Differentially Private Resource Allocation

Joann Qiongna Chen
University of California, Irvine

Tianhao Wang
University of Virginia

Zhikun Zhang
CISPA Helmholtz Center for Information Security, Stanford University

Yang Zhang
CISPA Helmholtz Center for Information Security

Somesh Jha
University of Wisconsin

Zhou Li
University of California, Irvine

ABSTRACT

Recent studies have shown that systems with limited resources like Metadata-private Messenger (MPM) suffer from side-channel attacks under *resource allocation (RA)*. In the case of MPM, which is designed to keep the identities and activities of both callers and callees private from network adversaries, an attacker can compromise a victim's friends and keep calling the victim to infer whether the victim is busy, which breaks the privacy guarantee of MPM.

In this work, we systematically study how to protect the privacy of RA against the aforementioned attacks with differential privacy (DP). Though DP has been tested by Angel et al. (IEEE S&P 2020) in protecting RA, which lets the allocator add dummy requests following a biased Laplace distribution to hide the existence of the victim and then assign resources randomly, we identify that this approach does not leverage the uncertainty from the attacker's view, thus leading to a loose bound of DP. As a result, *more than 40%* of the resources are wasted to satisfy DP. To make the DP solutions more practical, we precisely model the RA process from the attacker's view and present a thorough study of the noisy allocation mechanisms by considering different distributions, scales, and biases of noise. We identify four new mechanisms and prove that they all follow ϵ-DP (Angel et al. follow (ϵ, δ)-DP). Through theoretical and empirical analysis, we found these approaches can outperform Angel et al. by a large margin in privacy-utility tradeoff.

CCS CONCEPTS

• **Security and privacy → Privacy-preserving protocols**.

KEYWORDS

Differential Privacy; Resource Allocation; Privacy Amplification

ACM Reference Format:
Joann Qiongna Chen, Tianhao Wang, Zhikun Zhang, Yang Zhang, Somesh Jha, and Zhou Li. 2023. Differentially Private Resource Allocation. In *Annual Computer Security Applications Conference (ACSAC '23), December 04–08, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3627106.3627181

ACSAC '23, December 04–08, 2023, Austin, TX, USA

ACM ISBN 979-8-4007-0886-2/23/12.
https://doi.org/10.1145/3627106.3627181

1 INTRODUCTION

Resource allocation (RA) is a long-standing problem relevant to a variety of application scenarios, such as virtual machine assignment [46], storage allocation [43], network bandwidth management [44], and channel allocation [61]. Prior works mostly focus on the efficiency and cost of RA [9, 27, 30, 31, 33, 39, 49], e.g., how to improve the resource utilization and guarantee the quality of service to all users [30]. However, the privacy issues of RA have been overlooked for a long time and were only studied recently. Angel et al. [4] reveal that a powerful attacker can determine the existence of other parties in the RA system. Concretely, for an allocator managing limited resources, when one party requests resources, the number of resources the other parties can obtain will be affected. Therefore, the attacker can try to send a large volume of requests and use the allocation results to infer the existence of other users. Knowing the existence of others opens the door to more serious attacks that can infer users' activities. For example, although Metadata-private messengers (MPM) are designed to hide the calling activities between clients, such privacy guarantee can be breached with RA side-channel and traffic analysis [4].

Existing Resource Allocators. Most of the existing allocators (e.g., the first-in-first-out allocator) do not offer any privacy guarantee [3]. Recently, Angel et al. [3] proposed an allocator AKR[1] that satisfies differential privacy (DP) [17]. Angel et al. consider the scenario where the resource allocator owns a limited number of resources and the attacker controls a large number of clients. The attacker learns of the existence of another victim when the requests to the allocator are not fulfilled. To protect privacy during RA, AKR adds dummy requests to the real ones and then assigns resources to randomly chosen requests. The number of dummy requests follows the *biased Laplace distribution*, and by a standard *post-processing argument* in DP (explained in Section 2.3), the existence of the victim is differentially private to the attacker. While the dummy requests puzzle the attacker, we found that the utility of AKR is not satisfactory. For instance, to achieve an acceptable protection level of DP (with parameters $\epsilon = 2, \delta = 10^{-6}$) *more than 40% of the resources must be wasted* in its experiment setting.

Our Solution. Different from AKR, which implies the attacker knows the total number of requests after noise is added, we observe that the practical attacker only has a *partial* view of RA. Therefore we choose to model the RA privacy from the attacker's view. Due to the randomness introduced by RA, we benefit from *"privacy*

[1]The first letter of the authors' names.

amplification" [5, 20] through such modeling and achieve better privacy-utility tradeoff.

Then, we implemented the DP mechanisms under four noise distributions, including constant (CST), uniform (UNI), one-sided geometric (GEO), and double geometric (DGEO), and tailored them to our new modeling. We conduct a rigorous privacy analysis and derive *much tighter privacy bounds* than AKR. We prove GEO and DGEO always satisfy ϵ-DP under various parameters, while CST and UNI satisfy ϵ-DP under certain conditions. Interestingly, we find that adding a constant noise (CST), which obviously violates traditional DP, can be proven to satisfy DP in the context of RA, due to the randomness of the allocation process. On the other hand, AKR only considers *non-negative* Laplace noise and relies on the post-processing argument to satisfy (ϵ, δ)-DP.

Evaluation. We evaluate the proposed mechanisms empirically by simulating the RA process of Alpenhorn [42] with *5 million to 100 million* rounds of requests, to demonstrate the privacy-utility tradeoff in real-world settings. (1) GEO outperforms other mechanisms when ϵ is smaller (i.e., $\epsilon < 2$) and has relatively stable performance; (2) DGEO performs better with a larger ϵ ($\epsilon > 2$). Compared to AKR which wastes 44% of the resources, DGEO only wastes 10% of resources with $\epsilon = 2$. Moreover, when $\epsilon = 2.25$, AKR utilizes 60% of the resources while DGEO achieves 97% utilization. (3) Parameters of the mechanisms have to be carefully tuned and negative bias should be avoided. The advantage over AKR is especially surprising as AKR is supposed to have better utility under the relaxed (ϵ, δ)-DP, whereas our mechanisms follow the strict ϵ-DP. This justifies the effectiveness of our privacy analysis.

Contributions. The main contributions are summarized below:

- We conduct a rigorous privacy analysis of differentially private RA, and derive tighter privacy bounds under the attacker's view for four noisy mechanisms.
- We theoretically and empirically evaluate our proposed mechanisms. One mechanism, called GEO, leads to the best privacy-utility tradeoff and outperforms AKR by a large margin.
- We published the code in a GitHub repository [14].

2 BACKGROUND

2.1 Problem Definition

Resource allocation (RA) assigns limited resources to the requesting parties, and we focus on RA within computing systems in this paper. Examples include resource management in data centers [2], assignment of virtual machines (VMs) in cloud [46], cache allocation in computers [43], and channel allocation for Metadata-private Messenger (MPM) [42]. Below we first provide an abstract view of standard RA and describe its involved parties and procedure. Then, we describe the attackers' goals and capabilities in RA. The frequently used notations are defined in Table 1.

RA Parties and Procedure. Our abstraction of standard RA considers a scenario where an *allocator* allocates *resources* based on the *requests* submitted by a number of *clients*. The allocator can contain one server or a group of servers for fault-tolerance. In the setting of data center, the allocator can be a virtual machine manager (VMM), and the client can be a data center tenant. In the setting of MPM, where two users can set up a call in a private way, the allocator can be a callee and the client can be a caller.

Notation	Description
D, D'	Neighboring datasets differing in one victim
k	Number of available resources
m	Number of compromised clients
d	Number of noisy requests (can be negative)
y	Number of resources dispatched to attacker
x_ℓ, x_r, p, s, μ	Parameters of the noisy mechanisms

Table 1: Notations frequently used in this paper.

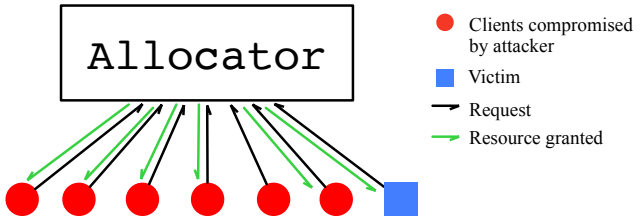


Figure 1: An example of RA. An *allocator* has six resources and the total number of requests sent by attacker is six. Privacy of the victim is violated when the attacker observes one of the requests is not fulfilled.

Regarding the RA procedure, we assume it takes rounds of interactions between the allocator and the clients. In each round, the allocator receives requests from its clients for resources (e.g., CPUs in a cloud and communication channels to be allocated to a caller in MPM) and makes the best efforts to serve the requests. Hence, for each request, the allocator either accepts it and allocates the resources, or rejects it when all resources have been occupied.

Following prior work [3], we assume the quantity of the resources is a limited number k, and all resources are identical. Each round, some clients send requests, and each request asks for one piece of resource. Because the resources are identical, the requests are also identical (except the requesters' IDs). We note that some assumptions can be relaxed (e.g., resources are not identical and each client can request multiple resources) to match different application scenarios, and we discuss these variations in Section 6.2.

Adversary Model. Since the clients' requests might not always be fulfilled under limited resources, the allocator's response could leak information about the existence of some clients. Figure 1 illustrates how such inference attack can be conducted. Formally, we assume the attacker in the strongest attack scenario who can:

- *compromise all clients except one victim client*, and we denote the number of compromised clients as m.
- know the number of available resources k before RA.
- compromise more clients than the resources, i.e., $m \geq k$, and all requests are submitted at the same time.

The attacker can tell there is a victim requesting a resource if less than k requests from the attacker are fulfilled.

We assume the adversary is malicious who can behave arbitrarily rather than being semi-honest. We only consider the privacy issues in RA and other issues like availability (e.g., the attacker blocks a victim from getting resources by overwhelming the allocator) are out of scope. We note that an adaptive attacker can exploit the correlation of results between multiple rounds, and infer more

information that weakens the allocation privacy. We propose a few approaches to tackle such adversary in Section 6.1.

Regarding the allocator, we assume it is trustworthy, and can see all clients and requests and add noises. Hence, the allocator can analyze the historical data to estimate the parameters to be used by our mechanisms without privacy issues. We also assume the communication between the victim and the allocator is secure, so the number of the victim's requests are not leaked.

Impact of RA Side-channel Leakage. Even though the information about the victim during RA is seemingly insignificant, it can be leveraged as a side channel to break privacy-enhancing technologies or make the subsequent attacks more effective.

Specifically, Angel et al. described an attack based on the RA side-channel [4] against MPM. MPM like Vuvuzela [61], Alpenhorn [42], Stadium [60] and Karaoke [41] hide both the message content and its metadata (including sender, receiver, time of communication, etc.) from the network adversaries. In essence, a user within an MPM initiates a conversation with her friend on an agreed time or round and encrypts the messages with a shared key. In the conversation round, the user initiates k channels to k friends (including the friend to have the "real" conversation). To avoid leaking metadata, users are forced to send and receive a message on each channel in each round [2]. Since MPM requires the clients to always be online, only the communicating parties of a client should be protected, while the client's existence is known.

It turns out the privacy guarantee of MPM can be entirely violated. As shown by Angel et al. [4], a user usually has a greater number of friends than k channels. When the attacker controls m ($m \geq k$) friends of the user and lets them call the user, if the user is busy (e.g., not responding) to more than $m - k$ callers controlled by the attacker, the attacker knows the user is communicating with others who are out of her control. Moreover, when the attacker compromises the friends of multiple users, she can infer which users are likely active in a given round with intersection and disclosure attacks [1, 45]. Specifically, the attacker can narrow down the possible sender-recipient pairs by ignoring all the idle users during the first round of calling. Then the attacker can build intersections of active users and keep reducing the set of possible sender-recipient pairs during additional rounds. Because the requests and resources are all identical under our assumptions, detecting such inference attack is also very challenging.

Existing Resource Allocators. We aim to design an RA that *hides the existence of the victim* while *maximizing request fulfillment*. One trivial solution that provides perfect privacy is to have the allocator withhold all the resources and reject every request, but obviously, this solution has zero utility. Angel et al. characterizes the existing allocators into (1) FIFO (first in, first out) allocator, (2) Uniform allocator, (3) Slot-based resource allocator (SRA) and (4) Randomized resource allocator (RRA) [3], while FIFO and uniform allocators are non-private and SRA and RRA are private. However, both SRA and RRA incur prominent utility loss.

[2] MPM is different from the normal messenger apps in that it can decline legitimate calls to provide metadata privacy. Yet, given that each conversation round has very small latency (e.g., measured in micro-seconds in the context of Alpenhorn [3]), the impact of call declining on user experience remains moderate.

2.2 A Primer on Differential Privacy

Our work applies differential privacy (DP) mechanisms on RA. We briefly overview DP in this subsection and describe how AKR applies DP to RA [3] in the next subsection.

In the standard (central) setting, a trusted data curator adds noise (e.g., through the Laplace mechanism or Geometric mechanism) to fulfill a DP notion (e.g., (ϵ, δ)-DP) given a query from a data consumer, which bounds the information leakage provably.

Definition 1 ((ϵ, δ)-Differential Privacy). *[17] An algorithm $\mathcal{M}$ satisfies (ϵ, δ)-differential privacy against an adversary, where $\epsilon, \delta \geq 0$, iff for any two neighboring datasets D and D', and any subset Y of all possible outcomes of algorithm $\mathcal{M}$, we have*

$$\Pr\left[\mathcal{M}(D) \in Y\right] \leq e^{\epsilon} \Pr\left[\mathcal{M}(D') \in Y\right] + \delta \tag{1}$$

We consider two datasets D and D' to be neighbors, denoted as $D \simeq D'$ if and only if $D = D^{'} + u$ or $D^{'} = D + u$, where $D + u$ denotes the dataset resulted from adding one user's data u to the dataset D. ϵ measures privacy loss at a differential change in data, which is also called privacy budget. δ models the probability when the algorithm $\mathcal{M}$ fails to be differentially private, which is also called "failure probability". The value of δ is normally very small in order to keep the algorithm satisfying DP most of the time. When $\delta = 0$, we simplify the $(\epsilon, 0)$-DP to ϵ-DP and call it pure DP.

Laplace Mechanism [17]. It computes a function f on input dataset D while satisfying ϵ-DP, by adding to $f(D)$ a random noise. The magnitude of the noise depends on GS_f, i.e., the *global* L_1 *sensitivity* of f, defined as (on any two neighboring datasets $D \simeq D'$),

$$\mathrm{GS}_f = \max_{D \simeq D'} ||f(D) - f(D')||_1 \tag{2}$$

When f outputs a single element, $\mathcal{M}$ can be written as:

$$\mathcal{M}(D) \quad = f(D) + \mathsf{L}\left(\frac{\mathrm{GS}_f}{\epsilon}\right) \tag{3}$$

where $\mathsf{L}(s)$ denotes a random variable sampled from the Laplace distribution with scale parameter s such that $\Pr[\mathsf{L}(s) = x] = \frac{1}{2s} e^{-|x|/s}$. When f outputs a vector, $\mathcal{M}$ adds independent samples of $\mathsf{L}\left(\frac{\mathrm{GS}_f}{\epsilon}\right)$ to each element of the vector.

Geometric Mechanism [40]. If the output domain is discrete, one can use this mechanism, which draws noise from the double geometric distribution: $\Pr[\mathrm{DG}(s) = x] = \frac{1-e^{-\frac{1}{s}}}{1+e^{-\frac{1}{s}}} e^{-\frac{1}{s}|x|/\mathrm{GS}_f}$, for $x \in \mathbb{Z}$. The Geometric mechanism satisfies ϵ-DP.

Composition. Two properties, i.e., *composition* and *post-processing*, of DP, are frequently used to build complicated algorithms from the basic mechanisms. Sequential composition states that combining multiple subroutines that satisfy DP for $(\epsilon_1, \delta_1), (\epsilon_2, \delta_2), \cdots$ results in a mechanism that satisfies (ϵ, δ)-DP for $\epsilon = \sum \epsilon_i$ and $\delta = \sum \delta_i$. Advanced composition, e.g., Rényi DP [48], provides smaller privacy degradation (ϵ grows sub-linearly). The post-processing property states that, any operation (post-process) of an (ϵ, δ)-DP algorithm's result still satisfies (ϵ, δ)-DP.

Definition 2 (Rényi Differential Privacy [48]). *A mechanism $\mathcal{M} : \mathcal{X} \to \mathcal{Y}$ is said to satisfy (ν, τ)-RDP if the following holds for*

any two neighboring datasets D, D'

$$\frac{1}{\nu-1}\log \mathbb{E}_{o\sim\mathcal{M}(D)}\left[\left(\frac{\Pr\left[\mathcal{M}(D)=o\right]}{\Pr\left[\mathcal{M}(D')=o\right]}\right)^{\nu}\right] \leq \tau.$$

Theorem 1. *[RDP Sequential Composition [48]] If $\mathcal{M}_1$ and $\mathcal{M}_2$ are (ν,τ_1)-RDP and (ν,τ_2)-RDP respectively then the mechanism combining the two $g(\mathcal{M}_1(D),\mathcal{M}_2(D))$ is $(\nu,\tau_1+\tau_2)$-RDP.*

Theorem 2. *[RDP to (ϵ,δ)-DP [48]] If a mechanism is (ν,τ)-RDP, then it also satisfies $(\tau+\frac{\log 1/\delta}{\nu-1},\delta)$-DP.*

2.3 Differentially Private Allocation in AKR

As all the requests are identical from the allocator's point of view, the key to providing privacy is to "control" the number of resources the attacker receives. Thus, AKR asks the allocator to add dummy requests. Specifically, AKR sets the dataset D to be all requests made by clients, and computes the noise $\mathsf{L}\left(\frac{\mathrm{GS}_f}{\epsilon}\right)$. To ensure the number of added requests (i.e., $\mathcal{M}(D)$ in Equation 3) is non-negative, a bias μ is added when sampling the Laplace noise so that the probability of the noise being negative is bounded by δ, which we refer to as the *biased Laplace distribution*. The workflow of AKR is:

- **Input**: k, μ, GS_f, ϵ, D
- Noise $d \leftarrow \left\lceil \max\left(0, \mu + \mathsf{L}\left(\frac{\mathrm{GS}_f}{\epsilon}\right)\right)\right\rceil$
- Set $Q \leftarrow |D| + d$ dummy requests
- $U \leftarrow$ uniformly select $\min(|Q|, k)$ items out of Q
- **Output**: U

Overall, AKR satisfies (ϵ,δ)-DP. Below is its DP proof.

Theorem 3 (DP Proof for AKR [3]). *Algorithm $\mathcal{M}$ is (ϵ,δ)-differentially private for $\epsilon = 1/s$ and $\delta = \int_{-\infty}^{1} \mathsf{L}(w|\mu,1/\epsilon)\,dw$. Specifically, for any subset of values L in the range $[f(D),\infty)$ of $\mathcal{M}$:*

$$\Pr\left[\mathcal{M}(D)\in L\right] \leq e^{\epsilon}\Pr\left[\mathcal{M}\left(D'\right)\in L\right]+\delta$$

and

$$\Pr\left[\mathcal{M}\left(D'\right)\in L\right] \leq e^{\epsilon}\Pr\left[\mathcal{M}(D)\in L\right]$$

where $f(S)$ computes the cardinality of set S.

Note that:

$$\delta = \int_{-\infty}^{1} \mathsf{L}(w|\mu,1/\epsilon)\,dw = \begin{cases} \frac{1}{2}e^{\epsilon(1-\mu)} & \text{if } \mu > 1 \\ 1-\frac{1}{2}e^{\epsilon(1-\mu)} & \text{if } \mu \leq 1 \end{cases}$$

We can see μ tends to be large in order to have a small δ.

Given that the noise is non-negative, what the attacker observes after allocation can be seen as a post-processing of the requests, as the victim's request is indistinguishable from the added dummy requests. Specifically, let Y be a random variable denoting the number of resource attacker gets. Since the attacker only learns which requests of her were fulfilled, from her point of view dummy requests and victim are indistinguishable. Thus for each value $l \in [0,k]$, $\Pr[Y=l|\mathcal{M}(D)=t] = \Pr[Y=l|\mathcal{M}(D')=t]$, where t is the number of requests with dummies. Combined with the inequalities governing the probabilities that $\mathcal{M}$ outputs each value of t for D and D', respectively. We have that $\Pr[Y=l|D] \leq e^{\epsilon}\Pr[Y=l|D'] + \delta$, and similarly with D and D' exchanged. Thus the distribution of the number of attacker's requests allocated are very close for D and D'.

3 MODELING RESOURCE ALLOCATION

In this section, we first demonstrate the problem of AKR's modeling of RA. Then, we present a taxonomy of different ways to "add noise" in RA and a general approach to model privacy.

3.1 Privacy Amplification from Allocation

We argue that AKR's modeling of RA leads to suboptimal utility due to the lack of consideration for the attacker's view and capabilities. Though AKR, by its definition, does not reveal the number of total requests each round, their proof indicates a stronger statement that the DP guarantee holds even when the attacker observes the *total* number of requests after noise is added (i.e., the number of requests from both the attacker and the victim). More specifically, their proof guarantees that the noisy total number of requests is bounded by (ϵ,δ)-DP when honest clients are added. However, such information is not actually accessible to the attacker, thus it creates a gap between the proof and the actual definition of the RA problem. Examining the attacker's view is crucial for privacy amplification in our study. By comprehending the capabilities and limitations of the attackers, we can construct a precise analysis and avoid unnecessary noise. In real-world scenarios, the capability of an attacker can be considerably limited, as they are typically not granted access to the internal states of an allocator. In fact, if the attacker can observe the internal states of an allocator, she just needs to access the number of requests *before* adding noise, which defeats all DP-based protection.

We note that such a modeling gap is common in DP for ease of proof. For example, in DP-SGD [22], the privacy guarantee is proved on each SGD step, implying that the attacker can observe the intermediate steps, but such information should not be accessible to the attacker. A similar case also appears in the proof of privacy blanket [6, Theorem 3.1] (which assumes the attacker has unrealistic extra information for the ease of proof) for the shuffle DP model.

Hence, we propose to more precisely model the attacker's capabilities and offer a tighter bound under the notion of DP. By conducting the privacy analysis from scratch, we present a set of "*privacy amplification*" results[3]. In this paper, the privacy amplification stems from the fact that the attacker only has a *partial* view of the allocation result. The attacker is aware of whether the other compromised clients receive the allocated resources, except for the one uncompromised client. Compared to AKR, which has to introduce larger noise to deter the (unrealistic) attacker, we can use smaller noise to satisfy DP. In Section 5.2 ("Why Models Attacker's View"), we elaborate the impact of privacy amplification.

3.2 Design Space

As described in Section 2.1, RA takes two steps: (1) receive a request, and (2) allocate the resource if the request is accepted. Hence, for privacy protection, the allocator can add noise to either (1) the number of requests (i.e., by adding dummy requests or removing some requests), or (2) the number of available resources (i.e., by withholding some available resource). After that, the allocator can randomly select requests and assign resources to them. Therefore, the design space for the allocator is composed of:

[3]Privacy amplification refers to the effect where we can prove the privacy cost is reduced after some operations (e.g., subsampling [5] and shuffling [20]).

- **DS1: Choosing Where to Add Noise.** The allocator can add noise to either the number of requests or the number of resources or both. Our analysis shows that randomizing the number of resources has the same effect as randomizing the number of requests (explained later), thus we focus on designing methods to add noise to the number of requests. In Section 6.4, we give a few real-world examples.
- **DS2: Choosing How Noise is Generated**. The allocator adds noise to the observed number of requests, and we have the flexibility to choose:
  - The distribution of the noise.
  - The range (support) of the distribution.

We found AKR only covered part of the design space: (1) AKR considered RA as post-processing and only adds *non-negative* noise (dummy requests) to the requests. (2) AKR did not consider distributions other than the Laplace distribution.

Adding Noise to Resource. Beyond adding noise to the requests, we can choose to add noise to the resources. Here we consider that the noise is always negative, or the resources are withheld from being assigned to clients. The positive noise can be seen as "creating" resources on the fly and assigning more than what is asked by a client, which could be impractical for a real-world system. Yet, we can prove that withholding any number of resources can be equivalently modeled as assigning them to dummy requests. Specifically, the allocator could withhold n resources from k requests, which results in $k-n$ random requests getting resources. This is equivalent to that n requests being randomly removed from the system (so that the rest $k-n$ requests are granted with resources). Thus, we only consider adding noise to requests.

3.3 Privacy Modeling

Under DS1, we model RA's privacy through the lens of DP as follows. We use d to denote the random variable for the number of noisy requests. D denotes the number of requests made in a round. Given two neighboring datasets D, D', w.l.o.g., we assume D' equals to D *plus* the honest request from the victim client[4]. RA's privacy can be quantified as:

$$\frac{\Pr\left[\mathsf{View}^{\mathcal{A}}_{\mathcal{M}}(D)=y\right]}{\Pr\left[\mathsf{View}^{\mathcal{A}}_{\mathcal{M}}(D')=y\right]}=\frac{\sum_{i=x_\ell}^{x_r}\Pr\left[d=i\right]\Pr\left[y\mid |D|+d\right]}{\sum_{i=x_\ell}^{x_r}\Pr\left[d=i\right]\Pr\left[y\mid |D'|+d\right]} \tag{4}$$

where $\mathsf{View}^{\mathcal{A}}_{\mathcal{M}}(\cdot)$ models the allocation outcomes in the attacker's view. Note that $\mathsf{View}^{\mathcal{A}}_{\mathcal{M}}$ differs from $\mathcal{M}$ in Equation 1 in that $\mathsf{View}^{\mathcal{A}}_{\mathcal{M}}$ is a partial view of the final allocation outcome. $\Pr[d=i]$ denotes the probability $d=i$, where d is a random variable and i is within some range $[x_\ell, x_r]$, and $\Pr[y\mid |D|+d]$ is the probability that attacker gets y resources. This equation measures the difference in the attacker's observation that is impacted by the one honest request. If $d\geq 0$, the allocator adds some dummy requests; $d<0$ models removing some requests (e.g., ignoring requests). Notice that Equation 4 follows ϵ-DP, which is different from AKR that follows (ϵ, δ)-DP.

With $\Pr[y\mid |D|+d]$, we are able to more precisely model RA privacy than AKR and captures the randomness introduce by RA, since y represents only the output in the attacker's view (i.e., $y\leq |D|$). We now describe the detailed analysis of $\Pr[y\mid |D|+d]$ under two cases: $d\geq 0$ and $d<0$. We enumerate all possible situations under RA and derive the exact probability expressions for $\Pr[y\mid |D|+d]$ and $\Pr[y\mid |D'|+d]$(i.e., Equation 5 and Equation 6).

Request Addition ($d\geq 0$). For the case of D, assuming there are m requests from D, given a specific number of dummy requests $d\geq 0$, we have:

$\forall (k-d)_+ \leq y\leq \min(k,m)$,

$$\Pr\left[y\mid |D|+d\right]=\frac{\binom{m}{y}\binom{d}{k-y}}{\binom{m+d}{k}} \tag{5}$$

$\Pr[y\mid |D|+d]=0$ if y is outside of the above range. y has to satisfy $y\leq \min(k,m)$ because what the attacker observes cannot exceed the total number of resources k or the number of requests m. Similarly, $y\geq (k-d)_+$ (we use x_+ to denote $\max(0,x)$) because there are only d other requests, so the attacker must get at least $(k-d)_+$ resources.

We only model the case when the number of requests $m\geq k-d$ because when $m<k-d$, all requests are fulfilled (no privacy leakage). In that case, $\Pr[y\mid |D|+d]=1$ for $y=m$ and $\Pr[y\mid |D|+d]=0$ otherwise.

The denominator of Equation 5 is $\binom{m+d}{k}$ because we have a total of $m+d$ requests and we allocate k resources to them (equivalent to choosing k from $m+d$ requests to allocate resources). Thus there are $\binom{m+d}{k}$ possible assignments. The numerator is $\binom{m}{y}\binom{d}{k-y}$ because, for the fixed set of m requests controlled by the attacker, y of them are fulfilled; there are $\binom{m}{y}$ possible assignments. Similarly, for the rest d requests, there are $\binom{d}{k-y}$ possible assignments. So all together there are $\binom{m}{y}\binom{d}{k-y}$ possible assignments that satisfy the constraint that y resources go to m processes.

For the case of D', which has an additional honest request, the attacker could receive one fewer resource. Thus we have:

$\forall (k-d-1)_+ \leq y\leq \min(k,m)$,

$$\Pr\left[y\mid \left|D'\right|+d\right]=\frac{\binom{m}{y}\binom{d+1}{k-y}}{\binom{m+d+1}{k}} \tag{6}$$

Similar to Equation 5, in Equation 6, when $m<k-d-1$, $\Pr[y\mid |D'|+d]=1$ for $y=m$ and $\Pr[y\mid |D'|+d]=0$ otherwise.

Request Removal ($d<0$). For the case of D (the honest request does not exist), when the number of added dummy requests is negative ($d<0$), some requests will be removed randomly. We have:

$$\Pr\left[y\mid |D|+d\right]=\begin{cases}1 & \text{if } y=\min(m+d,k)_+\\ 0 & \text{otherwise}\end{cases} \tag{7}$$

This case is simpler than "Request Addition", and what the attacker observes is deterministic: if after adding negative noise d, $m+d$ is still greater than k, then the attacker will always receive k resources; if $m+d\leq k$, then the attacker will always receive $m+d$ resources.

[4]For the other neighboring case (D' equals to D *minus* the honest request), the modeling and proofs are similar, so it is omitted in this version due to page limit.

For the case of D', there are $m+1+d$ requests, and we need to consider whether the honest request is fulfilled. Let $x = \min(m+1+d, k)_+$, which leads to two scenarios:

- Allocator assigns resources to the honest client: in this case, y can only be $x-1$. The probability of the allocator assigning resources to the honest client is $\frac{x}{m+1}$, which is equivalent to the case of selecting $x = \min(m+1+d, k)_+$ items from a total of $m+1$ items without replacement and that the honest client is selected.
- Allocator does not assign resources to the honest client: y must be x if the honest request is not fulfilled, which happens with probability $1 - \frac{x}{m+1}$.

Thus we have:

$$\Pr\left[y \mid \left|D'\right| + d\right] = \begin{cases} 1 - \frac{x}{m+1} & \text{if } y = x \\ \frac{x}{m+1} & \text{if } y = x-1 \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

where $x = \min(m+1+d, k)_+$.

We want to highlight that considering request removal (negative noise) is another key difference from AKR.

Attacker's Strategy. From the attacker's point of view, it is important to set m (the number of compromised clients) to a value that can maximize privacy leakage (i.e., maximize Equation 4). Recall that we assume k (resource capacity) is known to the attacker, and each client can submit at most one request (see Section 2.1). Following the previous analysis of request addition and request removal, we can derive the best attacker strategy below we follow this strategy for this rest of the paper.

Theorem 4. *The maximum privacy leakage happens when the attacker sends $m = k$ requests.*

Proof. We consider the cases of noise $d < 0$ and $d \geq 0$, and prove $m = k$ causes maximum privacy leakage in both cases.

First, considering the case when noise is non-negative ($d \geq 0$), the attacker's goal is to choose m to maximize the difference between the cases of D and D'. Note that the difference can only be observed when $m + d \geq k$ because otherwise, all requests will be granted with resources. To ensure $m + d \geq k$ for all $d \geq 0$, we have $m \geq k$. Based on the previous analysis, when $0 \leq d < k$, there is no privacy at $y = k - d - 1$, because

$$\Pr[y \mid |D| + d] = 0, \Pr\left[y \mid \left|D'\right| + d\right] = \frac{m!k!}{(k-d-1)!\,(m+d+1)!}$$

Thus it does not matter to the attacker what value to set to m in this case. For $d \geq k$, the privacy protection is given by

$$\frac{\Pr[y \mid |D| + d]}{\Pr[y \mid |D'| + d]} = \frac{\binom{m}{y}\binom{d}{k-y} / \binom{d+m}{k}}{\binom{m}{y}\binom{d+1}{k-y} / \binom{d+m+1}{k}} \leq 1 + \frac{k}{m+d+1-k}$$

In order to maximize the above, we need to set m to its minimum within the range of $m \geq k$, that is, $m = k$.

Now, we consider the case when negative noise ($d < 0$) is added. By observing Equation 7 and Equation 8, we know that to trigger the different outputs for case D and D' (i.e., $y = m + d$ for case D and $y = m + d + 1$ for case D'), $m + d$ needs to be $< k$. The difference of D and D' (privacy protection) is then given by

$$\frac{\Pr[y \mid |D| + d]}{\Pr[y \mid |D'| + d]} = \frac{1}{1 - \frac{m+1+d}{m+1}} = \frac{m+1}{-d} \tag{9}$$

To have $m + d < k$ (i.e., $d < k - m$) hold for all $d < 0$, we have $m \leq k$. Now, in order to maximize Equation 9, m is to be set to k. □

4 NOISY MECHANISMS

In this section, we analyze different noisy mechanisms under DS2. As the RA output is discrete, we choose discrete distributions for the mechanisms. Specifically, we consider constant, uniform, one-sided geometric, and double geometric distributions, and name them CST, UNI, GEO, DGEO for short. Though these mechanisms have been studied in the standard DP [24, 25], we conducted new theoretical analysis to derive tighter privacy bounds, which require extensive proof work as shown in Appendix A. In Table 2, a summary of different mechanisms is given. In particular, 1) we prove the DP bounds for all mechanisms, though CST and UNI only satisfy DP when certain conditions are met (i.e., noise sample space should be at least k); 2) our mechanisms outperform AKR in utility by a large margin[5].

4.1 Constant Noise (CST)

In this case, we consider request addition only, and the noise d always equals a constant number c. Observing Equations (5) and (6), the valid y support sets differ in one case where $y = k - d - 1$. But as long as $d \geq k$, both $\Pr[y \mid |D| + d]$ and $\Pr[y \mid |D'| + d]$ have the same valid set of $y \in \{0, 1, \cdots, \min(m, k)\}$, and the privacy can be quantified as:

$$\frac{\Pr[y \mid |D| + d]}{\Pr[y \mid |D'| + d]} = \frac{\frac{\binom{m}{y}\binom{d}{k-y}}{\binom{d+m}{k}}}{\frac{\binom{m}{y}\binom{d+1}{k-y}}{\binom{d+m+1}{k}}} = \frac{(m+d+1)(d+y+1-k)}{(m+d+1-k)(d+1)} \tag{10}$$

As a result, we have the following theorem:

Theorem 5. *Assuming an allocator has k resources, constant noise has to be at least k to satisfy DP.*

Proof. Suppose the resource allocated to the attacker is y, and the attacker always sends out $m = k$ requests. Then we have

$$\begin{cases} y \in \{(k-d)_+, (k-d)_+ + 1, \cdots, k\} & \text{when } D \\ y \in \{(k-d-1)_+, (k-d-1)_+ + 1, \cdots, k\} & \text{when } D' \end{cases} \tag{11}$$

Note that $y = (k-d)_+$ happens in D when all dummy requests get resources and the remaining resources go to the attacker. Similarly, $y = (k-d-1)_+$ happens in D' when the victim and all dummy requests get resources and the remaining resources go to the attacker. When $d \geq k$, $y \in \{0, 1, \cdots, k\}$ for both D and D'. Thus, given $m = k$ and Equation 10, we can give an upper-bound of its privacy leakage as:

$$e^{\epsilon} = \frac{\Pr[y \mid |D| + d]}{\Pr[y \mid |D'| + d]} = \frac{(m+d+1)(d+y+1-k)}{(m+d+1-k)(d+1)} \leq \frac{k+d+1}{d+1} \tag{12}$$

where $e^{\epsilon} = \frac{k+d+1}{d+1}$ is reached at $y = k$. □

This is a surprising result, as adding a fixed noise should not satisfy DP. In our case, adding a fixed noise still provides privacy because of the randomness of the allocation process. Still, we argue that it does not offer good utility. Due to the constraint $c \geq k$, the utility is never more than 0.5.

[5]The numbers come from simulation of Section 5.2. The theoretical analysis of utility has also been done, but omitted due to page limit.

	Privacy	Noise	Noise Sign	DP Condition	Utility (ϵ=0.65)	Utility (ϵ=1.7)	Utility (ϵ=2.3)
CST	ϵ-DP	Constant	+	Noise $c \geq k$	**0.50**	0.50	0.50
UNI	ϵ-DP	Discrete uniform	+/−	Right bound $x_r \geq k$	0.46	0.65	0.70
GEO	ϵ-DP	One-sided geometric	+/−	-	0.47	**0.82**	0.90
DGEO	ϵ-DP	Double geometric	+/−	-	0.44	0.77	**0.98**
AKR [3]	(ϵ, δ)-DP	Laplace	+	-	0.32	0.53	0.59

Table 2: A summary of different mechanisms and their utility under some representative ϵ values. Note that $k = 10$ and $\delta = 10^{-6}$.

4.2 Uniform Mechanism (UNI)

In this case, the discrete noise (it can be negative or non-negative) is drawn uniformly from $[x_\ell, x_r]$:

$$\Pr[d = i] = \frac{1}{x_r - x_\ell + 1},\ i = x_\ell, x_\ell + 1, \ldots, x_r \tag{13}$$

x_ℓ and x_r define the shape of distribution used in UNI, with x_ℓ defining the starting point. In Appendix A.1, we prove attacker's view satisfies DP when $x_r \geq k$ or $[-k-1, 0] \in [x_\ell, x_r]$.

Yet, our analysis shows UNI is also not recommended when the utility requirement is more critical. This is because the utility degrades linearly to negative noise when the number of requests equals the number of resources. In a nutshell, suppose the total number of requests is n and $n = k$. Removing requests causes less resource to be allocated with certainty while adding requests results in the same with a probability. With this, our goal is to have $x_\ell \geq 0$ and $x_r \geq k$ to achieve the best privacy and utility tradeoff, and Appendix B studies how these parameters should be determined.

4.3 One-sided Geometric Mechanism (GEO)

Intuitively, reducing the probability density of large noise can reduce the amount of dummy requests added, and thus improve utility. To this end, we adopt the geometric distribution within the range $[x_\ell, \infty)$ with the noise distribution:

$$\Pr[d = i] = p(1-p)^{i - x_\ell},\ i = x_\ell, x_\ell + 1, x_\ell + 2, \ldots \tag{14}$$

Like UNI, x_ℓ also models the starting point of the new distribution. For p, a larger value makes the noise decay faster and has negligible probability for large value i, thus improving utility. In terms of privacy, we can also prove attacker's view satisfies DP (See Appendix A.2). For the same reason in Section 4.2, negative noise has negative influence on utility in a deterministic way. Therefore, though GEO tolerates negative noise (i.e., x_ℓ can be negative), we do not recommend setting $x_\ell < 0$.

The two parameters x_ℓ and p both influence ϵ and utility: For $x_\ell > 0$, increasing x_ℓ reduces both ϵ and utility, and increasing p raises ϵ and utility. For $x_\ell < 0$, utility and privacy varies in different cases. Appendix B studies the parameter settings.

4.4 Double Geometric Mechanism (DGEO)

AKR adds a biased Laplace noise to the number of requests (explained in Section 2.3). Likely, we propose to draw the noise from a biased double geometric distribution:

$$\Pr[d = i] = \frac{1 - e^{-\epsilon}}{1 + e^{-\epsilon}} e^{-\epsilon|i-\mu|},\ \forall i \in \mathbb{Z} \tag{15}$$

We call $s = 1/\epsilon$ the scale of the noise and μ the bias of the noise. Adding double-geometric noise with a scale $1/\epsilon$ to the number of requests satisfies ϵ-DP [17, 40], and we prove it in Appendix A.3.

AKR chooses Laplace noise, which is similar to DGEO but in the continuous domain. AKR sets a positive bias μ so that the probability that the noise is negative is bounded by δ, and the authors prove AKR follows (ϵ, δ)-DP. In order to have a small δ (i.e., the probability of failing DP to be small), μ must be fairly large which leads to unsatisfactory utility. For example, when δ equals a common value of 10^{-6}, μ has to be *at least* 15 (it is even larger than the number of real requests and resources) to achieve $\epsilon = 1$ for $k = 10$.

Hence, accommodating negative noise without using a large bias is essential to high utility, and we show *it is possible*. In a nutshell, negative noise may relax the pre-allocation ϵ, but not necessarily introduce δ. Although negative noise introduces a discrepancy between the possible outcomes of D and D' from the attacker's view, as well as in the range of y (resources dispatched to the attacker), it does not violate DP when combined with non-negative noise as proved in Theorem 8 of Section A.3 (i.e., the attacker's view satisfies DP). In Section 5.2, we provide empirical analysis to show the impact of RA on utility and privacy from the attacker's view.

5 EVALUATION

In this section, we evaluate the privacy and utility of different mechanisms. Here we summarize the key results. 1) Our mechanisms outperform AKR by 11% to 65% in terms of utility (e.g., DGEO outperforms AKR by 53% given $\epsilon = 2$). GEO has a clear advantage for smaller ϵ while DGEO is able to achieve better utility with larger ϵ. 2) Different parameters can achieve similar privacy protection but lead to very different levels of utility.

5.1 Evaluation Setup

Settings. To compare different mechanisms in the privacy-utility tradeoff, we choose to simulate RA using a real-world system setting. Similar to AKR, we take Alpenhorn [42], an MPM, as one of our target systems. In essence, a user in Alpenhorn starts a conversation with his/her friend at an agreed time or round. In a conversation round, the user initiates k channels to k friends, then sends and receives messages on each channel to hide the real communication pattern. Section 2.1 describes how its privacy guarantee can be violated. The evaluation by AKR models how Alpenhorn allocates channels for requests to defend against allocation-based side-channel attacks. Similar to AKR, we set the resource capacity $k = 10$ for most of the experiments, meaning that a user has a maximum of 10 channels that can be established with other clients. We also experiment with larger k (15, 20) to test the scalability of the proposed mechanisms and AKR. AKR sets an upper bound to the number of requests in each round and considers at most 10% of them to be honest requests. We remove the upper bound and set the number of victim requests to at most 1 to simulate the worst case for the victim, as explained in Section 2.1. Note that AKR uses

a Poisson distribution to simulate the request number from all users while our total requests in case D and D' are fixed to m and $m+1$, respectively. Given that we assume at most one victim exists during allocation, we did not apply the Poisson distribution. Unless otherwise stated, we simulate *10 million* independent rounds of allocation with requests of attacker $m = k$ (the optimal attacker strategy, as proved in Theorem 4), and measure privacy and utility.

In Section 6.3, we try to justify the choice of simulation setup and discuss the limitations of simulation.

Metrics. We evaluate the performance of different mechanisms under three metrics: privacy, utility, and waiting overhead. Regarding privacy, we compute the empirical ϵ by Equation 4 with the simulation results, and the larger value indicates more privacy leakage. Theoretical ϵ can be derived from Theorem 5 to Theorem 8, but their values are not always computable. For the study of parameters (Appendix B), we compute some theoretical ϵ for the comparison.

As for the utility, we mainly measure the empirical *resource utilization* U, or how many (in ratio) resources are put into real use after allocation, from the simulation results. This differs from the classic DP that considers the *accuracy* of the analysis results as utility, or how close the noisy output is to the ground truth. The same utility measure is chosen by AKR as well. U is given by:

$$U = \sum_{j=0}^{k} \Pr[r = j] \frac{j}{k} \tag{16}$$

where r is the number of fulfilled requests, k is the number of resources, and $\Pr[r = j]$ is the probability of j requests being fulfilled.

While resource utilization is relevant to the overhead on the allocator, the overhead on the client can be measured by their waiting time (or waiting overhead). We use the probability of the victim getting the resource in any round, as the higher probability should lead to a shorter waiting time for the resource. For example, in Alpenhorn (original version that is not protected by DP) with k resources and m attacker requests, the probability that the victim gets resource $\Pr[V_a]$ is given by:

$$\Pr[V_a] = \frac{\binom{k-1}{m}\binom{1}{1}}{\binom{k}{m+1}} \tag{17}$$

Denoting the $\Pr[V_b]$ as the probability that the victim gets resources after DP, the ratio between $\Pr[V_a]$ and $\Pr[V_b]$ represents the amount of waiting overhead caused by DP mechanisms.

Implementation. We implement our code in Python 3.7.10 with NumPy 1.19.5 libraries. The implementation is open-sourced [14].

5.2 Evaluation Results

We compare the performance of different mechanisms, i.e., CST, UNI, GEO, DGEO, and AKR with simulation.

First, we enumerate different ϵ values for each mechanism and compute the *best* utility value, which is derived by searching in the space of possible mechanism parameters. Figure 2 illustrates the quantitative results of the tradeoff between privacy and utility. Note that, for AKR, since it is (ϵ, δ)-DP, we set $\delta = 10^{-6}$, which is commonly chosen by other DP works (Angel et al. even chooses a larger value, $\delta = 10^{-4}$ [3]).

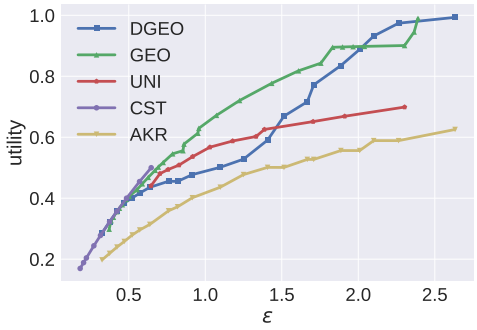


Figure 2: Comparison of different mechanisms. The ranges of ϵ for CST and UNI are limited. CST's utility never exceeds 0.5 because at least k dummy requests are required to make it differentially private. The utility of GEO does not increase when ϵ is between 1.8 to 2.3, and we speculate this is because the parameters leading to the optimal utility have not been discovered through simulation.

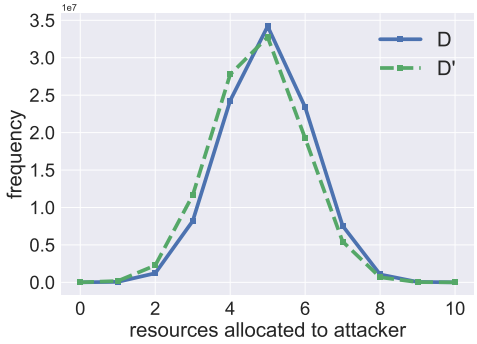


Figure 3: Allocation results by GEO with $p = 0.90$, which sets the bias to 10. The x-axis represents the number of fulfilled requests of the attacker, and the y-axis represents the frequency of each output out of 100 million rounds. We increase the simulation rounds from 10 million to 100 million in order to yield precise results.

In general, we found that all of our proposed mechanisms have better utility than AKR for *every* ϵ when the parameters are fine-tuned. Specifically, GEO has better utility given lower ϵ (i.e., under 2) while DGEO yields better utility given more relaxed ϵ (i.e., over 2). AKR reaches the utility of 0.58 with $(2, 10^{-6})$-DP, while GEO and DGEO are able to achieve the utility of 0.89 with 2-DP, increasing the utility by 0.31 (53%). Overall, the margin of DGEO over AKR ranges from 0.05 to 0.39, GEO is able to outperform AKR over a range of 0.08 to 0.36, and UNI is able to outperform AKR by at most 0.15. This result is surprising as (ϵ, δ)-DP usually yields better utility than ϵ-DP. We believe this is due to the fact that our mechanisms have the ability to accommodate negative noise, while AKR has to use a large bias to satisfy DP.

Since CST cannot achieve a utility value of more than 0.5, in the following experiments, we focus on the other mechanisms. In the previous experiment, we change the mechanism parameters to fit ϵ, but in the real-world deployment, the parameters are determined ahead. Here we evaluate the impact of the parameters related to bias. For DGEO and AKR, they are represented by μ. For UNI and

GEO, the starting point (x_ℓ for UNI and GEO) models the bias. We configure bias to a small value of 10. With the utility targeting 0.5, GEO and DGEO are able to bound privacy with ϵ of 0.80 and 0.77. UNI and AKR result in much higher ϵ at 1.28 and 1.5. Hence, with a small bias, our mechanisms can protect allocation with better privacy while achieving the same utility as AKR.

So far, the prior experiments quantitatively measure how the mechanisms perform. Like Angel et al. [3], we visualize privacy protection under a fixed set of parameters. Specifically, we measure the difference in allocation results (D and D') based on the number of resources allocated to the attacker. Figure 3 shows the visualization of GEO, when bias is configured to 10. The lines of D and D' stay close, suggesting the privacy leakage of GEO is small.

Regarding the waiting overhead $\Pr[V_a]$, we found UNI, GEO, DGEO and AKR reach 1.45, 1.92, 1.91 and 1.88 when configuring bias to 10, suggesting our mechanisms either have similar or lower waiting overhead than AKR. Still, we acknowledge that such overhead is significant and we discuss this issue in Section 6.3.

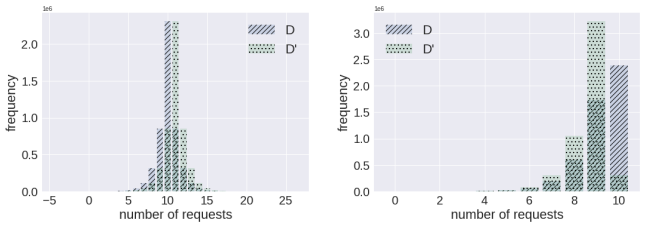
Impact of Parameters. To assess the impact of mechanism parameters, we compute the privacy and utility values theoretically, as explained in Section 5.1. Here we summarize the guideline for setting parameters and leave the details to Appendix B.

For UNI, one should avoid large x_r as the privacy benefit diminishes and utility drops noticeably. Regarding x_ℓ, we found negative values do not offer good privacy and small x_ℓ is necessary to maintain good privacy. For GEO, a negative starting point x_ℓ should be avoided as it does no good to utility or privacy. We suggest that a small positive starting point x_ℓ with a moderately high p value would be optimal for GEO. For example, an x_ℓ of 3 with $p = 0.7$ can achieve reasonable privacy $\epsilon = 1.24$ and a good utility of 0.75. Our evaluation in Appendix B also indicates that a small positive bias μ with a scale s around 1 would be optimal for DGEO. For the larger resource capacity k, GEO and DGEO still perform well.

Why Models Attacker's View. In Section 4.4, we argue that modeling the attacker view is better than modeling the whole view that is adopted by AKR. Here we justify this claim under the same simulation. Figure 4 shows an example with the zero-mean ($\mu = 0$) double-geometric distribution under simulation. Given two different cases D and D', Figure 4a depicts the difference of output before allocation and Figure 4b shows the output after allocation from the attacker's view, assuming DGEO with scale 1 is applied to allocate $k = 10$ resources, and D contains $m = 10$ requests. Our study shows that the existence of the victim can drastically affect the portion of resources an attacker can get after allocation.

Original ϵ used	4	2	1	0.5	0.2
Empirical ϵ before RA	4.00	2.00	1.03	0.54	0.27
Empirical ϵ after RA	3.28	2.26	2.01	1.90	1.79
Theoretical bound of ϵ after RA	3.29	2.26	2.07	1.91	1.79

Table 3: Comparison of different settings of DGEO with $k = 10$. We use 5 different ϵ values (first row). Row 2 shows the empirical ϵ is close to the original ϵ, which indicates our simulation has only small errors. Row 3 is the empirical ϵ after RA, which deviates from the original ϵ. The last row shows our theoretical bound of ϵ given in Theorem 8 is close to the empirical value.



(a) Before RA. **(b) After RA.**

Figure 4: Distribution of output over 5 million runs. Before RA, we draw noise from a double geometric distribution with $\epsilon = 1$ and $k = 10$. After RA, the distribution changes, and the privacy leakage increases (the empirical ϵ rises to 2.07).

Table 3 uses four different zero-mean double geometric distributions to further explain why RA itself should be part of the privacy modeling. First, when the original ϵ decreases, more noise is expected, which leads to an increase in privacy protection for both before and after RA. However, given a relatively high scale (i.e., small ϵ), the privacy protection after RA can be 6 times worse than that before RA. Such extra information leakage is an indicator that the privacy budget is affected by RA.

We also take a step forward to measure the privacy amplification caused by modeling the attacker's view. We adjust AKR by replacing its Laplace noise with double geometric noise, which we denote as AKR-DGEO, and compare it with DGEO. As their noise mechanisms become the same, we can exemplify the privacy-utility tradeoff without and with privacy amplification. Our empirical analyses indicate that, for a utility measure of approximately 0.42, privacy amplification results in a decrease in the privacy parameter ϵ from 1.00 to 0.59. Likewise, when the utility measure is near 0.60, ϵ diminishes from 2.00 to 1.43 after amplification.

6 DISCUSSION

6.1 Privacy Consumption over Multiple Rounds

Like Angel et al. [3], our analysis focuses on a single round. Privacy normally degrades over multiple rounds rapidly. For instance, naively applying the sequential composition property of DP over multiple rounds deteriorates the privacy guarantee (i.e., ϵ) linearly. Inspired by previous work, we identify three ways to curb privacy consumption: (1) using advanced composition [48] to reduce the total ϵ, (2) reusing noise for repeated requests [21, 59], and (3) bounding the number of requests. Though relaxations could happen for the attacker's background knowledge [16], our approach does not limit the attacker's background knowledge but rather their view, and therefore we believe composition works in our case. Next, we discuss how the three methods can be applied in more detail.

Using Advanced Composition. Traditional composition theorem in DP may result in a union bound over noise, which is suboptimal. Avoiding union bound for multiple queries has been an important open problem in differential privacy [58]. The well-known advanced composition theorem [18] adjusts pure DP to approximate (ϵ, δ)-DP with $\delta > 0$ to yield better composition results. In cases where the attacker interacts with the allocator over multiple rounds, we argue that the leakage can be modeled by the k-fold adaptive composition [18].

Mironov [48] proposed new bounding techniques for advanced composition under Rényi DP (RDP) to this end. In our case, we can transform ϵ-DP to (α, ϵ)-RDP for any $\alpha > 0$ [48], compose RDP with Theorem 1, and transform back to (ϵ, δ)-DP with Theorem 2. Popular DP libraries like Opacus have supported RDP advanced composition [53]. Alternatively, we can utilize Equations 4 and 5 in [52] to derive the (ϵ, δ)-DP bound directly and employ numerical methods [26] to obtain more accurate results.

Yet, it is an open question to directly prove the RDP guarantee for our mechanisms (to avoid conversions mentioned above and compose better). One possible route is to follow the proof of the discrete Gaussian mechanism [11] and we leave it as a future work.

Reusing Noise. When new incoming requests are from the same set of clients of the previous round, the server can avoid consuming an extra privacy budget by reusing the noise generated for the previous round [21, 59]. In this way, the attacker gains no more information than the previous round while the server consumes no extra budget. Specifically, the output of the algorithm remains the same if we fix the randomness that happens in a certain round. Thus, the server can utilize a persistent secret key for a pseudo-random function (PRF) over the same set of clients, where in each round the server is able to simulate the same randomness for the same set of clients.

Bounding the Number of Requests. Drawing from [19], we can simplify the privacy analysis by eliminating the need to consider every RA round for each client by capping client requests over a period (e.g., a maximum of 2 calls daily for MPM clients).

6.2 Other Settings

Though our study primarily examines clients submitting binary requests for a single resource under worst-case privacy, it can be extended to (1) the non-binary setting in which clients can submit requests for more than one resource, (2) the multi-resource setting in which there are multiple kinds of resources and clients can request arbitrary resources, and (3) the average-case privacy.

Non-binary Requests that *Can* be Fulfilled Partially. This setting can be transformed into the binary case by casting each non-binary request as multiple binary requests. The global sensitivity will be changed to the number of maximum requests per client.

Non-binary Requests that *Cannot* be Fulfilled Partially. The problem is transformed into an optimization problem aiming for maximum utilization of resources [36]. In general, the allocator picks the requests that maximize its target function. The allocator can add noise to the number of requests, which we expect to yield worse utility compared to our primary setting. This is because when requests for large resources are added or removed from the allocator, a great amount of resources are wasted.

Multiple-resources Allocation. A multiple-resource allocator deals with multiple types of resources simultaneously. In this setting, the privacy protection of the allocator subjects to sequential composition, thus the overall privacy depends on the summation of all privacy losses. The intuition is that the privacy leakage of each allocation can be seen as auxiliary information, and be combined with leakage from allocations of other types of resources.

Multiple Honest Requests. Multiple honest requests in allocation happen when the attacker is not strong enough to control all other clients except the victim. Assume the requests are binary in this setting and the attacker does not know the resource distribution among the honest requests. In this case, the honest requests (other than the one from the victim) are equivalent to the dummy requests in our primary setting because the distribution among them remains unknown to the attacker. Therefore, we can add less noise in this setting in order to achieve the same privacy guarantee. We have justified the above assumption by experimenting with DGEO (the results are omitted due to page limit).

6.3 Limitations

Empirical Study on Privacy. The privacy analysis in our evaluation is empirical-based (i.e., ϵ's are calculated empirically based on our simulation result). We choose simulation for two main reasons. First, we aim to compare the privacy-utility tradeoff of different mechanisms at different privacy parameters (e.g., Figure 2), and the computational overhead will be very high if the experiments are executed on large-scale real-world systems. Second, for the MPM system we evaluate, there is no published dataset about its communication data, so we have to simulate the allocations. In fact, Angel et al. took a similar approach to evaluate privacy empirically [3], and the scale of our simulation is comparable or larger (from 5 million rounds to 100 million rounds). Simulation has been leveraged to evaluate other privacy-preserving systems for the same reason, like differentially oblivious databases [54]. We also acknowledge the limitation of our simulation, which does not fully approximate real-world, large-scale systems.

Efficiency. Adding dummies results in higher waiting overhead because the clients now need to go through more rounds in order to get the desired resources. However, once the resources are allocated, no additional delay should be observed.

The spatial overhead due to serving the dummy clients could be prominent, especially for systems that operate on very limited resources. The same limitation exists in AKR, and the overhead is often unavoidable for systems leveraging DP. On the other hand, our approach provides better resource utilization than AKR, e.g., 98% under DGEO and 59% under AKR when $\epsilon = 2.3$. Higher resource utilization also leads to smaller waiting overhead. For example, for an approach with 40% utilization, the chances for a user to get resource allocated within 5 dialing rounds in Alpenhorn is about 99%. Our proposed mechanisms all surpass 40% as shown in Table 2.

Attacks against DPRA. Potential side-channel attacks against DP algorithms, such as timing attacks [32], may compromise our DPRA, but require adaptation to the RA setting.

6.4 Real-world Examples and Utility Analysis

Here we first give a few examples of how the noise under $d \geq 0$ and $d < 0$ can be instantiated in real-world systems. We follow the basic setting as described in Section 2.1 first (i.e., all resources are identical and one request asks for one piece of resource).

- In the cloud setting, users request for VMs and whether they are served is based on the available resources like CPU and memory. When $d > 0$, the allocator creates dummy VMs that potentially occupy resources. When $d < 0$, not all the requested resources are allocated to the VM (even though there are available resources).
- Inside a computer, requests to cache resources (e.g., cache ways) are automatically generated during a memory access, which can lead to cache side-channel attacks [66]. $d > 0$ will assign cache ways to dummy programs and $d < 0$ will skip the caching of some memory content. Either option will reduce the accuracy of the attack which relies on cache contention between attacker and victim.
- In MPM, the requests are from a user's friends who intend to start a conversation in a round. Noise $d > 0$ is to add fake friends and $d < 0$ means to reject some requests.

For more complex allocators, we can extend the DP mechanisms following Section 6.2. For example, the buddy system manages memory in the power of two increments [37] and we can support it by considering the memory requests as non-binary. When concurrent requests are supported by multiple resource pools (e.g., hypervisor resource pools [62]), multiple-resources allocation can be applied.

Regarding the results of the privacy-utility tradeoff (e.g., summarized in Table 2), we argue they are practical in the real-world setting. For example, a study of Google Cloud shows the resource utilization is 40% - 60% and the resource waste due to early task termination is 4.53 - 14.22% [23]. In this case, the utility after DGEO and GEO should be acceptable (e.g., 0.82 for GEO at $\epsilon = 1.7$).

7 RELATED WORK

Joint DP. We focus on the partial view of the attacker. The Joint DP definition proposed by Kearns et al. [36] formalizes this intuition, primarily to compute equilibrium in games with incomplete information [36, 55, 56]. Note that Joint DP is just a definition, and classic DP primitives like the Laplace mechanism are still used. We are the first to formally investigate the design space and adapt various DP mechanisms to RA.

Private Matching and Allocation. Our problem can be seen as a variation of the private allocation/matching problem, through which users have (non-binary) valuations for products (potentially in multiple rounds), and the goal is to maximize welfare while protecting users' private value for each good. Existing works [10, 15, 29, 34, 35, 50] have applied DP algorithms (e.g., Laplace mechanism) that are asymptotically interesting. Our modeling of RA is different and we explored different noisy mechanisms.

Biased Noise. AKR employs biased noise to satisfy DP, while DGEO uses it to improve the privacy-utility tradeoff. Biased noise has been examined before. Mazloom and Gordon [47] introduced a modified 2-sided geometric distribution to generate noise that enables differentially private access patterns with high efficiency. DJoin [51] cuts Laplace noise at zero to provide distributed queries with DP. Shrinkwrap [7] offers a truncated Laplace mechanism for differentially private data federation, where dummies are introduced to pad intermediate results. He et al. [28] proposes a model for private record linkage, allowing the disclosure of the true matching records while keeping the protocol executions indistinguishable when non-matching records are replaced.

DP Against Side-channel Leakage. The leakage from RA can be considered as allocation-based side channels [3]. A more common type of side channel is consumption-based, which happens when the system resources (e.g., network bandwidth and cache) are consumed. A number of works have applied DP to protect the system against the latter type of leakage. The protected resources/services include `procfs` of system statistics [64], streaming traffic [67], Trusted Execution Environment (TEE) [65], health data (e.g., ECG data) [57], task schedules [13], and packet scheduler [8].

Another related line of work is differentially oblivious [12], which was proposed to address the fundamental limitation of ORAM (Oblivious RAM). Though ORAM can protect the program's secret by hiding its memory access pattern, it incurs a very high performance overhead. By converting full obliviousness to differential obliviousness, one can obtain meaningful privacy with little overhead [12, 38, 63]. While this paper also hides a victim's secret (i.e., its existence at a certain time), it considers an orthogonal adversary model where the attacker observes part of the true results without any mechanism to hide the victim-related information.

8 CONCLUSION

In this paper, we studied the problem of privacy protection designated under resource allocation and systematically modeled it through the lens of differential privacy. Specifically, we identified the key issues of a prior system AKR and propose to consider negative noise and mechanisms other than the standard Laplace noise. We designed four different mechanisms, CST, UNI, GEO, and DGEO, and proved they all satisfy ϵ-DP. In both theoretical and empirical analysis, we found our mechanisms outperform AKR in utility ranging from 11% to 65% given a privacy budget ϵ. Among the proposed mechanisms, we recommend GEO, which has a good privacy-utility tradeoff and performs especially well when ϵ is small (e.g., less than 2). Ultimately, we hope to use this work to attract more attention to the privacy issues of resource allocation and encourage new privacy-preserving solutions to be designed.

ACKNOWLEDGMENTS

We thank our shepherd and anonymous reviewers for their valuable suggestions. This project was supported by NSF CNS-2220434, CNS-2220433 and OAC-2319988. We thank Yiyang Hu from UCInspire for the help.

REFERENCES

[1] Dakshi Agrawal and Dogan Kesdogan. 2003. Measuring anonymity: The disclosure attack. *IEEE Security & privacy* 1, 6 (2003), 27–34.
[2] Sebastian Angel, Hitesh Ballani, Thomas Karagiannis, Greg O'Shea, and Eno Thereska. 2014. End-to-end performance isolation through virtual datacenters. In *11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14)*. 233–248.
[3] Sebastian Angel, Sampath Kannan, and Zachary Ratliff. 2020. Private resource allocators and their applications. In *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 372–391.
[4] Sebastian Angel, David Lazar, and Ioanna Tzialla. 2018. What's a little leakage between friends?. In *Proceedings of the 2018 Workshop on Privacy in the Electronic Society*. 104–108.

[5] Borja Balle, Gilles Barthe, and Marco Gaboardi. 2018. Privacy amplification by subsampling: Tight analyses via couplings and divergences. *Advances in Neural Information Processing Systems* 31 (2018).

[6] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. 2019. The privacy blanket of the shuffle model. In *Annual International Cryptology Conference*.

[7] Johes Bater, Xi He, William Ehrich, Ashwin Machanavajjhala, and Jennie Rogers. 2018. Shrinkwrap: efficient sql query processing in differentially private data federations. *Proceedings of the VLDB Endowment* 12, 3 (2018), 307–320.

[8] Andrew Beams, Sampath Kannan, and Sebastian Angel. 2021. Packet scheduling with optional client privacy. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 3415–3430.

[9] Anton Beloglazov, Jemal Abawajy, and Rajkumar Buyya. 2012. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. *Future generation computer systems* 28, 5 (2012), 755–768.

[10] Felix Brandt. 2006. How to obtain full privacy in auctions. *International Journal of Information Security* 5, 4 (2006), 201–216.

[11] Clément L Canonne, Gautam Kamath, and Thomas Steinke. 2020. The discrete gaussian for differential privacy. *Advances in Neural Information Processing Systems* 33 (2020), 15676–15688.

[12] TH Hubert Chan, Kai-Min Chung, Bruce M Maggs, and Elaine Shi. 2019. Foundations of differentially oblivious algorithms. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2448–2467.

[13] Chien-Ying Chen, Debopam Sanyal, and Sibin Mohan. 2021. Indistinguishability Prevents Scheduler Side Channels in Real-Time Systems. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 666–684.

[14] Joann Qiongna Chen. 2023. Code of this project. https://github.com/dpra-dp/dpra.

[15] Rachel Cummings, Michael Kearns, Aaron Roth, and Zhiwei Steven Wu. 2015. Privacy and truthful equilibrium selection for aggregative games. In *International Conference on Web and Internet Economics*. Springer, 286–299.

[16] Damien Desfontaines and Balázs Pejó. 2020. Sok: differential privacies. *Proceedings on privacy enhancing technologies* 2020, 2 (2020), 288–313.

[17] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*. Springer, 265–284.

[18] Cynthia Dwork, Guy N Rothblum, and Salil Vadhan. 2010. Boosting and differential privacy. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE, 51–60.

[19] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. 2018. Amplification by Shuffling: From Local to Central Differential Privacy via Anonymity. *arXiv preprint arXiv:1811.12469* (2018).

[20] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. 2019. Amplification by shuffling: From local to central differential privacy via anonymity. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2468–2479.

[21] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response. In *CCS*.

[22] Vitaly Feldman, Ilya Mironov, Kunal Talwar, and Abhradeep Thakurta. 2018. Privacy amplification by iteration. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 521–532.

[23] Peter Garraghan, Paul Townend, and Jie Xu. 2013. An analysis of the server characteristics and resource utilization in google cloud. In *2013 IEEE International Conference on Cloud Engineering (IC2E)*. IEEE, 124–131.

[24] Quan Geng and Pramod Viswanath. 2015. Optimal noise adding mechanisms for approximate differential privacy. *IEEE Transactions on Information Theory* 62, 2 (2015), 952–969.

[25] Arpita Ghosh, Tim Roughgarden, and Mukund Sundararajan. 2012. Universally utility-maximizing privacy mechanisms. *SIAM J. Comput.* 41, 6 (2012), 1673–1693.

[26] Sivakanth Gopi, Yin Tat Lee, and Lukas Wutschitz. 2021. Numerical composition of differential privacy. *Advances in Neural Information Processing Systems* 34 (2021), 11631–11642.

[27] Abdul Hameed, Alireza Khoshkbarforoushha, Rajiv Ranjan, Prem Prakash Jayaraman, Joanna Kolodziej, Pavan Balaji, Sherali Zeadally, Qutaibah Marwan Malluhi, Nikos Tziritas, Abhinav Vishnu, et al. 2016. A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems. *Computing* 98, 7 (2016), 751–774.

[28] Xi He, Ashwin Machanavajjhala, Cheryl Flynn, and Divesh Srivastava. 2017. Composing differential privacy and secure computation: A case study on scaling private record linkage. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1389–1406.

[29] Justin Hsu, Zhiyi Huang, Aaron Roth, Tim Roughgarden, and Zhiwei Steven Wu. 2014. Private matchings and allocations. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*. 21–30.

[30] Hameed Hussain, Saif Ur Rehman Malik, Abdul Hameed, Samee Ullah Khan, Gage Bickler, Nasro Min-Allah, Muhammad Bilal Qureshi, Limin Zhang, Wang Yongji, Nasir Ghani, et al. 2013. A survey on resource allocation in high performance distributed computing systems. *Parallel Comput.* 39, 11 (2013), 709–736.

[31] Boqi Jia, Honglin Hu, Yu Zeng, Tianheng Xu, and Yang Yang. 2018. Double-matching resource allocation strategy in fog computing networks based on cost efficiency. *Journal of Communications and Networks* 20, 3 (2018), 237–246.

[32] Jiankai Jin, Eleanor McMurtry, Benjamin IP Rubinstein, and Olga Ohrimenko. 2022. Are we there yet? timing and floating-point attacks on differential privacy systems. In *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 473–488.

[33] Ramesh Johari and John N Tsitsiklis. 2004. Efficiency loss in a network resource allocation game. *Mathematics of Operations Research* 29, 3 (2004), 407–435.

[34] Laura Kang and David C Parkes. 2006. Passive verification of the strategyproofness of mechanisms in open environments. In *Proceedings of the 8th international conference on Electronic commerce: The new e-commerce: innovations for conquering current barriers, obstacles and limitations to conducting successful business on the internet*. 19–30.

[35] Sampath Kannan, Jamie Morgenstern, Aaron Roth, and Zhiwei Steven Wu. 2014. Approximately stable, school optimal, and student-truthful many-to-one matchings (via differential privacy). In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*. SIAM, 1890–1903.

[36] Michael Kearns, Mallesh Pai, Aaron Roth, and Jonathan Ullman. 2014. Mechanism design in large games: Incentives and privacy. In *Proceedings of the 5th conference on Innovations in theoretical computer science*. 403–410.

[37] Kenneth C Knowlton. 1965. A fast storage allocator. *Commun. ACM* 8, 10 (1965).

[38] Ilan Komargodski and Elaine Shi. 2021. Differentially Oblivious Turing Machines. In *12th Innovations in Theoretical Computer Science Conference (ITCS 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

[39] J Kok Konjaang, JY Maipan-uku, and Kumangkem Kennedy Kubuga. 2016. An efficient max-min resource allocator and task scheduling algorithm in cloud computing environment. *arXiv preprint arXiv:1611.08864* (2016).

[40] Yu-Hsuan Kuo, Cho-Chun Chiu, Daniel Kifer, Michael Hay, and Ashwin Machanavajjhala. 2018. Differentially private hierarchical count-of-counts histograms. *arXiv preprint arXiv:1804.00370* (2018).

[41] David Lazar, Yossi Gilad, and Nickolai Zeldovich. 2018. Karaoke: Distributed private messaging immune to passive traffic analysis. In *13th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 18)*. 711–725.

[42] David Lazar and Nickolai Zeldovich. 2016. Alpenhorn: Bootstrapping secure communication without leaking metadata. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*. 571–586.

[43] Dong Li, Minsoo Rhu, Daniel R Johnson, Mike O'Connor, Mattan Erez, Doug Burger, Donald S Fussell, and Stephen W Redder. 2015. Priority-based cache allocation in throughput processors. In *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 89–100.

[44] Ratul Mahajan, Steven M Bellovin, Sally Floyd, John Ioannidis, Vern Paxson, and Scott Shenker. 2002. Controlling high bandwidth aggregates in the network. *ACM SIGCOMM Computer Communication Review* 32, 3 (2002), 62–73.

[45] Nayantara Mallesh and Matthew Wright. 2010. The reverse statistical disclosure attack. In *International Workshop on Information Hiding*. Springer, 221–234.

[46] Carlo Mastroianni, Michela Meo, and Giuseppe Papuzzo. 2011. Self-economy in cloud data centers: Statistical assignment and migration of virtual machines. In *European Conference on Parallel Processing*. Springer, 407–418.

[47] Sahar Mazloom and S Dov Gordon. 2018. Secure computation with differentially private access patterns. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. 490–507.

[48] Ilya Mironov. 2017. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*. IEEE, 263–275.

[49] Amir Nahir, Ariel Orda, and Danny Raz. 2015. Resource allocation and management in cloud computing. In *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*. IEEE, 1078–1084.

[50] Moni Naor, Benny Pinkas, and Reuban Sumner. 1999. Privacy preserving auctions and mechanism design. In *Proceedings of the 1st ACM Conference on Electronic Commerce*. 129–139.

[51] Arjun Narayan and Andreas Haeberlen. 2012. DJoin: Differentially private join queries over distributed databases. In *10th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 12)*. 149–162.

[52] Milad Nasr, Shuang Songi, Abhradeep Thakurta, Nicolas Papernot, and Nicholas Carlin. 2021. Adversary instantiation: Lower bounds for differentially private machine learning. In *2021 IEEE Symposium on security and privacy (SP)*. IEEE.

[53] Pytorch. 2022. Opacus RDP. https://github.com/pytorch/opacus/blob/main/opacus/accountants/rdp.py.

[54] Lianke Qin, Rajesh Jayaram, Elaine Shi, Zhao Song, Danyang Zhuo, and Shumo Chu. 2022. Adore: Differentially Oblivious Relational Database Operators. *Proceedings of the VLDB Endowment* 16, 4 (2022), 842–855.

[55] Ryan M Rogers and Aaron Roth. 2014. Asymptotically truthful equilibrium selection in large congestion games. In *Proceedings of the fifteenth ACM conference on Economics and computation*. 771–782.

[56] Aaron Roth. 2013. Differential privacy, equilibrium, and efficient allocation of resources. In *2013 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 1593–1597.

[57] Nazir Saleheen, Supriyo Chakraborty, Nasir Ali, Md Mahbubur Rahman, Syed Monowar Hossain, Rummana Bari, Eugene Buder, Mani Srivastava, and Santosh Kumar. 2016. mSieve: differential behavioral privacy in time series of mobile sensor data. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. 706–717.

[58] Thomas Steinke and Jonathan Ullman. 2021. Open problem - avoiding the union bound for multiple queries. https://differentialprivacy.org/open-problem-avoid-union/.

[59] Pratiksha Thaker, Mihai Budiu, Parikshit Gopalan, Udi Wieder, and Matei Zaharia. 2020. Overlook: Differentially Private Exploratory Visualization for Big Data. *arXiv preprint arXiv:2006.12018* (2020).

[60] Nirvan Tyagi, Yossi Gilad, Derek Leung, Matei Zaharia, and Nickolai Zeldovich. 2017. Stadium: A distributed metadata-private messaging system. In *Proceedings of the 26th Symposium on Operating Systems Principles*. 423–440.

[61] Jelle Van Den Hooff, David Lazar, Matei Zaharia, and Nickolai Zeldovich. 2015. Vuvuzela: Scalable private messaging resistant to traffic analysis. In *Proceedings of the 25th Symposium on Operating Systems Principles*. 137–152.

[62] VMware. 2019. Managing Resource Pools. https://docs.vmware.com/en/VMware-vSphere/8.0/vsphere-resource-management/GUID-60077B40-66FF-4625-934A-641703ED7601.html.

[63] Sameer Wagh, Paul Cuff, and Prateek Mittal. 2018. Differentially Private Oblivious RAM. *Proceedings on Privacy Enhancing Technologies* 4 (2018), 64–84.

[64] Qiuyu Xiao, Michael K Reiter, and Yinqian Zhang. 2015. Mitigating storage side channels using statistical privacy mechanisms. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. 1582–1594.

[65] Min Xu, Antonis Papadimitriou, Ariel Feldman, and Andreas Haeberlen. 2018. Using differential privacy to efficiently mitigate side channels in distributed analytics. In *Proceedings of the 11th European Workshop on Systems Security*. 1–6.

[66] Yuval Yarom and Katrina Falkner. 2014. FLUSH+ RELOAD: A high resolution, low noise, L3 cache side-channel attack. In *USENIX Security Symposium 2014*.

[67] Xiaokuan Zhang, Jihun Hamm, Michael K Reiter, and Yinqian Zhang. 2019. Statistical privacy for streaming traffic. In *Proceedings of the 26th ISOC Symposium on Network and Distributed System Security*.

A PROOFS OF ϵ-DP FOR NOISY MECHANISMS

A.1 Uniform Mechanism

Theorem 6. *Assume the server has k resources. Adding a random noise drawn uniformly from $\{x_\ell, x_\ell+1, \ldots, x_r\}$ (both x_ℓ and $x_r \geq k$ are integers) to the number of requests satisfies ϵ-DP, where*

$$e^\epsilon \leq \max_{y\in Y}\left(\frac{f(y)+\sum_{i=\max(x_\ell,k-y)}^{x_r} g(i)}{f_1(y)+\sum_{i=\max(x_\ell,(k-y-1)_+)}^{x_r} g(i+1)}\right) \tag{18}$$

where $Y=\{1,2,\ldots,k\}$, $g(i)=\frac{(k!)^2(i!)^2}{y!((k-y)!)^2(i-k+y)!(k+i)!}$,
$f(y)=\mathbb{1}_{x_\ell+k\leq y<k}+(-k-x_\ell)\,\mathbb{1}_{y=0,x_\ell<-k}$,
$f_1(y)=\frac{k-y+1}{k+1}\mathbb{1}_{x_\ell+k+1\leq y\leq k}+\frac{y+1}{k+1}\mathbb{1}_{x\ell+k\leq y<k}+$
$(-k-1-x_\ell)\,\mathbb{1}_{y=0,x_\ell<-k-1}$.

Assume an allocator has k resources. W.l.o.g., D contains m requests and D' contains $m+1$ requests. Before going further into examination of the privacy, we first consider the value of m. For the view of an attacker, it is crucial to set m to an optimal value that causes maximum leakage during allocation. This optimal value is k is shown in previous analysis in Section 3.2.

We examine the probability the attacker gets assigned y resources after allocation. In the case of D,

$$\begin{aligned}
&\Pr\left[\mathsf{View}^{\mathcal{A}}_{\mathcal{M}}(D)=y\right]=\sum_{i=-\infty}^{+\infty}\Pr[d=i]\Pr[y\mid |D|+d]\\
=&\sum_{i=\max(x_\ell,k-y)}^{x_r}\Pr[d=i]\Pr[y\mid |D|+d]+\Pr[d=y-k]\cdot 1\\
&+\Pr[d<-k]\,\mathbb{1}_{y=0}\\
=&\frac{1}{x_r-x_\ell+1}\left(\sum_{i=\max(x_\ell,k-y)}^{x_r}\frac{\binom{m}{y}\binom{i}{k-y}}{\binom{m+i}{k}}\right.\\
&\left.+\mathbb{1}_{x_\ell+k\leq y<k}+(-k-x_\ell)\,\mathbb{1}_{y=0,x_\ell<-k}\right)\\
=&\frac{1}{x_r-x_\ell+1}\left(\mathbb{1}_{x_\ell+k\leq y<k}+(-k-x_\ell)\,\mathbb{1}_{y=0,x_\ell<-k}\right.\\
&\left.+\sum_{i=\max(x_\ell,k-y)}^{x_r}\frac{k!k!i!i!}{y!\,((k-y)!)^2\,(i-k+y)!(k+i)!}\right)
\end{aligned}$$

Similarly, for the case of D',

$$\begin{aligned}
&\Pr\left[\mathsf{View}^{\mathcal{A}}_{\mathcal{M}}(D')=y\right]=\sum_{i=-\infty}^{+\infty}\Pr[d=i]\Pr\left[y\mid \left|D'\right|+d\right]\\
=&\sum_{i=\max(x_\ell,k-y-1,0)}^{x_r}\Pr[d=i]\Pr\left[y\mid \left|D'\right|+d\right]\\
&+\Pr[d<-k-1]\,\mathbb{1}_{y=0}\\
&+\frac{k-y+1}{k+1}\Pr[d=y-k-1]+\frac{y+1}{k+1}\Pr[d=y-k]\\
=&\frac{1}{x_r-x_\ell+1}\left(\sum_{i=\max(x_\ell,k-y-1,0)}^{x_r}\frac{\binom{m}{y}\binom{i+1}{k-y}}{\binom{m+i+1}{k}}+\left(\frac{y+1}{k+1}\right)\mathbb{1}_{x_\ell+k\leq y<k}\right.\\
&\left.+\frac{k-y+1}{k+1}\mathbb{1}_{x_\ell+k+1\leq y\leq k}+(-k-1-x_\ell)\,\mathbb{1}_{y=0,x_\ell<-k-1}\right)\\
=&\frac{1}{x_r-x_\ell+1}\left(\frac{k-y+1}{k+1}\mathbb{1}_{x_\ell+k+1\leq y\leq k}+\frac{y+1}{k+1}\mathbb{1}_{x\ell+k\leq y<k}\right.\\
&+(-k-1-x_\ell)\,\mathbb{1}_{y=0,x_\ell<-k-1}+\\
&\left.\sum_{i=\max(x_\ell,(k-y-1)_+)}^{x_r}\frac{(k!)^2\,((i+1)!)^2}{y!(k-y)!^2(i+1-k+y)!(k+i+1)!}\right)
\end{aligned}$$

Therefore, privacy protection here satisfies

$$e^\epsilon \leq \max_{y\in Y}\left(\frac{f(y)+\sum_{i=\max(x_\ell,k-y)}^{x_r} g(i)}{f_1(y)+\sum_{i=\max(x_\ell,(k-y-1)_+)}^{x_r} g(i+1)}\right)$$

where $Y=\{1,2,\ldots,k\}$, $g(i)=\frac{(k!)^2(i!)^2}{y!((k-y)!)^2(i-k+y)!(k+i)!}$,
$f(y)=\mathbb{1}_{x_\ell+k\leq y<k}+(-k-x_\ell)\,\mathbb{1}_{y=0,x_\ell<-k}$,
$f_1(y)=\frac{k-y+1}{k+1}\mathbb{1}_{x_\ell+k+1\leq y\leq k}+\frac{y+1}{k+1}\mathbb{1}_{x\ell+k\leq y<k}$
$+(-k-1-x_\ell)\,\mathbb{1}_{y=0,x_\ell<-k-1}$.

A.2 One-sided Geometric Mechanism

Theorem 7. *Assume the server has k resources. Adding a random noise drawn from the geometric distribution (with parameter p and starting from integer x_ℓ) to the number of requests satisfies ϵ-DP, where*

$$e^\epsilon \leq \max_{y\in Y}\left(\frac{f(y)+\frac{p((k)!)^2}{y!((k-y)!)^2}\sum_{i=\max(k-y,x_\ell)}^{\infty} g(i)}{f_1(y)+\frac{p(1-p)^{-1}((k)!)^2}{y!((k-y)!)^2}\sum_{i=x_0}^{\infty} g(i+1)}\right)$$

and $Y=\{0,1,\ldots,k\}$, $g(i)=\frac{(1-p)^{i-x_\ell}(i!)^2}{(i-k+y)!(k+i)!}$,
$f(y)=(1-p)^{y-k-x_\ell}\mathbb{1}_{k+x_\ell\leq y<k}+\left(1-(1-p)^{-k-x_\ell}\right)\mathbb{1}_{y=0,x_\ell<-k}$,
$f_1(y)=\frac{p}{k+1}(-y+k+1)(1-p)^{y-k-1-x_\ell}\mathbb{1}_{k+1+x_\ell\leq y\leq k}+\frac{1+y}{k+1}p(1-p)^{y-k-x_\ell}\mathbb{1}_{k+x_\ell\leq y<k}+\left(1-(1-p)^{-k-x_\ell-1}\right)\mathbb{1}_{y=0,x_\ell<-k-1}$, $x_0=\max((k-y-1)_+,x_\ell)$.

We provide the detailed proof as follows. Given an allocator with k resources and an attacker sending $m = k$ requests, we assess the probability of the attacker being allocated y resources.

$$\begin{aligned}
&\Pr\left[\mathsf{View}^{\mathcal{A}}_{\mathcal{M}}(D) = y\right] \\
=&\sum_{i=\max(k-y,x_\ell)}^{\infty} p(1-p)^{i-x_\ell}\frac{\binom{m}{y}\binom{i}{k-y}}{\binom{m+i}{k}} + f(y) \\
=&\frac{p(k!)^2}{y!\left((k-y)!\right)^2}\sum_{i=\max(k-y,x_\ell)}^{\infty}\frac{(1-p)^{i-x_\ell}\,(i!)^2}{(i-k+y)!(k+i)!} + f(y)
\end{aligned}$$

where
$f(y) = p(1-p)^{y-k-x_\ell}\mathbb{1}_{k+x_\ell\le y<k} + \left(1-(1-p)^{-k-x_\ell}\right)\mathbb{1}_{y=0,x_\ell<-k}$.
Similarly, for the other case,

$$\begin{aligned}
&\Pr\left[\mathsf{View}^{\mathcal{A}}_{\mathcal{M}}(D') = y\right] \\
=&\sum_{i=x_0}^{\infty} p(1-p)^{i-x_\ell}\frac{\binom{m}{y}\binom{i+1}{k-y}}{\binom{m+i+1}{k}} + f'(y) \\
=&\sum_{i=x_0}^{\infty}\frac{p(1-p)^{i-x_\ell}(k!)^2((i+1)!)^2}{y!((k-y)!)^2(i+1-k+y)!(k+i+1)!} + f_1(y)
\end{aligned}$$

where $x_0 = \max((k-y-1)_+, x_\ell)$ and

$$\begin{aligned}
f_1(y) =&\frac{(-y+k+1)p(1-p)^{y-k-1-x_\ell}}{k+1}\mathbb{1}_{k+1+x_\ell\le y\le k} \\
&+\frac{(1+y)p(1-p)^{y-k-x_\ell}}{k+1}\mathbb{1}_{k+x_\ell\le y<k} \\
&+\left(1-(1-p)^{-k-x_\ell-1}\right)\mathbb{1}_{y=0,x_\ell<-k-1}
\end{aligned}$$

Given the above numerator and denominator, we have the privacy protection satisfies

$$e^{\epsilon} \le \max_{y\in Y}\left(\frac{f(y) + \frac{p((k)!)^2}{y!((k-y)!)^2}\sum_{i=\max(k-y,x_\ell)}^{\infty} g(i)}{f_1(y) + \frac{p(1-p)^{-1}((k)!)^2}{y!((k-y)!)^2}\sum_{i=x_0}^{\infty} g(i+1)}\right)$$

where $Y = \{0,1,\ldots,k\}$, $g(i) = \frac{(1-p)^{i-x_\ell}(i!)^2}{(i-k+y)!(k+i)!}$,
$f(y) = (1-p)^{y-k-x_\ell}\mathbb{1}_{k+x_\ell\le y<k} + \left(1-(1-p)^{-k-x_\ell}\right)\mathbb{1}_{y=0,x_\ell<-k}$,
$f_1(y) = \frac{p}{k+1}(-y+k+1)(1-p)^{y-k-1-x_\ell}\mathbb{1}_{k+1+x_\ell\le y\le k} + \frac{1+y}{k+1}p(1-p)^{y-k-x_\ell}\mathbb{1}_{k+x_\ell\le y<k} + \left(1-(1-p)^{-k-x_\ell-1}\right)\mathbb{1}_{y=0,x_\ell<-k-1}$ and
$x_0 = \max((k-y-1)_+, x_\ell)$.

A.3 Double Geometric Mechanism

Theorem 8. *Assume the server has k resources. Adding a random noise drawn from the double geometric distribution (with bias μ and scale s) to the number of requests satisfies ϵ-DP, where*

$$e^{\epsilon} \le \max_{y\in Y}\left(\frac{f(y) + \sum_{i=(k-y)_+}^{+\infty} e^{-\frac{1}{s}|i-\mu|}g(i)}{f_1(y) + \sum_{i=(k-y-1)_+}^{+\infty} e^{-\frac{1}{s}|i-\mu|}g(i+1)}\right)$$

where $Y = \{1,2,\ldots,k\}$, $g(i) = \frac{(k!)^2(i!)^2}{u!((k-y)!)^2(i-k+y)!(k+i)!}$,
$f(y) = e^{-\frac{1}{s}|y-k-\mu|}\mathbb{1}_{y\ne k} + \sum_{i=-\infty}^{-k-1} e^{-\frac{1}{s}|i-\mu|}\mathbb{1}_{y=0}$,
$f_1(y) = \frac{e^{-\frac{1}{s}|y-k-1-\mu|}(k-y+1)}{k+1} + \frac{e^{\frac{-1}{s}|y-k-\mu|}(y+1)}{k+1}\mathbb{1}_{y\ne k} + \sum_{i=-\infty}^{-k-2} e^{-\frac{1}{s}|i-\mu|}\mathbb{1}_{y=0}$, and s is the scale parameter in double geometric distribution.

Here $f(y)$ and $f_1(y)$ in Theorem 8 are from negative noise and the summations are from positive noise. When positive noise is being added, the probability of the attacker getting y allocation can be straightforwardly calculated by substituting $\Pr[d=i]$ in Equation 4 with the biased double geometric distribution. For negative noise, the attacker can only get $y < k$ with a probability of $e^{-\epsilon|y-k-\mu|}$ for the case of D. Whereas for the case of D' the attacker can still get k resources if noise equals to -1, and the victim is removed. Or else, the attacker will get $y < k$ resources in all other negative noise cases. This whole process is given by $f_1(y)$. Finally, the privacy bound in Theorem 8 is derived from the worst case y.

Given an allocator with k resources and an attacker sending $m = k$ requests, we assess the probability the attacker is allocated y resources.

$$\begin{aligned}
&\Pr\left[\mathsf{View}^{\mathcal{A}}_{\mathcal{M}}(D) = y\right] = \sum_{i=-\infty}^{+\infty}\Pr[d=i]\Pr[y \mid |D|+d] \\
=&\frac{1-e^{-\frac{1}{s}}}{1+e^{-\frac{1}{s}}}\left(e^{-\frac{1}{s}|y-k-\mu|}\mathbb{1}_{y\ne k} + \sum_{i=-\infty}^{-k-1} e^{-\frac{1}{s}|i-\mu|}\mathbb{1}_{y=0}\right. \\
&\left.+\sum_{i=(k-y)_+}^{+\infty} e^{-\frac{1}{s}|i-\mu|}\frac{\binom{m}{y}\binom{i}{k-y}}{\binom{m+i}{k}}\right) \\
=&\frac{1-e^{-\frac{1}{s}}}{1+e^{-\frac{1}{s}}}\left(\sum_{i=(k-y)_+}^{+\infty}\frac{k!k!i!i!e^{-\frac{1}{s}|i-\mu|}}{y!((k-y)!)^2(i-k+y)!(k+i)!}\right. \\
&\left.+e^{-\frac{1}{s}|y-k-\mu|}\mathbb{1}_{y\ne k} + \sum_{i=-\infty}^{-k-1} e^{-\frac{1}{s}|i-\mu|}\mathbb{1}_{y=0}\right)
\end{aligned}$$

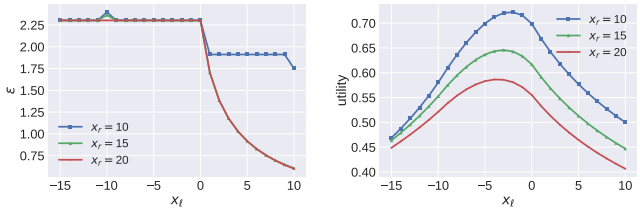
Similarly,

$$\begin{aligned}
&\Pr\left[\mathsf{View}^{\mathcal{A}}_{\mathcal{M}}(D') = y\right] = \sum_{i=-\infty}^{+\infty}\Pr[d=i]\Pr\left[y \mid \left|D'\right|+d\right] \\
=&\frac{1-e^{-\frac{1}{s}}}{1+e^{-\frac{1}{s}}}\left(\frac{e^{-\frac{1}{s}|y-k-1-\mu|}(k-y+1)}{k+1} + \sum_{i=-\infty}^{-k-2} e^{-\frac{1}{s}|i-\mu|}\mathbb{1}_{y=0}\right. \\
&+\frac{e^{-\frac{1}{s}|y-k-\mu|}(y+1)}{k+1}\mathbb{1}_{y\ne k} \\
&\left.+\sum_{i=(k-y-1)_+}^{+\infty}\frac{e^{-\frac{1}{s}|i-\mu|}(k!)^2((i+1)!)^2}{y!((k-y)!)^2(i+1-k+y)!(k+i+1)!}\right)
\end{aligned}$$

Given the above numerator and denominator, we have privacy protection as follows

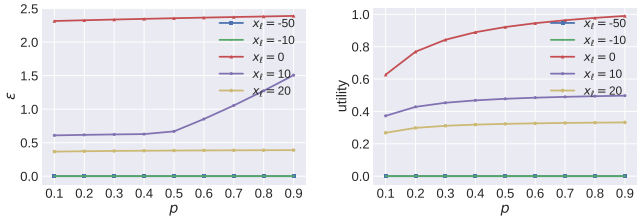
$$e^{\epsilon} \le \max_{y\in Y}\left(\frac{f(y) + \sum_{i=(k-y)_+}^{+\infty} e^{-\frac{1}{s}|i-\mu|}g(i)}{f_1(y) + \sum_{i=(k-y-1)_+}^{+\infty} e^{-\frac{1}{s}|i-\mu|}g(i+1)}\right)$$

where $Y = \{1,2,\ldots,k\}$, $g(i) = \frac{(k!)^2(i!)^2}{u!((k-y)!)^2(i-k+y)!(k+i)!}$,
$f(y) = e^{-\frac{1}{s}|y-k-\mu|}\mathbb{1}_{y\ne k} + \sum_{i=-\infty}^{-k-1} e^{-\frac{1}{s}|i-\mu|}\mathbb{1}_{y=0}$,
$f_1(y) = \frac{e^{-\frac{1}{s}|y-k-1-\mu|}(k-y+1)}{k+1} + \frac{e^{-\frac{1}{s}|y-k-\mu|}(y+1)}{k+1}\mathbb{1}_{y\ne k} + \sum_{i=-\infty}^{-k-2} e^{-\frac{1}{s}|i-\mu|}\mathbb{1}_{y=0}$, and s is the scale parameter.



(a) ϵ of UNI given different x_ℓ and x_r. **(b) Utility of UNI given different x_ℓ and x_r.**

Figure 5: Impact of x_ℓ and x_r on UNI.



(a) ϵ of GEO given different p and x_ℓ. **(b) Utility of GEO given different p and x_ℓ.**

Figure 6: Impact of p and x_ℓ on GEO.

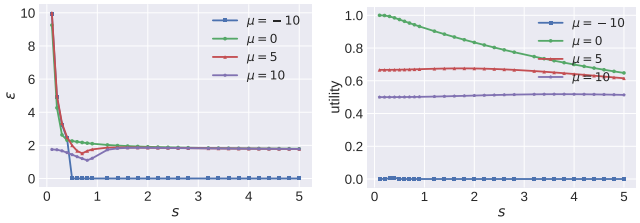
B IMPACT OF PARAMETERS

Starting Point x_ℓ and End Point x_r of UNI. In Figure 5, we display privacy and utility across various x_r ($x_r = 10, 15, 20$) and x_ℓ values (along the x-axis). Notably, $x_r = 15$ largely mirrors $x_r = 20$ in terms of ϵ, even though $x_r = 20$ is expected to offer superior privacy. Regarding utility, $x_r = 10$ consistently ranks highest for different x_ℓ, followed by $x_r = 15$ and $x_r = 20$. Regarding x_ℓ, increasing its value enhances privacy (resulting in a lower ϵ), with utility peaking when x_ℓ ranges between $[-5, 0]$. However, we observe two outliers related to x_ℓ in Figure 5a. First, a peak is observed when $x_\ell = -10$, because all requests in D are removed deterministically but the probability of the same situation for D' is $\frac{1}{k+1}$, where victim exists. Second, when $x_\ell = x_r = 10$, ϵ drops to 1.75 because this special case implies that the attacker gets no resource in the victim's absence.

Geometric Parameter p and Starting Point x_ℓ of GEO. Figure 6 depicts how p and x_ℓ affect GEO. For $x_\ell = -50$ and $x_\ell = -10$, both ϵ and utility approach 0 due to the high likelihood of request removal. At $x_\ell = 0$, utility is high but ϵ consistently exceeds 2. For $x_\ell = 10, 20$, ϵ is below 1.5, with utility rising as x_ℓ increases. For p, its influence on ϵ is minimal, except at $x_\ell = 10$ where ϵ increases sharply after $p = 0.5$. Utility consistently grows with p across all settings.

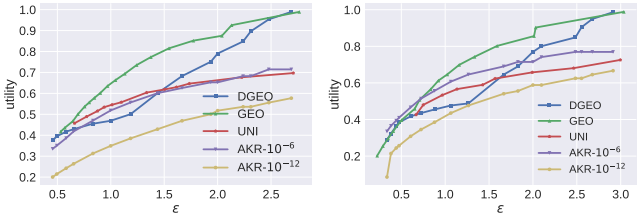
Geometric Scale s and Bias μ of DGEO. In DGEO, the scale parameter s determines the noise's decay rate. A smaller s results in noise more closely concentrated around the bias μ. μ introduces more noise to the allocation, impacting post-allocation privacy. We evaluate the influence of these parameters on privacy and utility, presenting the findings in Figure 7. Introducing bias μ improves privacy, especially when $s < 1$. For larger s, the distribution resembles a discrete uniform, keeping ϵ stable (around 2 for $\mu \geq 0$). s has limited utility impact unless $\mu = 0$.

Resource Capacity k. We set k to 10 for the prior experiments like Angel et al. [3]. Here we test our mechanisms and AKR on



(a) ϵ of DGEO given different scale s and bias μ. **(b) Utility of DGEO given different scale s and bias μ.**

Figure 7: Impact of s and μ on DGEO.



(a) Comparison of mechanisms under k=15. **(b) Comparison of mechanisms under k=20.**

Figure 8: Privacy protection and utility under $k = 15, 20$. The ranges for the x-axis differ for k because not all utility values can be derived under every ϵ.

$k = 15, 20$. Figure 8 shows the privacy-utility tradeoff. For AKR, besides the default $\delta = 10^{-6}$, we also evaluate $\delta = 10^{-12}$, bringing its privacy closer to ϵ-DP. Figure 8 illustrates that δ significantly impacts AKR's utility, with average gaps of 0.2 for $k = 15$ and 0.1 for $k = 20$. GEO and DGEO still perform well for these new k values and better than AKR.