Privacy-Preserving Analytics: The Browser nightmares

Hamed Haddadi

@realhamed

MSN 2019

https://haddadi.github.io/

Imperial College London

Disclaimers

- The views presented here are personal and do not present those of any browser vendor

- The architecture discussions, where not published, are presented solely for the purpose of discussions within the attendees.

Crash Reports vs. Telemetry

The modern browser is like a dynamic operating system.

An evolving jungle of content, style, third party code, analytics, advertising, tracking, personalisation, extensions, permissions, etc...

Bug/crash reports are not enough for working out when things go wrong

Regular "telemetry" reports need to be collected from a variety of the above signals.

However telemetry and tracking/fingerprinting go hand in hand....

Chrome, attempt 1 (CCS'14)

RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response

Úlfar Erlingsson
Google, Inc.
ulfar@google.com

Vasyl Pihur
Google, Inc.
vpihur@google.com

Aleksandra Korolova
University of Southern California
korolova@usc.edu

ABSTRACT

Randomized Aggregatable Privacy-Preserving Ordinal Response, or RAPPOR, is a technology for crowdsourcing statistics from end-user client software, anonymously, with strong privacy guarantees. In short, RAPPORs allow the forest of client data to be studied, without permitting the possibility of looking at individual trees. By applying randomized response in a novel manner, RAPPOR provides the mechanisms for such collection as well as for efficient, high-utility analysis of the collected data. In particular, RAPPOR permits statistics to be collected on the population of client-side strings with strong privacy guarantees for each client, and without linkability of their reports.

This paper describes and motivates RAPPOR, details its differential-privacy and utility guarantees, discusses its practical deployment and properties in the face of different attack models, and, finally, gives results of its application to both synthetic and real-world data.

asked to flip a fair coin, in secret, and answer "Yes" if it comes up heads, but tell the truth otherwise (if the coin comes up tails). Using this procedure, each respondent retains very strong deniability for any "Yes" answers, since such answers are most likely attributable to the coin coming up heads; as a refinement, respondents can also choose the untruthful answer by flipping another coin in secret, and get strong deniability for both "Yes" and "No" answers.

Surveys relying on randomized response enable easy computations of accurate population statistics while preserving the privacy of the individuals. Assuming absolute compliance with the randomization protocol (an assumption that may not hold for human subjects, and can even be non-trivial for algorithmic implementations [23]), it is easy to see that in a case where both "Yes" and "No" answers can be denied (flipping two fair coins), the true number of "Yes" answers can be accurately estimated by $2(Y - 0.25)$, where Y is the proportion of "Yes" responses. In expectation, respondents will provide the true answer 75% of the time, as

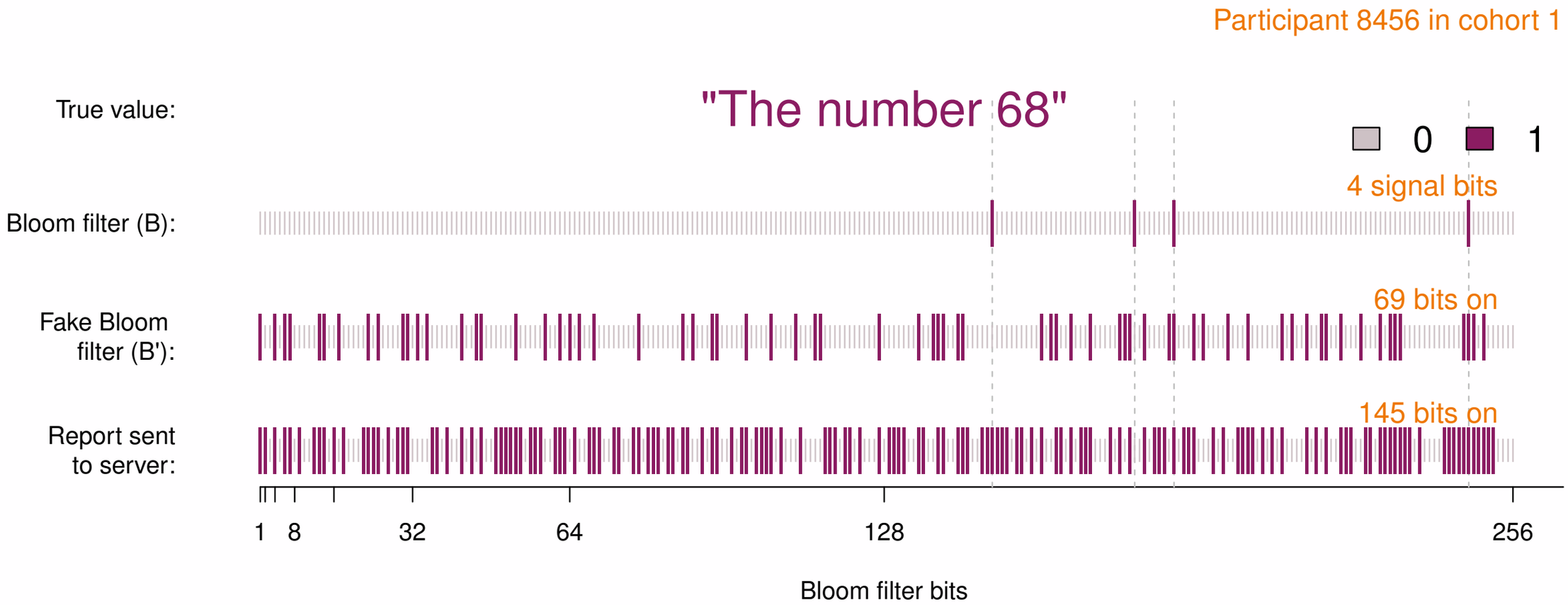


Figure 1: Life of a RAPPOR report: The client value of the string "The number 68" is hashed onto the Bloom filter B using h (here 4) hash functions. For this string, a Permanent randomized response B' is produces and memoized by the client, and this B' is used (and reused in the future) to generate Instantaneous randomized responses S (the bottom row), which are sent to the collecting service.

- RAPPOR gracefully handles multiple data collections from the same client by providing well-defined longitudinal differential privacy guarantees.
- Highly tunable parameters allow to balance risk versus utility over time, depending on one's needs and assessment of likelihood of different attack models.
- RAPPOR is purely a client-based privacy solution. It eliminates the need for a trusted third- party server and puts control over client's data back into their own hands.

Imperial College London

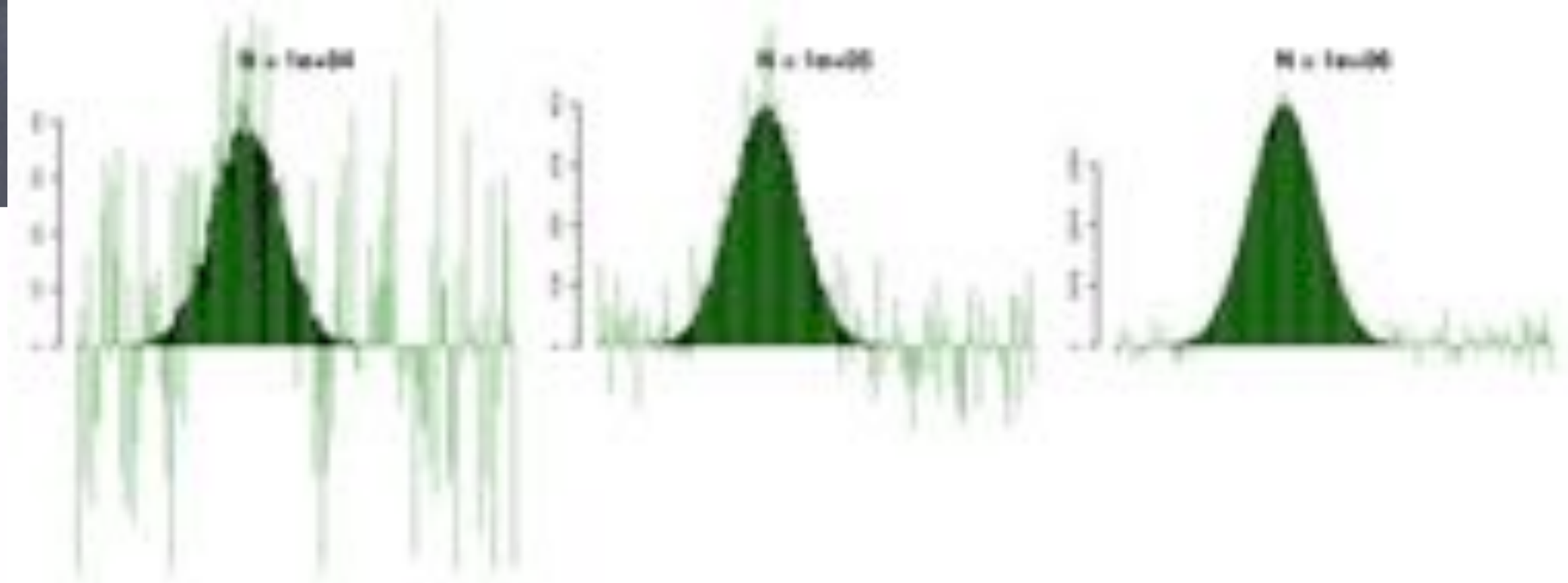


Figure 4: Simulations of learning the normal distribution with mean 50 and standard deviation 10. The RAPPOR privacy parameters are $q = 0.75$ and $p = 0.5$, corresponding to $\epsilon = \ln(3)$. True sample distribution is shown in black; light green shows the estimated distribution based on the decoded RAPPOR reports. We do not assume a priori knowledge of the Normal distribution in learning. If such prior information were available, we could significantly improve upon learning the shape of the distribution via smoothing.

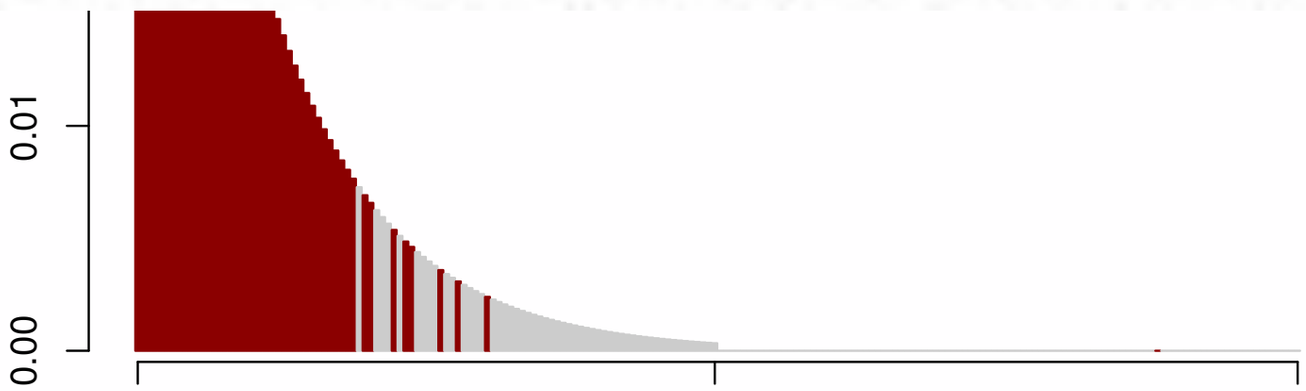


Figure 5: Population of strings with their true frequencies on the vertical axis (0.01 is 1%). Strings detected by RAPPOR are shown in dark red.

Randomized Response

Imperial College London

So, what's the issue?

- **Too good? Utility can go down?** "…. there are strict limits to the utility of locally-differentially-private analyses. Because each reporting individual performs independent coin flips, any analysis results are perturbed by noise induced by the properties of the binomial distribution."

- **Niche/unique cases will be ignored?** "… best suited for measuring the most frequent elements in data from peaky power-law distributions.."

- **Hard to navigate through changes:** ".. opaque, fixed, and statistical nature of the data collected. Not only does this prevent exploratory data analysis and any form of manual vetting, but it also ren- ders the reported data incompatible with the existing tools and processes of standard engineering practice. "

Chrome, attempt 2 (SOSP'17)

PROCHLO: Strong Privacy for Analytics in the Crowd

Andrea Bittau* Úlfar Erlingsson* Petros Maniatis* Ilya Mironov* Ananth Raghunathan*
David Lie‡ Mitch Rudominer° Ushasree Kode° Julien Tinnes° Bernhard Seefeld°

*Google Brain ‡Google Brain and U. Toronto °Google

Abstract

The large-scale monitoring of computer users' software activities has become commonplace, e.g., for application telemetry, error reporting, or demographic profiling. This paper describes a principled systems architecture—*Encode, Shuffle, Analyze* (ESA)—for performing such monitoring with high utility while also protecting user privacy. The ESA design, and its PROCHLO implementation, are informed by our practical experiences with an existing, large deployment of privacy-preserving software monitoring.

With ESA, the privacy of monitored users' data is guaranteed by its processing in a three-step pipeline. First, the data is *encoded* to control scope, granularity, and randomness. Second, the encoded data is collected in batches subject to a randomized threshold, and blindly *shuffled*, to break linkability and to ensure that individual data items get "lost in the crowd" of the batch. Third, the anonymous, shuffled data is *analyzed* by a specific analysis engine that further prevents statistical inference attacks on analysis results.

ESA extends existing best-practice methods for sensitive-data analytics, by using cryptography and statistical techniques to make explicit how data is elided and reduced in precision, how only common-enough, anonymous data is analyzed, and how this is done for only specific, permitted purposes. As a result, ESA remains compatible with the established workflows of traditional database analysis.

Strong privacy guarantees, including differential privacy, can be established at each processing step to defend against malice or compromise at one or more of those steps. PROCHLO develops new techniques to harden those steps, including the *Stash Shuffle*, a novel scalable and efficient oblivious-shuffling algorithm based on Intel's SGX, and new applications of cryptographic secret sharing and blinding. We describe ESA and PROCHLO, as well as experiments that validate their ability to balance utility and privacy.

1. Introduction

Online monitoring of client software behavior has long been used for disparate purposes, such as measuring feature adoption or performance characteristics, as well as large-scale error-reporting [34]. For modern software, such monitoring may entail systematic collection of information about client devices, their users, and the software they run [17, 60, 69]. This data collection is in many ways fundamental to modern software operations and economics, and provides many clear benefits, e.g., it enables the deployment of security updates that eliminate software vulnerabilities [62].

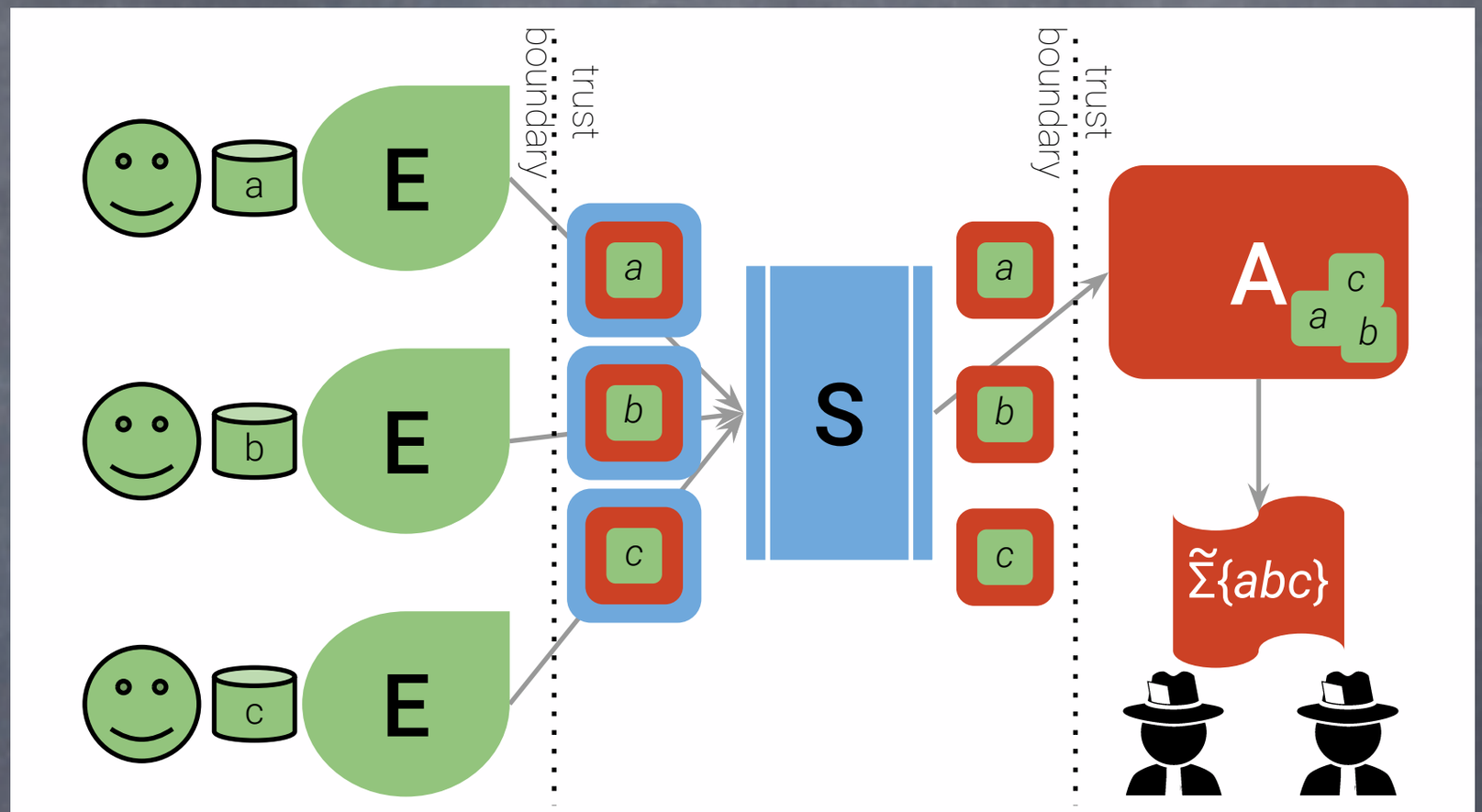
For such data, the processes, mechanisms, and other means of privacy protection are an increasingly high-profile concern. This is especially true when data is collected automatically and when it is utilized for building user profiles or demographics [21, 69, 71]. Regrettably, in practice, those concerns often remain unaddressed, sometimes despite the existence of strong incentives that would suggest otherwise. One reason for this is that techniques that can guarantee privacy exist mostly as theory, as limited-scope deployments, or as innovative-but-nascent mechanisms [5, 7, 25, 28].

We introduce the *Encode, Shuffle, Analyze* (ESA) architecture for privacy-preserving software monitoring, and its PROCHLO implementation.[1] The ESA architecture is informed by our experience building, operating, and maintaining the RAPPOR privacy-preserving monitoring system for the Chrome Web browser [28]. Over the last 3 years, RAPPOR has processed up to billions of daily, randomized reports in a manner that guarantees local differential privacy, without assumptions about users' trust; similar techniques have since gained increased attention [6, 7, 70, 74]. However, these techniques have limited utility, both in theory and in our experience, and their statistical nature makes them ill-suited to standard software engineering practice.

Our ESA architecture overcomes the limitations of systems like RAPPOR, by extending and strengthening current

Imperial College London

Encode, Shuffle, Analyze



ESA architecture: Encode, shuffle, and analyze.

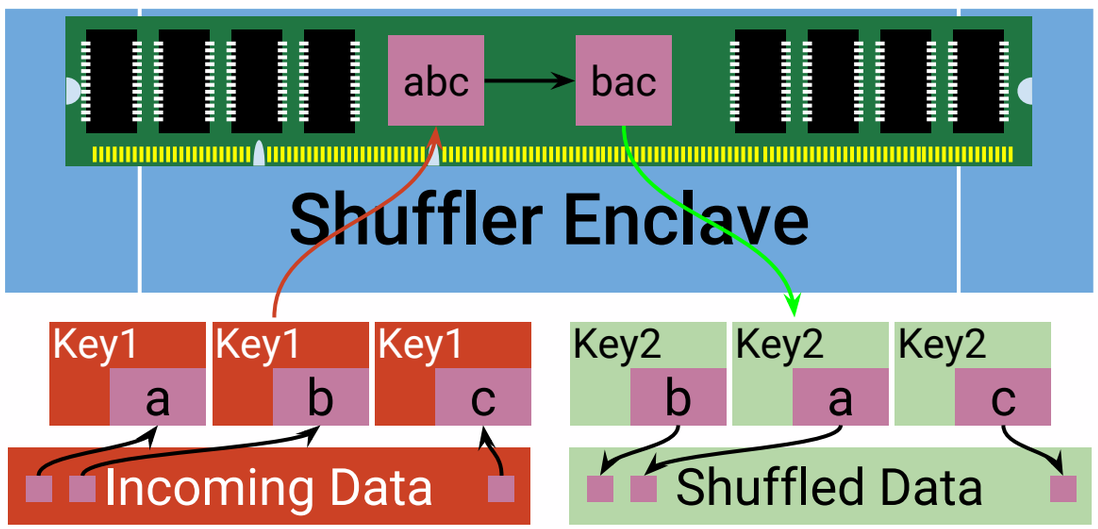


Figure 2: Primitive operation of an oblivious shuffler.

Imperial College London

Results are more promising

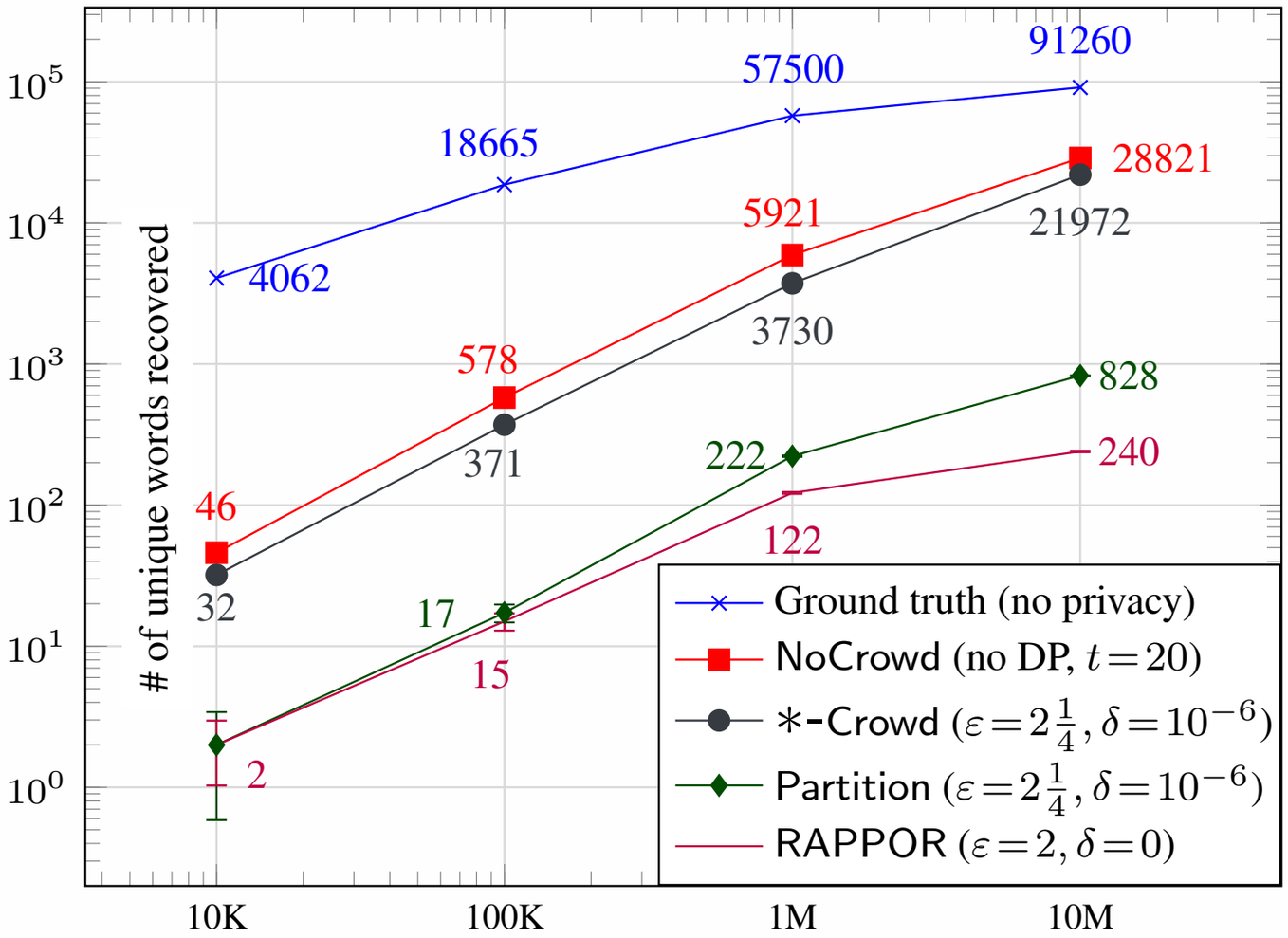


Figure 5: A log-log-graph of the number of unique words recovered (Y-axis) on samples of 10 thousand to 10 million Vocab words (X-axis). Using results in The $*$-Crowd line results from using word hashes as the crowd-IDs, whereas NoCrowd offers less privacy, using a naïve threshold of 20 and no crowds. For comparison, RAPPOR and Partition show how pure local-differential privacy offers far less accuracy and much higher variance (error bars) even when augmented with partitions as described in §2.2.

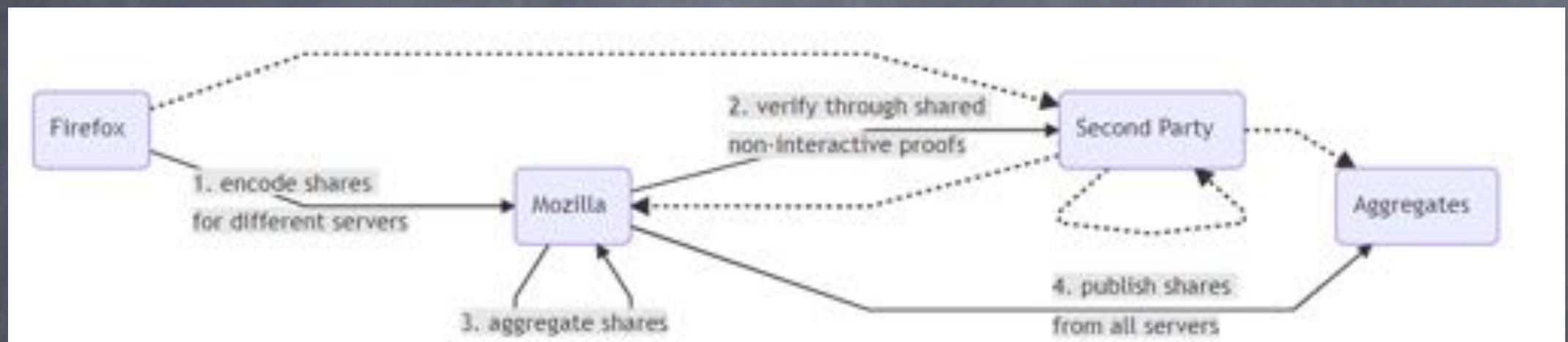
# Movies	# Users	# Reports	Score (RMSE)	
			no privacy	PROCHLO
200	90K	1.77M	0.9579	0.9595†
2K	353K	335M	0.9414	0.9420
18K	480K	22.6B	0.9222	0.9242

Table 5: Utility of the Flix evaluation; lower numbers are better. (†To account for sparsity, the threshold was set to 5.)

Imperial College London

Mozilla tries Prio (NSDI 2017)

Pretty similar to PrivAd, Adnostic, etc



Each Prio client holds a private data value (e.g., its current location), and a small set of servers compute statistical functions over the values of all clients (e.g., the most popular location).

https://crypto.stanford.edu/prio/

What next?

- Clear dilemma between privacy and utility
- How do we trust the client? (hint: TEE)
- How do we trust the server? (hint: TEE)
- How do we trust the third party? (timing, collusion, encryption)
- How much data do we really need?
- Click fraud is the biggest threat here? Or is it Ad blocking?
- How do monetary incentives (BAT?) nudge the system?

References

- RAPPOR: CCS'14
- PROCHLO: SOSP '17
- Prio: NSDI'17,
- https://hacks.mozilla.org/2018/10/testing-privacy-preserving-telemetry-with-prio/

For more information, software, and papers:

https://brave.com/research/