Community Mobility Reports [4] all use $\epsilon > 1$ and none of them have faced a successful privacy attack [7]. Importantly, these three examples represent a diversity of data types and data aggregation techniques, so it is reasonable for NIST to suggest that this guidance is broadly applicable.

While we illustrated in Section 3.3.2 that (ϵ, δ)-differential privacy (Definition 2) and (ϵ, δ)-noiseless privacy (Definition 4) are substantively different in four critical ways, we start our privacy analysis of **Count**() by assuming the NIST guidance [7] on ϵ translates to noiseless privacy. We do so precisely because ϵ has the same meaning in both definitions: it defines the multiplicative loss bound between the probabilities of two outcomes, or datasets. Of course, (ϵ, δ)-differential privacy takes the added step of injecting noise based on ϵ (described in Section 2.2), but it does so only to achieve this multiplicative loss bound in face of worst-case adversarial knowledge.

The parameter δ, just as in (ϵ, δ)-differential privacy (Definition 2), sets the tolerated failure probability for the privacy definition itself. As shown in Lemma 3, this tolerated failure probability is the maximum likelihood with which the absolute value of the difference in logs from Lemma 3 can fail to be bounded by ϵ. Based on work by Nissim et al. we assume an "application-specific" $\delta = 10^{-9}$ [22]. Different applications may have different failure tolerances, but for SCRAM we seek to set a high standard of privacy, and therefore a low failure rate. We believe the guidance from Nissim et al. ensures exactly this [22].

3.4.3 Setting up the Binomial Analysis

With a mathematical definition for noiseless privacy, a concrete model of the count function **Count**(), and constraints on both ϵ and δ, we are able to analyze the baseline privacy guarantees of (ϵ, δ)-noiseless privacy (Definition 4) for the count function **Count**(). Given our interest in privacy guarantees when n is small (e.g. $n < 20$ in SCRAM), a natural place to begin is analyzing how (ϵ, δ)-noiseless privacy guarantees evolve as n grows, for a fixed p. This is the first step toward showing when (ϵ, δ)-noiseless privacy is attainable, given specific values of δ, ϵ, n, and p.

Our results show that (ϵ, δ)-noiseless privacy is attainable for $n \geq 31$ using $0 < \epsilon < 5$, when $\delta = 10^{-9}$. This is a stronger privacy guarantee than that offered by Theorem 1, as it offers privacy for a concrete values of δ, not just an asymptotic privacy guarantee. And, fortunately, this privacy guarantee applies when n is small.

The Process to Compute the Minimum ϵ

Using the probability mass function (PMF) of the binomial random variable (defined in Section 3.3.1) $\mathcal{C}$, which represents the output of the count function, we developed computational tools in the Python programming language [20] to explicitly compute Equation 3.3 in Lemma 3. The general process for computing this quantity is as follows:

1. We specify n and p as constants to define the binomial random variable $\mathcal{B}(n, p)$, such that $\mathcal{C} = \mathcal{B}(n, p)$.

2. Given $\mathcal{C} = \mathcal{B}(n, p)$, we define the actual value of $\mathcal{C}$ as $\mathcal{C} = a$, and we iterate through all possible values $a \in \{0, ..., n\}$.

3. For each a, we compute the minimum ϵ required to satisfy the inequality in Lemma 3 given $\mathcal{C} = a$.

4. After computing an ϵ for every a, we repeatedly exclude the a value with the greatest associated ϵ and subtract its probability mass from δ, until the probability mass on the a in question exceeds the remaining value of δ. We then

take the maximum ϵ value over the a values that have not been excluded to determine the smallest ϵ possible to achieve (ϵ, δ)-noiseless privacy.

In order to explain the computation described in Step 3 above, we first recall the MLE contained in Lemma 3 (Equation 3.3):

$$\Big| \ln(\Pr[\mathcal{B}(n-1,p) = a]) - \ln(\Pr[\mathcal{B}(n-1,p) = a-1]) \Big| \leq \epsilon$$

As described in Step 3 above, we can compute the minimum ϵ necessary to satisfy the MLE from Lemma 3 for a given $\mathcal{C} = \mathcal{B}(n,p) = a$ using $\mathbf{PMF}(a; \mathcal{B}(n,p))$ (defined in Section 3.3.1) in the following way:

$$\Pr[\mathcal{B}(n-1,p) = a] = \mathbf{PMF}(a; \mathcal{B}(n-1,p)) \tag{3.4}$$

$$\Pr[\mathcal{B}(n-1,p) = a-1] = \mathbf{PMF}(a-1; \mathcal{B}(n-1,p)) \tag{3.5}$$

$$\epsilon = \Big| \log[\mathbf{PMF}(a; \mathcal{B}(n-1,p))] - \log[\mathbf{PMF}(a-1; \mathcal{B}(n-1,p))] \Big| \tag{3.6}$$

The application of this process is best illustrated through an example. Consider the task of computing the minimum ϵ required to satisfy (ϵ, δ=0)-noiseless privacy for $\mathcal{C} = \mathcal{B}(n = 20, p = 0.5)$. Executing the steps above for each possible $\mathcal{C} = a$ for all $a \in \{0, \ldots, n\}$, we plot the minimum value of ϵ required for each a in Figure 3-2.

As we can see in Figure 3-2, under the binomial distribution $\mathcal{B}(n = 20, p = 0.5)$, all $\mathcal{C} = a$ for $a \in \{1, ..., n-1 = 19\}$ can be bounded by ϵ values such that $0 < \epsilon < 5$. Again, these values of ϵ are considered by NIST to offer strong privacy [7]. However, in accordance with our intuition that $\mathcal{C} = 0$ and $\mathcal{C} = n$ can never be private (described in Section 3.4.1) we see in Figure 3-2 that the ϵ value required to bound these failure cases is infinite; there is no ϵ that can bound these failure cases.

The Private Range

With the understanding that there are certain values of $\mathcal{C} = a \in \{0, ..., n\}$ for which privacy cannot be attained, the question becomes, "Can we ever offer minimum privacy guarantees for all possible outputs of $\mathbf{Count}(T)$?" This brings us to the notion

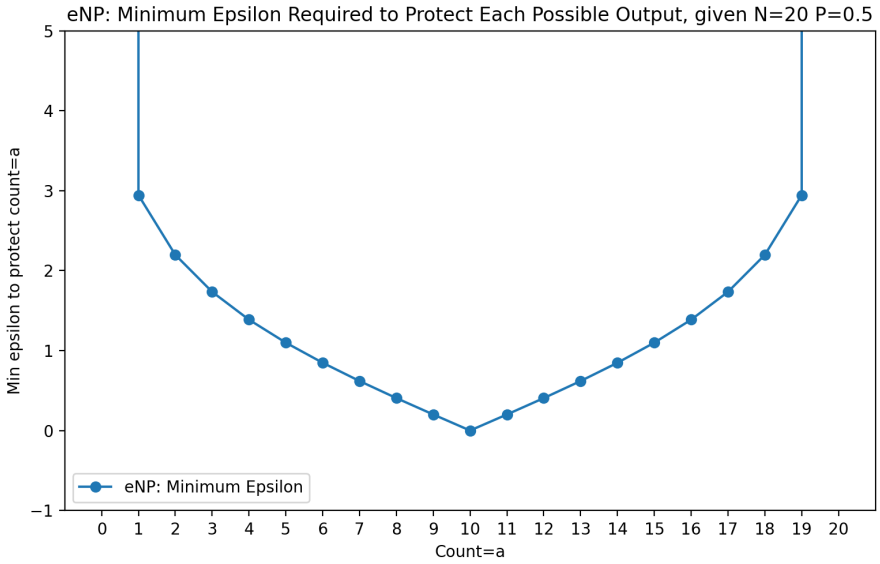

Figure 3-2: Minimum ϵ to Bound Each Count for $\mathcal{C} = \mathcal{B}(20, 0.5)$

of the private range, which we denote as **PrivateRange**. The private range refers to the range of outputs $\mathcal{C} = a \in \{0, ..., n\}$ for which privacy, noiseless privacy in this case, is achieved. The private range, of course, will never contain $\mathcal{C} = 0$ or $\mathcal{C} = n$, so in the best case **PrivateRange** $= \{1, \ldots, n-1\}$. More generally, the all counts $\mathcal{C} = a \notin$ **PrivateRange** must occur with combined probability less than or equal to δ.

Importantly, if we use (ϵ, δ=0)-noiseless privacy, the answer to the question of interest, "Can we ever offer minimum privacy guarantees for all possible outputs of **Count**(T)?" is no. Per Definition 4, (ϵ, δ=0)-noiseless privacy can fail with probability $\delta = 0$. Clearly, the failure cases of $\mathcal{C} = 0$ and $\mathcal{C} = n$ occur with probability greater than 0, but by setting $\delta = 0$ we are mandating that all outputs $\mathcal{C} = a \in \{0, ..., n\}$ fall into the private range, **PrivateRange**. When **PrivateRange** $= \{0, ..., n\}$, noiseless privacy is not attainable.

However, once we specify $\delta > 0$, we can indeed offer minimum privacy guarantees for the whole output space of counts, $\mathcal{C} = a \in \{0, ..., n\}$, under certain distributions, particularly those distributions where the failure cases occur with combined probability less than or equal to δ. In order to understand when privacy is attainable, we

compute combinations of ϵ, n and p that result in $\delta = 10^{-9}$ capturing the failure cases. Critically, since we assume δ to be predetermined and application-specific, we cannot adjust δ at will in order to obtain privacy.

3.4.4 Binomial Analysis: Explaining ϵ as n Grows

Establishing δ and ϵ: The Specific Case of $\mathcal{B}(\mathbf{n}, \mathbf{0.5})$

We begin our analysis by examining the relationships between ϵ, δ, n, and p by zooming in on the specific input distribution $\mathcal{B}(n, 0.5)$ for all $n \in \{3, \ldots, 100\}$. We set the lower bound on n at 3 because privacy in data aggregation only makes sense when there are more than two data providers; if $n = 2$, any data that is not owned by one party is necessarily the data of the other party. We set the upper bound on n at 100 because we are primarily interested in attaining privacy for small numbers of data providers and we have defined small as $n \leq 100$.

Setting $p = 0.5$ and iterating over all $n \in \{3, \ldots, 100\}$, we compute the minimum ϵ required to achieve $(\epsilon, \delta = 10^{-9})$-noiseless privacy. Since we have set $\delta > 0$, we allow our private range **PrivateRange** to exclude the failure cases under certain binomial distributions $\mathcal{B}(n, 0.5)$. In doing so, we find that once n is "large enough," privacy is achieved. We discuss the meaning of "large enough" in the analysis ahead.

Our results are shown in Figure 3-3. We explain these results by first examining three key sections of the plot. Then, we describe four high level trends in Figure 3-3 to explain each key section. We later show that these high level trends hold true even as we vary the value of p.

Three Key Sections

1. ($n < 31$): For these n values, there is no value of ϵ that can provide (ϵ, $\delta = 10^{-9}$)-noiseless privacy. This, as we will show shortly, is because the combined probability of the failure cases, $\mathcal{C} = 0$ and $\mathcal{C} = n$, is greater than δ, until $n \geq 31$.

2. ($31 \leq n < 37$): At $n = 31$, we have $\mathcal{C} = \mathcal{B}(n = 31, p = 0.5)$, and this is the first distribution for which we can achieve (ϵ, $\delta = 10^{-9}$)-noiseless privacy. For

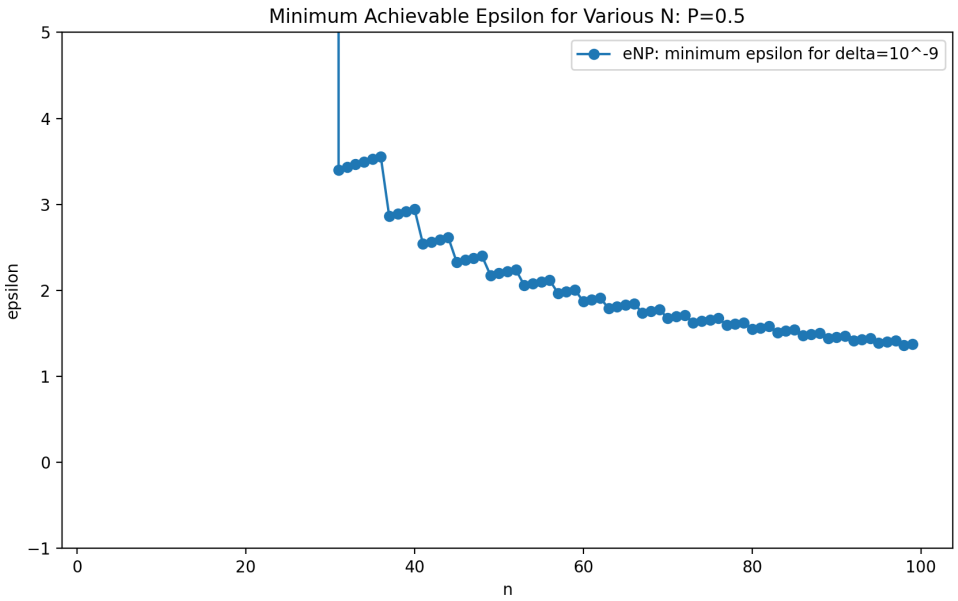

Figure 3-3: Minimum ϵ to Achieve Noiseless Privacy when $p = 0.5$

each count in this range, we observe that the ϵ value required to achieve privacy grows with n.

3. $(37 \leq n)$: We see a significant drop in the ϵ value required to achieve $(\epsilon, \delta = 10^{-9})$-noiseless privacy and then a continuation of the stair-step behavior, where the required ϵ value grows with n and drops, repeatedly.

Four High Level Trends

1. The most extreme count $\mathcal{C}$ in the **PrivateRange** sets the lower bound on the minimum ϵ required to satisfy Lemma 3. When we say "the most extreme count," we mean the count $\mathcal{C}$ in the **PrivateRange** that is farthest from the mean μ of the output distribution, as defined in Definition 4. In this analysis, the output distribution is $\mathcal{B}(n \in \{3, \ldots, 100\}, 0.5)$.

2. The more extreme the upper or lower bound on **PrivateRange**, the larger the ϵ value required to satisfy the equation in Lemma 3. Intuitively, this can be explained as the farther a count $\mathcal{C}$ is from the mean μ of $\mathcal{B}(n \in \{3, \ldots, 100\}, 0.5)$, the better an adversary is able to predict a participant's bit, using Bayesian

reasoning. If the lower or upper bound of the **PrivateRange** is far from the mean μ, the **PrivateRange** offers a lower standard of privacy because ϵ has to be bigger.

3. As n increases, the extreme counts become more extreme. In mathematical terms, this means that with larger n there is a larger output space $\mathcal{C} = a \in \{0, ..., n\}$, and therefore there is greater distance between the mean μ and count values $\mathcal{C}$ on the ends of distribution $\mathcal{B}(n \in \{3, \ldots, 100\}, 0.5)$.

4. As n increases, the extreme counts become less likely. In mathematical terms, this means that, at least for $\mathcal{C} = \mathcal{B}(n, p)$, the probability mass associated with the counts farthest from the mean μ necessarily have the least probability mass.

Equipped with an understanding of these trends, we can use them to explain the three sections of the graph in Figure 3-3.

For $(n < 31)$, the most extreme counts for $\mathcal{C} = \mathcal{B}(n < 31, 0.5)$ are the failure cases of $\mathcal{C} = 0$ and $\mathcal{C} = n$. As we explained, there is no ϵ to make these counts private, and though these failure cases are becoming less and less likely, it is not until $n = 31$ that they occur with combined probability less than δ.

For $(31 \leq n < 37)$, the failure cases finally occur with combined probability less than δ, so they are captured by our tolerated failure rate of $\delta = 10^{-9}$ and can be removed from the **PrivateRange**. Now, **PrivateRange** $= \{1, \ldots, n-1\}$. Once this is true, we can find an ϵ value for which (ϵ, $\delta = 10^{-9}$)-noiseless privacy is met. The reason the required ϵ value increases until a drop off at $n = 37$ is explained by the fact that the lower and upper bounds of the **PrivateRange**, which are now $\mathcal{C} = 1$ and $\mathcal{C} = n - 1$, occur with too high probability to be captured by the failure rate δ. In other words, δ is only large enough to capture the failure cases of $\mathcal{C} = 0$ and $\mathcal{C} = n$, but no other outputs occur with small enough probability to be captured by δ, as well. At the same time, the lower and upper bounds on the **PrivateRange** are stretching farther and farther from the mean of the binomial distribution as n grows. Consequently, the ϵ value required by Lemma 3 must grow.

For $(37 \leq n)$, the stair-step behavior is explained by similar logic. At every

"step down," the most extreme counts in the **PrivateRange**, which were previously setting the lower bound on ϵ, finally occur with combined probability less than δ, meaning they are captured by the failure rate δ. Once they are captured by the failure rate δ, they can be removed from the **PrivateRange**. The most extreme counts in this updated, and specifically smaller, **PrivateRange** are slightly less extreme and therefore require a smaller ϵ to bound them. As n increases, these extremes become more extreme, requiring larger ϵ until the next "step down."

Clearly, it is the case that, given a set probability p, (ϵ, δ)-noiseless privacy can only be achieved when n is "large enough." So, what constitutes "large enough?" Why is $n = 31$ large enough when $p = 0.5$? These questions are answered, in short, by observing that until n reaches a certain value, the failure cases of $\mathcal{C} = 0$ and $\mathcal{C} = n$ occur with greater combined probability than δ, as shown in Figure 3-4. This means that these privacy-less counts occur with too high probability to be captured by the allowed failure rate δ.

Establishing ϵ and δ: The General Case of $\mathcal{B}(\mathbf{n}, \mathbf{0} < \mathbf{p} < \mathbf{1})$

This analysis is supported by the following logarithmic plots (Figure 3-4, Figure 3-6, and Figure 3-7) that depict how the combined probability mass associated with the two failure cases shrinks as n increases. As seen in Figure 3-4, it is not until $n = 31$, the same n value for which we get our first finite ϵ in Figure 3-3, that the probability mass of the failure cases can be captured by δ. Naturally, we do this same analysis for varying values of p to confirm that these high level trends hold across various binomial distributions, and they do.

For $p = 0.75$ in Figure 3-5, we see that we get our first finite ϵ value only when $n = 72$. For $n < 72$ we get no privacy at all, which is significantly worse than the guarantees we have under $\mathcal{B}(n = n, p = 0.5)$. Figure 3-6 explains the reason there is no privacy for $n < 72$ is that the combined probability mass of the failure cases is greater than δ up until $n = 72$. For $p = 0.9$, we do not get a finite ϵ value for any $n \leq 100$. Figure 3-7 confirms this by showing the combined probability mass of the failure cases is larger than δ for all $n \leq 100$.

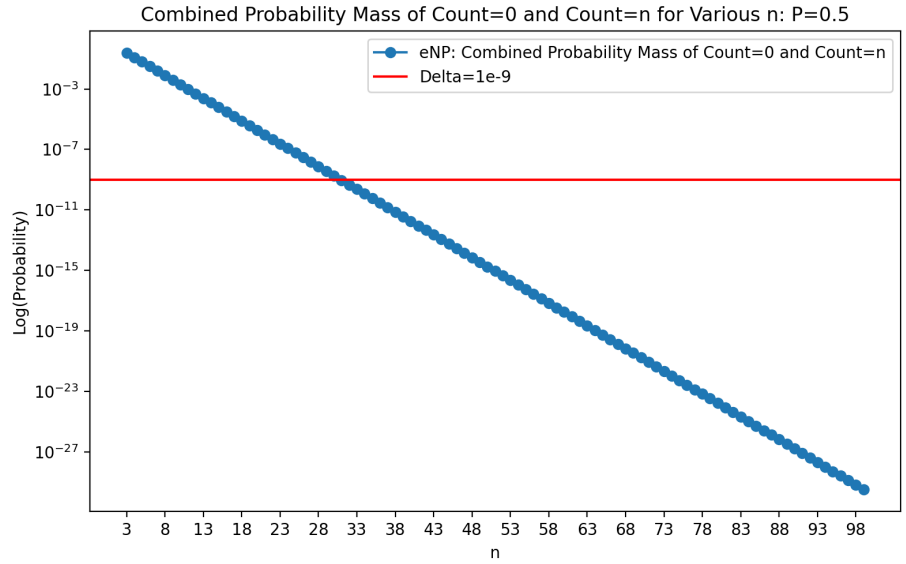

Figure 3-4: Noiseless: Combined Probability Mass of Failure Cases when $p = 0.5$

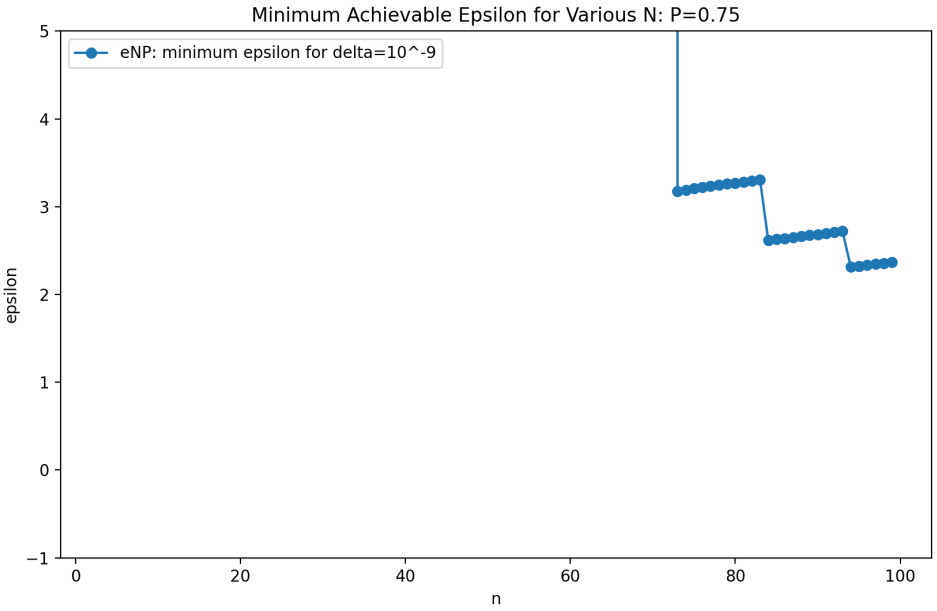

Figure 3-5: Minimum ϵ to Achieve Noiseless Privacy when $p = 0.75$

Together, Figures 3-3 through 3-7 reveal that the more p deviates from 0.5, the larger the n required to achieve noiseless privacy. As we can see, in the cases with extreme p values, the combined probability mass of the failure cases remains larger

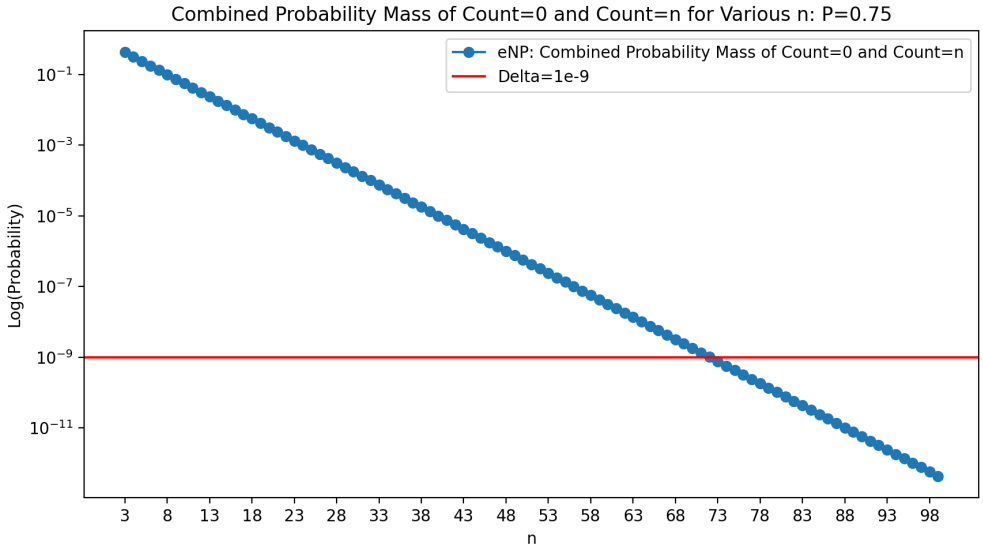

Figure 3-6: Noiseless: Combined Probability Mass of Failure Cases when $p = 0.75$

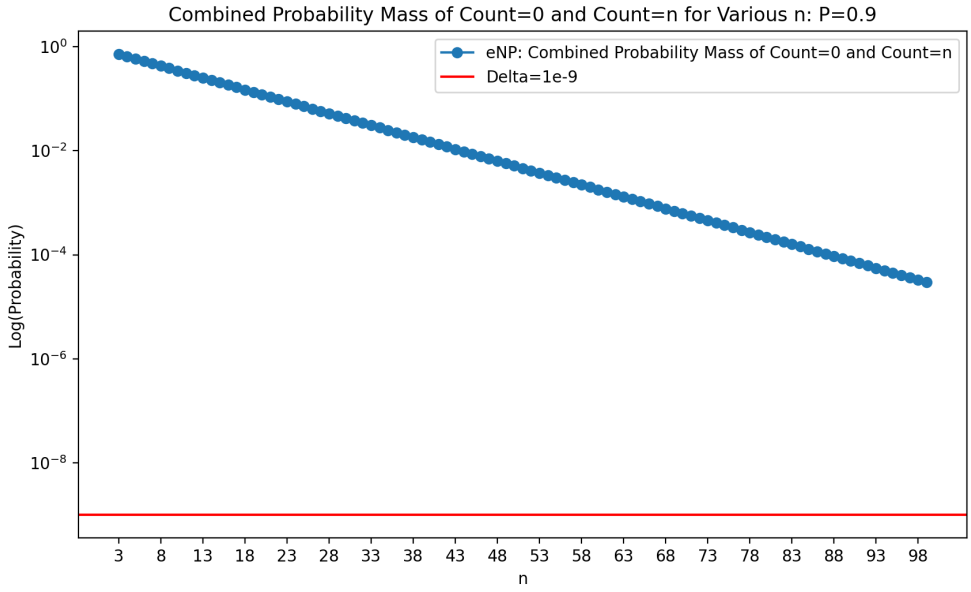

Figure 3-7: Noiseless: Combined Probability Mass of Failure Cases when $p = 0.9$

for longer. This means that the failure cases are not captured by δ until n reaches a higher value than under $\mathcal{B}(n, 0.5)$. In particular, when $p = 0.9$, we expect $\mathcal{C} = (0.9 \cdot n)$. While seeing $\mathcal{C} = 0$ would be very unlikely, seeing $\mathcal{C} = n$ would actually occur with significant probability. For this reason, as p tends toward 1, the increasing likelihood of seeing $\mathcal{C} = n$ dominates the combined probability of failure. As p tends toward 0, the increasing likelihood of $\mathcal{C} = 0$ dominates the combined probability of failure. This point is further illustrated by Figure 3-8, where p is skewed toward 0 and the probability mass of $\mathcal{C} = 0$ dominates the combined probability of failure. In Figure 3-9, where p is skewed toward 1, the probability mass of $\mathcal{C} = n$ dominates the combined probability of failure.

3.4.5 Failures of Noiseless Privacy

In summary, our analysis of the behavior of δ and ϵ across $n \in \{0, \ldots, n\}$ and across varying p, such that $0 < p < 1$, illuminates the central problem with noiseless privacy: No matter how ϵ is set, for a fixed δ, privacy cannot be achieved when n falls below a certain threshold. This threshold is set by the interaction between the distribution of the input data $\mathcal{D}$, the application-specific failure rate δ, and the **PrivateRange**. Privacy can only be achieved when δ is greater than the combined probability mass of $\mathcal{C} = 0$ and $\mathcal{C} = n$ under the given distribution of input data. Only when this condition is met does the **PrivateRange** exclude the failure cases, which is necessary to provide privacy.

Assuming $\delta = 10^{-9}$, privacy cannot be achieved for $n < 31$ under the best-case input distribution of $p = 0.5$, assuming each input is a repeated and independent **Bernoulli**(p). Critically, there are many data aggregation applications, including SCRAM, where privacy guarantees are needed for single-digit n, but noiseless privacy does not offer such a guarantee. We introduce deniable privacy to solve this problem.

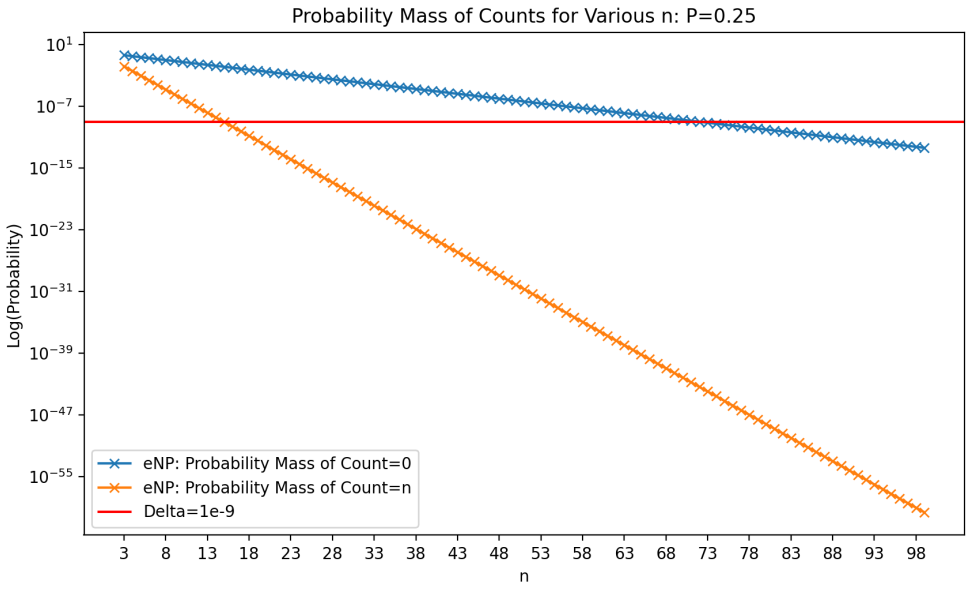

Figure 3-8: Noiseless: Probability Mass of Each Failure Case when $p = 0.25$

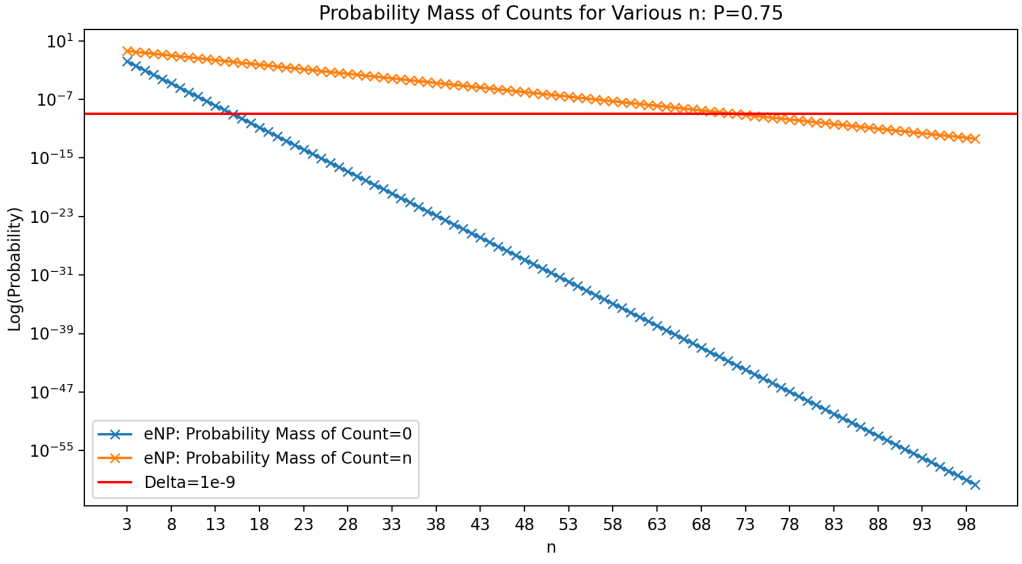

Figure 3-9: Noiseless: Probability Mass of Each Failure Case when $p = 0.75$

3.5 Deniable Privacy

In Section 3.4.5, we observed that the failure cases of $\mathcal{C} = 0$ and $\mathcal{C} = n$ drive the lack of noiseless privacy (Definition 4) for the count function **Count**() when $n < 31$, given our application-specific $\delta = 10^{-9}$. In an effort to address this, we naturally ask the question, "What happens if we modify the count function itself, such that it never releases counts of 0 or n?" This leads us to our central contribution of deniable privacy.

3.5.1 Defining the Deniable Count Function

Deniable privacy is a privacy mechanism that relies on a modified version of the count function **Count**() that we denote as $\textbf{Count}_{deniable}()$. We denote the random variable that represents that actual value of $\textbf{Count}_{deniable}(T)$ for some input vector T from distribution $\mathcal{D}$ as $\mathcal{C}_{deniable}$. We define $\textbf{Count}_{deniable}()$ as follows:

Definition 6 (Deniable Count Function)**.**

$$\boldsymbol{Count}_{deniable}(T) = \begin{cases} 1, \ \textit{if } \boldsymbol{Count}(T) \in \{0, 1\} \\ \boldsymbol{Count}(T), \ \textit{if } \boldsymbol{Count}(T) \in \{2, \ldots, n-2\} \\ n-1, \ \textit{if } \boldsymbol{Count}(T) \in \{n-1, n\} \end{cases}$$

where T is some input string input string $\langle t_1 t_2 \ldots t_n \rangle \in \{0,1\}^n$ and ***Count***() *is the count function as defined in Section 3.3.3.*

Assuming each input bit t_i for $i \in [n]$ is sampled as a repeated and independent Bernoulli trial, like we assumed throughout our analysis of (ϵ, δ)-noiseless privacy, the output distribution of $\textbf{Count}_{deniable}()$ is very similar to that of **Count**(), which was presented in Figure 3-1. However, there are two differences:

1. The probability mass assigned to $\mathcal{C}_{deniable} = 1$ is the combined probability mass of $\mathcal{C} = 0$ and $\mathcal{C} = 1$.

2. The probability mass assigned to $\mathcal{C}_{deniable} = n - 1$ is the combined probability mass of $\mathcal{C} = n - 1$ and $\mathcal{C} = n$.

Provided both functions operate over input data that follows our assumed input distribution of repeated and independent Bernoulli trials, the similarity of the output distribution of $\textbf{Count}_{deniable}()$ to that of the $\textbf{Count}()$ can be seen in Figure 3-10, while the difference in assignment of probability mass can be seen with the logarithmic scale in Figure 3-11.

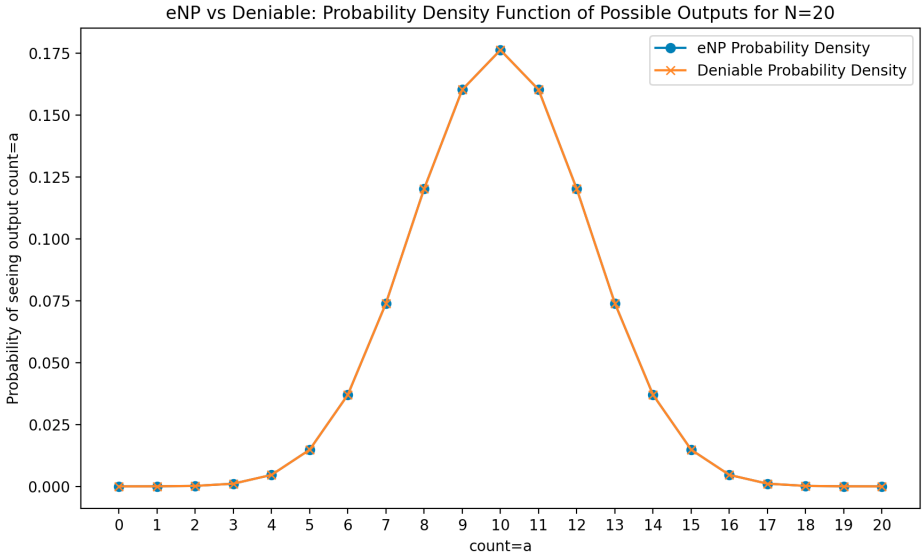

Figure 3-10: Output Distributions: Deniable Count vs. Count

3.5.2 Defining Deniable Privacy

We define deniable privacy as satisfying noiseless privacy via $\textbf{Count}_{deniable}()$, instead of $\textbf{Count}()$. We now show that $\textbf{Count}_{deniable}()$ satisfies the definition of (ϵ, δ)-noiseless privacy (Definition 4) in the same fashion as we showed $\textbf{Count}()$ satisfies the definition of (ϵ, δ)-noiseless privacy. Specifically, we adapt Lemmas 1-3 to use $\textbf{Count}_{deniable}()$, instead of $\textbf{Count}()$. In doing so, we produce Lemmas 4-6.

Lemma 4 (Deniable Count MLE Ratio ([10]))**.** *Let T be from the distribution $\mathcal{D}$ where each bit is 1 with probability p. Let $\textbf{Count}_{deniable}(T) = \mathcal{C}_{deniable}$ be the count of*

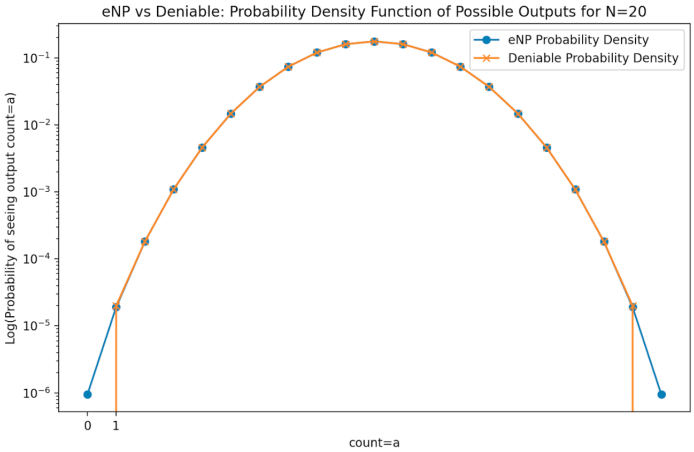

Figure 3-11: Output Distributions: Deniable Count vs. Count (Logarithmic Scale)

the number of bits in $T \in \{0,1\}^n$ *which are 1 and let* a *be any element from the set* $\{1,\ldots,n-1\}$. $\boldsymbol{Count}_{deniable}(T)$ *satisfies* (ϵ,δ)*-noiseless privacy if for all* $i \in [n]$ *the following ratio holds with probability* $1-\delta$*:*

$$\left|\ln(\Pr[\mathcal{C}_{deniable}=a|t_i=0]) - \ln(\Pr[\mathcal{C}_{deniable}=a|t_i=1])\right| \leq \epsilon \tag{3.7}$$

Because $\textbf{Count}_{deniable}()$ is a deterministic function, just like $\textbf{Count}()$, they are interchangeable from the perspective of noiseless privacy (Definition 4). We therefore substitute $\textbf{Count}()$ in Lemma 1 for $\textbf{Count}_{deniable}()$ in Lemma 4. Of course, we must update the valid output space to be $\{1,\ldots,n-1\}$.

Lemma 5 ((Deniable Count)-Binomial Equivalence ([10], proof of Theorem 5))**.**

$$\Pr[\mathcal{C}_{deniable}=a|t_i=0] = \Pr[\mathcal{B}(n-1,p)=a]$$
$$\Pr[\mathcal{C}_{deniable}=a|t_i=1] = \Pr[\mathcal{B}(n-1,p)=a-1]$$

where T *is from the distribution* $\mathcal{D}$ *where each bit* t_i *is 1 with probability* p*.* $\mathcal{C}_{deniable} = \boldsymbol{Count}_{deniable}(T)$ *is the count of number of bits in* $T \in \{0,1\}^n$ *which are 1 and* a *is any element from the set* $\{1,\ldots,n-1\}$*.*

Again, because $\textbf{Count}_{deniable}()$ is interchangeable with $\textbf{Count}()$ from the perspective of noiseless privacy (Definition 4), we simply substitute $\textbf{Count}()$ in Lemma 2 for

$\mathbf{Count}_{deniable}()$ in Lemma 5 and update the valid output space to be $\{1, \ldots, n-1\}$.

In Section 3.4 we produced Lemma 3 by combining Lemma 1 and Lemma 2. In this section, we produce Lemma 6 by combining Lemma 4 and Lemma 5.

Lemma 6 (Bounded Binomial Ratio using Deniable Count Function)**.** (ϵ, δ)*-Noiseless privacy for* $\boldsymbol{Count}_{deniable}(T) = \mathcal{C}_{deniable}$ *holds if, with probability* $1 - \delta$*:*

$$\left| \ln(\Pr[\mathcal{B}(n-1, p) = a]) - \ln(\Pr[\mathcal{B}(n-1, p) = a-1]) \right| \leq \epsilon \tag{3.8}$$

where the input $T \in \{0,1\}^n$ *is from* $\mathcal{D}$ *and is comprised of individual bits* t_i *for all* $i \in [n]$*. Further, each* t_i *is* 1 *with probability* p *and* 0 *otherwise.* $\mathcal{C}_{deniable} = \boldsymbol{Count}_{deniable}(T)$ *is number of bits in* $T \in \{0,1\}^n$ *which are* 1*. And,* a *is any element from the set* $\{1, \ldots, n-1\}$*, which represents the true value of* $\mathcal{C}_{deniable}$*.*

Crucially, deniable privacy is not a new definition of privacy, per se. Instead it is a mechanism to achieve (ϵ, δ)-noiseless privacy by explicitly adding the minimal amount of noise necessary to mask the failure cases of the general count function $\mathbf{Count}()$. Deniable privacy is a small yet powerful adjustment to $\mathbf{Count}()$ such that (ϵ, δ)-noiseless privacy can be satisfied without requiring δ to capture the failure cases. Equally important, deniable privacy provides possible innocence [21] (Definition 1) because given $\mathcal{C}_{deniable} = a \in \{1, \ldots, n-1\}$, no one can claim with certainty that a particular participant input 0 or 1 using just the output $\mathcal{C}_{deniable} = a$. As an example, if $\mathcal{C}_{deniable} = n - 1$, each participant can claim that she was the one individual who input 0, even though this is quite unlikely. Using deniable privacy, of course, is equivalent to adding noise, but the noise that is added is limited in scope and it is only applied to the most unlikely of outputs. All other counts are noise free. In contrast with differential privacy, where noise is added to all outputs, deniable privacy presents clear utility benefits by reducing the number of noisy outputs it will produce.

3.5.3 Applying Deniable Privacy

Analyzing this new definition via similar computational tools also contained in our GitHub repository [20], we obtain improvements in the privacy guarantees we can

offer to SCRAM, or any secure data aggregation platform implementing **Count**(T) for small n. Our results are shown in Figure 3-12.

Specifically, as illustrated by our results in Figure 3-12, deniable privacy offers privacy for computations of any size $n > 2$, whereas the noiseless approach only offers privacy for $n > 30$, under a best case distribution of inputs. Further, the guarantees offered by deniable privacy, as defined by ϵ and δ, comply with NIST's guidance [7] on strong ϵ and the Nissim et al. guidance [22] on δ.

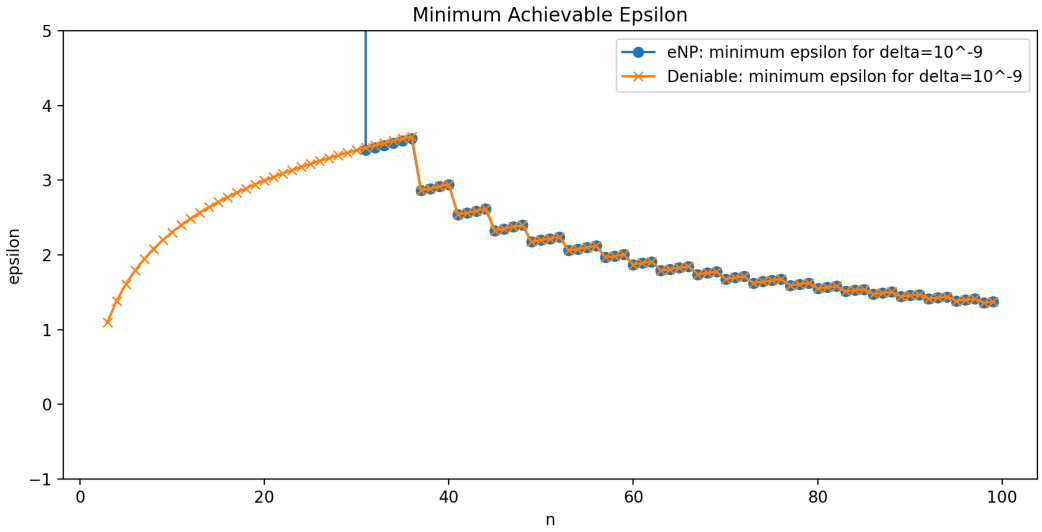

Figure 3-12: Noiseless vs. Deniable Privacy: Minimum Achievable ϵ when $p = 0.5$

We now explain Figure 3-12 in detail. First, we point out key sections of the plot. Then, we offer explanations of the behavior in each section using high level trends, just as we did in Section 3.4. As we go through our analysis, we designate one privacy approach as having "better performance" than another approach when it is able to offer privacy guarantees for a smaller ϵ value. The value of δ remains set at 10^{-9}, just as it was in our assessment of noiseless privacy from Section 3.4.

Three Key Sections

1. $(3 \leq n < 31)$: In this section, we see there is no finite ϵ that can provide (ϵ, δ)-noiseless privacy when using **Count**() but there are finite ϵ values that

can provide (ϵ, δ)-noiseless privacy when using $\textbf{Count}_{deniable}()$. As we can see, the ϵ value required by deniable privacy is growing with n.

2. $(31 \leq n < 37)$: In this section, we see that (ϵ, δ)-noiseless privacy can be achieved by both $\textbf{Count}()$ and $\textbf{Count}_{deniable}()$. Further, the ϵ value required by $\textbf{Count}()$ to satisfy (ϵ, δ)-noiseless privacy is actually smaller than that required by $\textbf{Count}_{deniable}()$ in this range. This represents a small range of n for which the noiseless approach actually performs better than the deniable approach. Even so, the difference in these ϵ values is approximately 0.02, which may not be significant depending on the data aggregation application.

3. $(37 \leq n)$: This section occurs after the first "step down." The performance of the two privacy approaches, noiseless and deniable, is the same here because their minimum achievable ϵ values are equivalent.

To explain these three key sections, we offer high level trends that mirror those we previously enumerated.

Four High Level Trends

1. The most extreme count in the **PrivateRange** sets the lower bound on the minimum ϵ required to satisfy the equation in Lemma 6. When we say "the most extreme count," we mean the count $\mathcal{C}_{deniable} = a$ for $a \in \textbf{PrivateRange}$ that is farthest from the mean μ of the random variable $\mathcal{C}_{deniable}$

2. The more extreme the upper or lower bound on **PrivateRange**, the larger the ϵ value must be to satisfy the equation in Lemma 6. Mathematically, this means that the farther $\mathcal{C}_{deniable} = a$ is from the mean μ of the random variable $\mathcal{C}_{deniable}$, the better an adversary is able to predict a participant's input bit, using Bayesian reasoning.

3. As n increases, the extreme counts become more extreme. In mathematical terms, this means that with larger n there is a larger output space, $\mathcal{C}_{deniable} =$

$a \in \{1, ..., n-1\}$ in the case of deniable privacy, and therefore there is greater distance between the mean μ of $\mathcal{C}_{deniable}$ and the extreme counts.

4. As n increases, the extreme counts become less likely. In mathematical terms, this means that the probability mass associated with the counts farthest from the mean μ of $\mathcal{C}_{deniable}$ necessarily have the least probability mass.

For $(3 \leq n < 31)$, we can explain the fact that ϵ increases with n because, similar to the trends in noiseless privacy, the extreme counts of $\mathcal{C}_{deniable} = 1$ and $\mathcal{C}_{deniable} = n - 1$ set the minimum achievable ϵ but these counts are becoming more extreme as n grows. In other words, they are stretching farther from the mean μ.

For $(31 \leq n < 37)$, we can explain the slight difference in performance between the noiseless approach and the deniable approach by recognizing that the probability masses of the extreme cases differ slightly between the two for this range of n. Specifically, we remember that (ϵ, δ)-noiseless privacy was first achieved by **Count**() at $n = 31$ because, at this n-value, $\mathcal{C} = 0$ and $\mathcal{C} = n$ were first captured by the failure rate δ. In the case of deniable privacy, $\mathcal{C}_{deniable} = 0$ and $\mathcal{C}_{deniable} = n$ are not valid outputs, and their probability masses are added to the probability masses already at $\mathcal{C}_{deniable} = 1$ and $\mathcal{C}_{deniable} = n-1$, respectively. So, with lower probability mass on the extreme counts of $\mathcal{C} = 1$ and $\mathcal{C} = n - 1$, the noiseless approach is able to outperform deniable privacy.

For $(37 \leq n)$, we can explain the familiar stair-step behavior using the same logic as before. What is important to address, however, is the fact that the noiseless approach and the deniable approach have identical performance in this range because δ captures the same number of extreme counts in each instance. More simply, **PrivateRange** is equivalent for deniable privacy and noiseless privacy when $n \geq 37$.

3.5.4 Assessing Deniable Privacy Across p-Values

Just as before, we would like to see how deniable privacy performs as the p-value used to parameterize the input distribution skews away from $p = 0.5$. Figure 3-13 offers a performance comparison between noiseless privacy and deniable privacy

when $p = 0.75$, which by symmetry is the same as the performance comparison when $p = 0.25$. Figure 3-14 offers a performance comparison between noiseless privacy and deniable privacy when $p = 0.9$, which by symmetry is the same as the performance comparison when $p = 0.1$. Importantly, no matter what the skew, deniable privacy offers privacy using NIST's strong ϵ values for all $n > 2$, given $\delta = 10^{-9}$. In contrast, under noiseless privacy, a p skewed away from $p = 0.5$ means larger n is required to achieve privacy. Under deniable privacy, the skew of p does not impact whether privacy can be achieved.

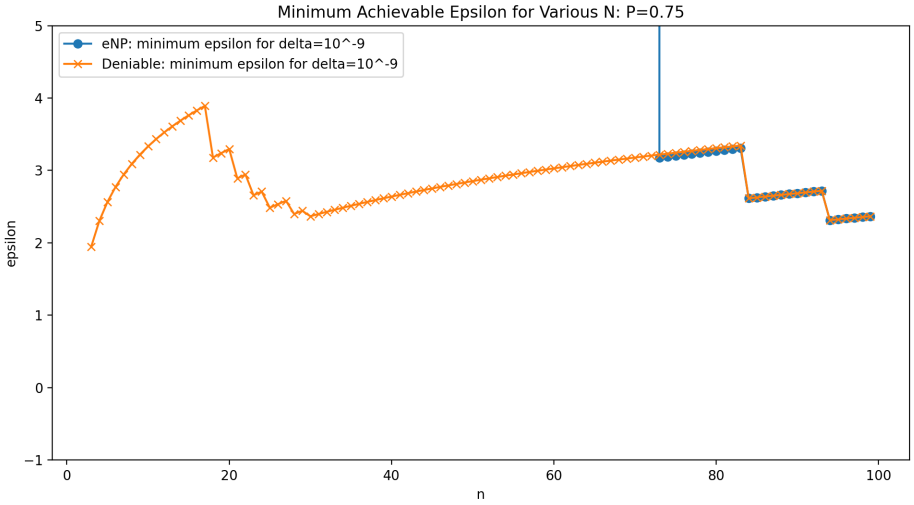

Figure 3-13: Noiseless vs. Deniable Privacy: Minimum Achievable ϵ when $p = 0.75$

3.6 k-Deniable Privacy

3.6.1 Defining the k-Deniable Count Function

We now extend the notion of deniable privacy by defining k-deniable privacy, which is simply a generalized notion of deniable privacy that relies on what we call the k-deniable count function. We denote the k-deniable count function as $\textbf{Count}_{k-deniable}()$. We denote the random variable that represents the value of $\textbf{Count}_{k-deniable}(T)$ for some input vector T under distribution $\mathcal{D}$ as $\mathcal{C}_{k-deniable}$. We define $\textbf{Count}_{k-deniable}()$ in Definition 7.

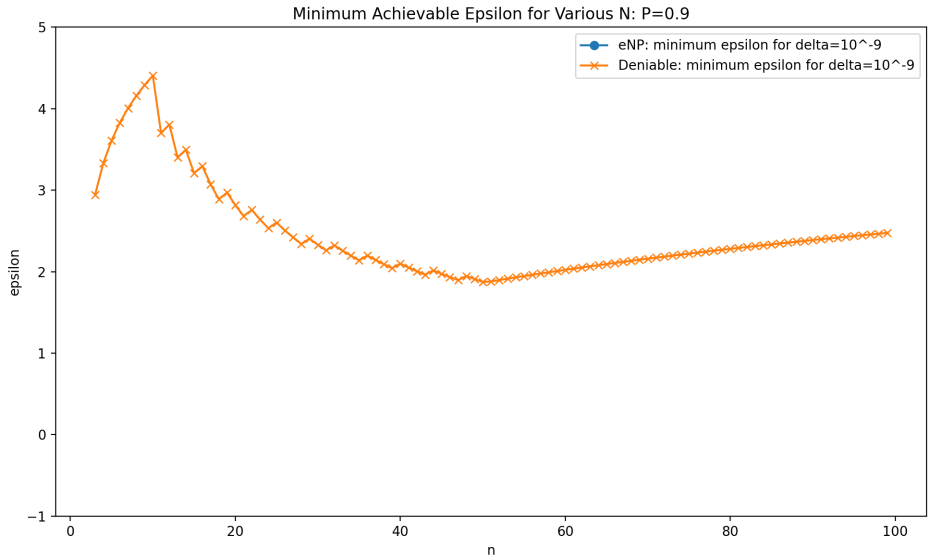

Figure 3-14: Noiseless vs. Deniable Privacy: Minimum Achievable ϵ when $p = 0.9$ *Note:* ***Count***() *does not satisfy noiseless privacy for* $n \leq 100$ *when* $p = 0.9$ *and* $\delta = 10^{-9}$*, so it does not appear in this plot.*

Definition 7 (k-Deniable Count Function)**.**

$$\textbf{\textit{Count}}_{k-deniable}() = \begin{cases} k, \ \textit{if } \textbf{\textit{Count}}(T) \in \{0, \ldots, k\} \\ \textbf{\textit{Count}}(T), \ \textit{if } \textbf{\textit{Count}}(T) \in \{k+1, \ldots, n-k-1\} \\ n-k, \ \textit{if } \textbf{\textit{Count}}(T) \in \{n-k, \ldots, n\} \end{cases}$$

where T *is some input string input string* $\langle t_1 t_2 \ldots t_n \rangle \in \{0,1\}^n$ *and* ***Count***() *is the count function as defined in Section 3.3.3.*

There are two key differences between the output distribution of $\textbf{Count}_{k-deniable}()$ and that of **Count**():

1. The probability mass assigned to $\mathcal{C}_{k-deniable} = k$ for is the combined probability mass of $\{\mathcal{C} = 0, \ldots, \mathcal{C} = k\}$.

2. The probability mass assigned to $\mathcal{C}_{k-deniable} = n-k$ is the combined probability mass of $\{\mathcal{C} = n-k, \ldots, \mathcal{C} = n\}$.

In a similar fashion to the deniable count function $\mathbf{Count}_{deniable}()$, any $\mathcal{C} = \{0, ..., k\}$ will be output as $\mathcal{C}_{k-deniable} = k$ and any $\mathcal{C} = \{n-k, ..., k\}$ will be output as $\mathcal{C}_{k-deniable} = n - k$.

3.6.2 Analyzing k-Deniable Privacy

This generalization to k-deniable privacy is helpful primarily because certain applications may wish to mask more than just the inherent failure cases of the count function $\mathbf{Count}()$, namely $\mathcal{C} = 0$ and $\mathcal{C} = n$. Take the immediate example of defending against other participants, who know their inputs. In the noiseless approach, if $\mathcal{C} = 1$ then the individual who input 1 knows with certainty the inputs of all other data providers. In the deniable approach, if $\mathcal{C}_{deniable} = 1$ then this is also true, provided some individual did in fact input 1. However, under 2-deniable privacy, this problem is solved. Even if a singular individual input 1, she will see $\mathcal{C}_{k-deniable} = 2$ and be uncertain if she is is the only individual who input 1. Under k-deniable privacy for $k \geq 2$, data contributors must collude in order to breach the standard of possible innocence (Definition 1, [21]).

To protect against scenarios like the one described above, Figure 3-15 and our GitHub repository [20] offer analytical tools to quantify privacy for different values of k. Our results are shown in Figure 3-15. The key takeaway from k-deniable privacy, as shown in Figure 3-15, is that larger k-values enable smaller ϵ values when n is very small (i.e. $n < 40$). This is because the most extreme output that can be released by $\mathbf{Count}_{k-deniable}(T)$, for some input bit string T, becomes less extreme as k increases. All in all, for a data aggregator who wishes to mask more than the inherent failure cases of $\mathbf{Count}()$, k-deniable privacy is a reasonable mechanism to use, especially if she is confident that the data contributors will not collude.

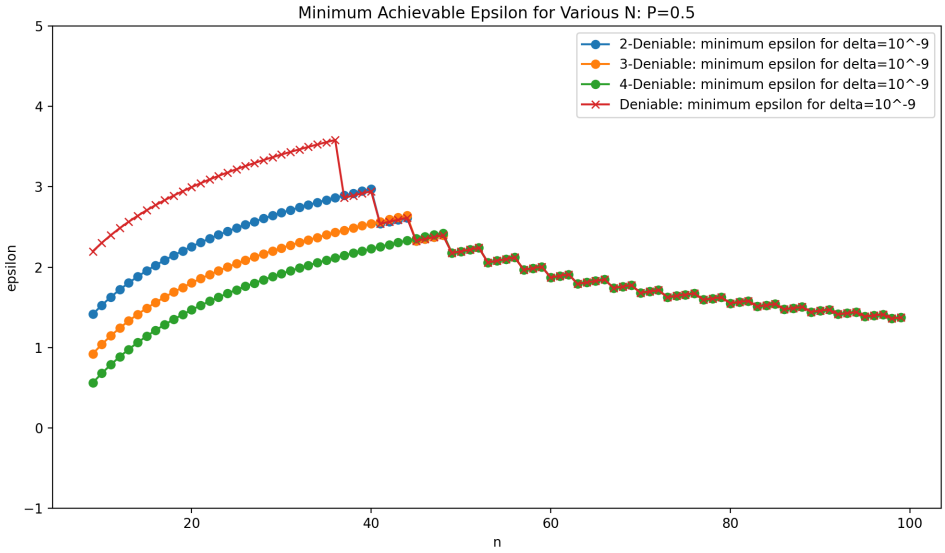

Figure 3-15: k-Deniable Privacy when $p = 0.5$

Chapter 4

Conclusion

In this work, we have shown three key results. First, we have applied the Bhaskar et al. notion of noiseless privacy [10] to a real-world data aggregation platform, namely SCRAM [13], and proven that privacy of the count function can be achieved for relatively small numbers of data contributors ($n \geq 31$) without introducing any noise to the count function's output. Second, we have defined a new privacy mechanism, deniable privacy, that ensures privacy of a modified count function for any number of data contributors ($n > 2$), while introducing the minimum amount of noise necessary to deliver possible innocence (Definition 1, [21]) to every data contributor. Third, we have presented a generalized notion of deniable privacy, called k-deniable privacy, that ensures possible innocence (Definition 1, [21]) for all data contributors in a count computation of any size, even in the face of collusion between data contributors.

Throughout this work, we have also made an effort to explain the differential privacy (Definition 2, [15]) parameters of ϵ and δ in intuitive terms, offering the notion of the **PrivateRange** as a new tool to describe the interaction between δ and the underlying distribution $\mathcal{D}$ of the input data. It is our hope that this analysis advances efforts to make privacy understandable and intuitive for practitioners whose primary objective is not necessarily privacy.

4.1 Recommendations

We offer one broad recommendation to privacy-aware data owners: Seek privacy guarantees that are intuitive and explainable over those that are described only by their parameters. This is easier said than done, as much of the privacy literature operates in terms of ϵ and δ. However, we believe the value of our work on deniable privacy is best described not in terms of the particular ϵ and δ values we computed, but instead in terms of the end goal of possible innocence (Definition 1, [21]). With this in mind, we feel there is substantial work to be done within the privacy field to translate privacy defined by parameters to privacy defined by outcomes. Bridging the gap between privacy parameters and privacy outcomes will enable data owners to make better-informed privacy decisions with their data.

To further illustrate this point, NIST's guidance [7] suggests certain ϵ values are strong because Google or Apple use them, but our analysis shows that the privacy afforded by a particular set of privacy parameters depends heavily on the particular application, including the specific function that is being computed. Consequently, a data owner should not equate standardized privacy parameters with standardized privacy outcomes.

We also offer one primary recommendation for data aggregators: Work closely with your data providers to define what they intend to learn from the aggregate and what they intend to keep private throughout the computation. As we have experienced with SCRAM, without an explicit understanding of these two information categories, it is nearly impossible to make the optimal trade-off between privacy and utility. However, once constraints are explicitly defined (e.g. SCRAM's constraints as defined in Section 2.1.2) it becomes easier to design and deploy privacy mechanisms, like deniable privacy, tailored to the privacy needs of the application.

4.2 Future Work

There are four avenues for future work that we believe require immediate attention. First, while this work offers intuition behind the meaning and effect of δ by tying it to the notion of the **PrivateRange**, it does not sufficiently explore the meaning of ϵ. Does $0 < \epsilon < 5$ really provide strong privacy as stated by NIST [7]? If so, under what circumstances? One approach to answering these questions would be to define a method that translates ϵ into an adversary's likelihood of success in predicting some well-defined private attribute. We see this as a critical next step in our research. Second, we believe it is necessary to analyze noiseless privacy [10] and deniable privacy under different distributions of the inputs. For example, though we assumed all input bits to SCRAM's count function to be independent and repeated Bernoulli trials, it is quite possible that they are correlated, and perhaps not even Bernoulli. Understanding how privacy guarantees change in the face of correlation and varying input distributions is essential. Third, the count function is not the only useful data aggregation primitive. Future work should certainly explore the privacy properties of other data aggregation functions, especially the sum function. Fourth, and perhaps most importantly, we believe the impact of auxiliary information on the privacy of secure data aggregation functions must be analyzed in greater detail. Our analysis in this work assumed the absence of auxiliary information from the computation environment, but in practice this is not a realistic assumption. As we discussed, differential privacy is enticing precisely because it protects against arbitrary auxiliary information, but we believe substantial work is left to be done in defining less noisy mechanisms that protect against specific classes of auxiliary information. Work across these four avenues will certainly advance privacy offerings in the field of secure data aggregation.

Bibliography

[1] Apple differential privacy technical overview.

[2] Learning with privacy at scale - apple.

[3] Cyber: Getting to grips with a complex risk, 2017.

[4] Ahmet Aktay, Shailesh Bavadekar, Gwen Cossoul, John Davis, Damien Desfontaines, Alex Fabrikant, Evgeniy Gabrilovich, Krishna Gadepalli, Bryant Gipson, Miguel Guevara, Chaitanya Kamath, Mansi Kansal, Ali Lange, Chinmoy Mandayam, Andrew Oplinger, Christopher Pluntke, Thomas Roessler, Arran Schlosberg, Tomer Shekel, Swapnil Vispute, Mia Vu, Gregory Wellenius, Brian Williams, and Royce J Wilson. Google covid-19 community mobility reports: Anonymization process description (version 1.1), 2020.

[5] Tania M. Alarcon Falconi, Bertha Estrella, Fernando Sempértegui, and Elena N. Naumova. Effects of Data Aggregation on Time Series Analysis of Seasonal Infections. *International Journal of Environmental Research and Public Health*, 17(16), 2020.

[6] Anwaar Ali, Junaid Qadir, Raihan ur Rasool, Arjuna Sathiaseelan, Andrej Zwitter, and Jon Crowcroft. Big data for development: applications and techniques. *Big Data Analytics*, 1(1):2, July 2016.

[7] Thelma Allen. Differential Privacy: Future Work & Open Challenges, January 2022. Last Modified: 2022-01-24T12:00-05:00.

[8] Borja Balle and Yu-Xiang Wang. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising, 2018.

[9] Tridib Bandyopadhyay, Vijay S. Mookerjee, and Ram C. Rao. Why it managers don't go for cyber-insurance products. *Commun. ACM*, 52(11):68–73, nov 2009.

[10] Raghav Bhaskar, Abhishek Bhowmick, Vipul Goyal, Srivatsan Laxman, and Abhradeep Thakurta. Noiseless database privacy. Cryptology ePrint Archive, Report 2011/487, 2011. https://ia.cr/2011/487.

[11] US Census Bureau. Census bureau sets key parameters to protect privacy in 2020 census results, Oct 2021.

[12] Ronald Cramer, Ivan Bjerre Damgaard, and Jesper Buus Nielsen. *Secure multiparty computation and secret sharing*. Cambridge University Press, Cambridge, England, July 2015.

[13] Leo de Castro, Andrew W. Lo, Taylor Reynolds, Fransisca Susan, Vinod Vaikuntanathan, Daniel Weitzner, and Nicolas Zhang. Scram: A platform for securely measuring cyber risk. *Harvard Data Science Review*, 2(3), 9 2020. https://hdsr.mitpress.mit.edu/pub/gylaxji4.

[14] Cynthia Dwork. Differential Privacy: A Survey of Results. In *Theory and Applications of Models of Computation—TAMC*, volume 4978 of *Lecture Notes in Computer Science*, pages 1–19. Springer Verlag, April 2008. Edition: Theory and Applications of Models of Computation—TAMC.

[15] Cynthia Dwork and Aaron Roth. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3-4):211–407, 2013.

[16] Simson L. Garfinkel, John M. Abowd, and Sarah Powazek. Issues encountered deploying differential privacy. In *Proceedings of the 2018 Workshop on Privacy in the Electronic Society*, WPES'18, page 133–137, New York, NY, USA, 2018. Association for Computing Machinery.

[17] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, STOC '87, page 218–229, New York, NY, USA, 1987. Association for Computing Machinery.

[18] Kyra H. Grantz, Hannah R. Meredith, Derek A. T. Cummings, C. Jessica E. Metcalf, Bryan T. Grenfell, John R. Giles, Shruti Mehta, Sunil Solomon, Alain Labrique, Nishant Kishore, Caroline O. Buckee, and Amy Wesolowski. The use of mobile phone data to inform analysis of COVID-19 pandemic epidemiology. *Nature Communications*, 11(1):4961, September 2020.

[19] Andrei Lapets, Kinan Dak Albab, Rawane Issa, Lucy Qin, Mayank Varia, Azer Bestavros, and Frederick Jansen. Role-based ecosystem for the design, development, and deployment of secure multi-party data analytics applications. In *2019 IEEE Cybersecurity Development (SecDev)*, pages 129–140, 2019.

[20] Eric Pence and Leo de Castro. scram_privacy, May 2022.

[21] Michael K. Reiter and Aviel D. Rubin. Crowds: Anonymity for web transactions. *ACM Trans. Inf. Syst. Secur.*, 1(1):66–92, nov 1998.

[22] Alexandra Wood, Micah Altman, Aaron Bembenek, Mark Bun, Marco Gaboardi, James Honaker, Kobbi Nissim, David R. OBrien, Thomas Steinke, and Salil Vadhan. Differential privacy: A primer for a non-technical audience. *Vanderbilt Journal of Entertainment & Technology Law*, 21(1):209–275, 2018.

[23] Andrew Chi-Chih Yao. How to generate and exchange secrets. In *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*, pages 162–167, 1986.