

Lecture 5: Differential Privacy (cont'd)

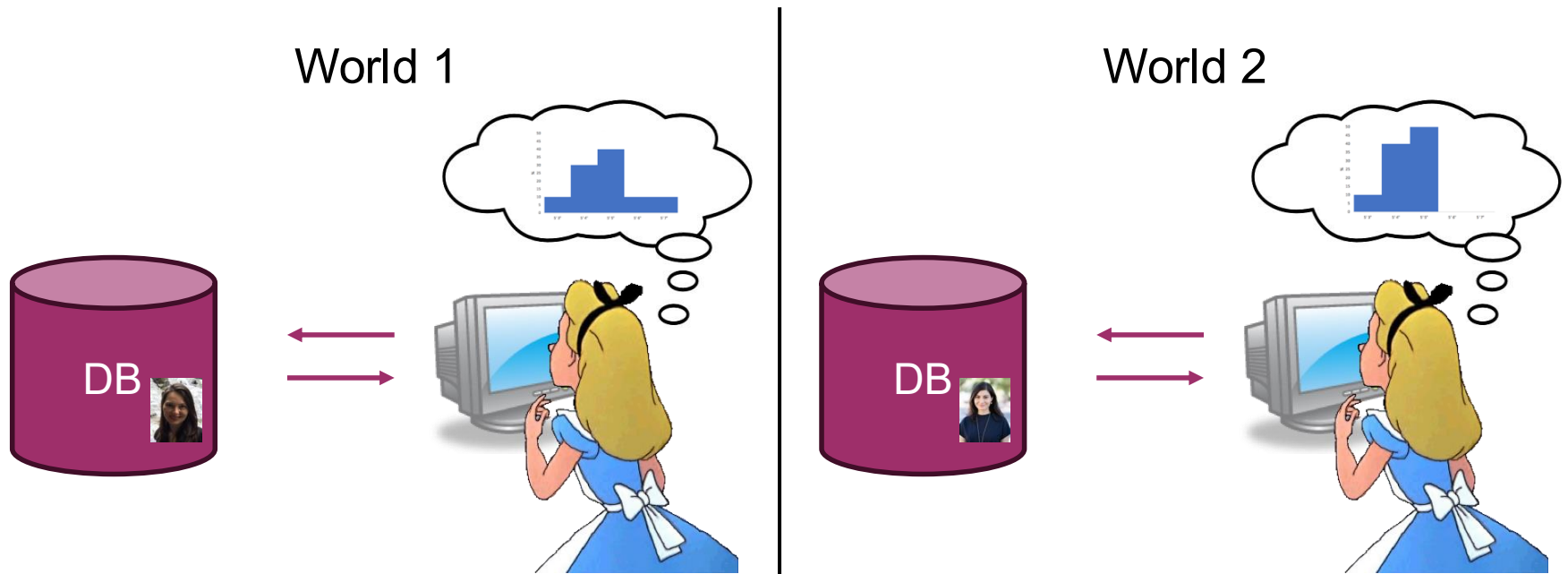
CS 181E

Fall 2026

Review: Anonymization is hard

- Personal data showing up in unexpected places
- Errors suppressing data
- **Thinking only about personal data / PII is not sufficient**
- It's hard to reason about what is/is not identifiable
- Availability of **auxiliary information**
 - Extremely hard to control

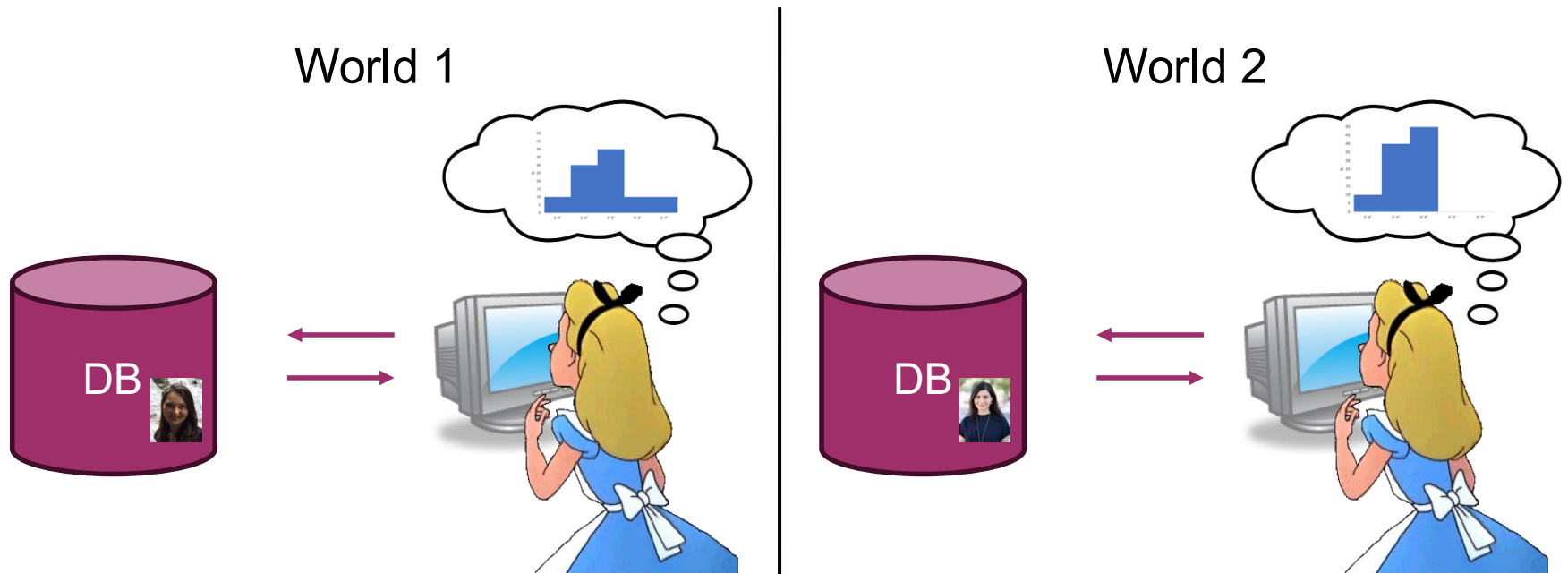
Review: Differential Privacy



A query Q is differentially private if the answer you get in the world with my data is included is pretty much the same as the answer you would get otherwise



Review: Differential Privacy

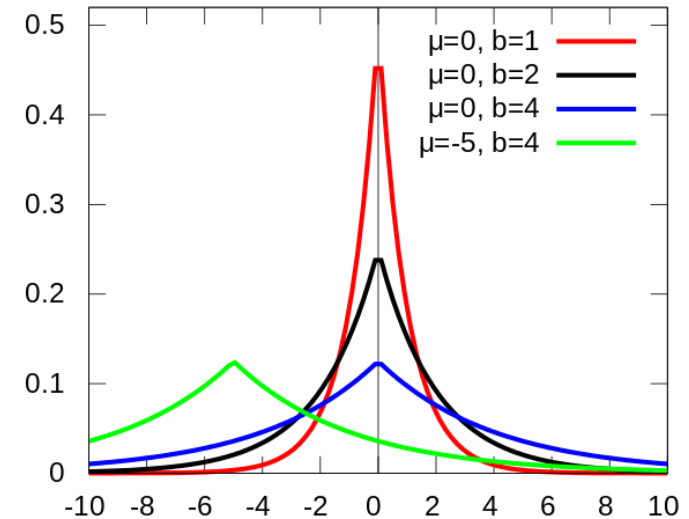


A query Q is ϵ -differentially private if $\forall x \in \text{Range}(Q), \forall D, r \in D,$
 $\Pr[Q(D + r) = x] \leq e^\epsilon \cdot \Pr[Q(D + r') = x]$



Review: Laplacian Mechanism

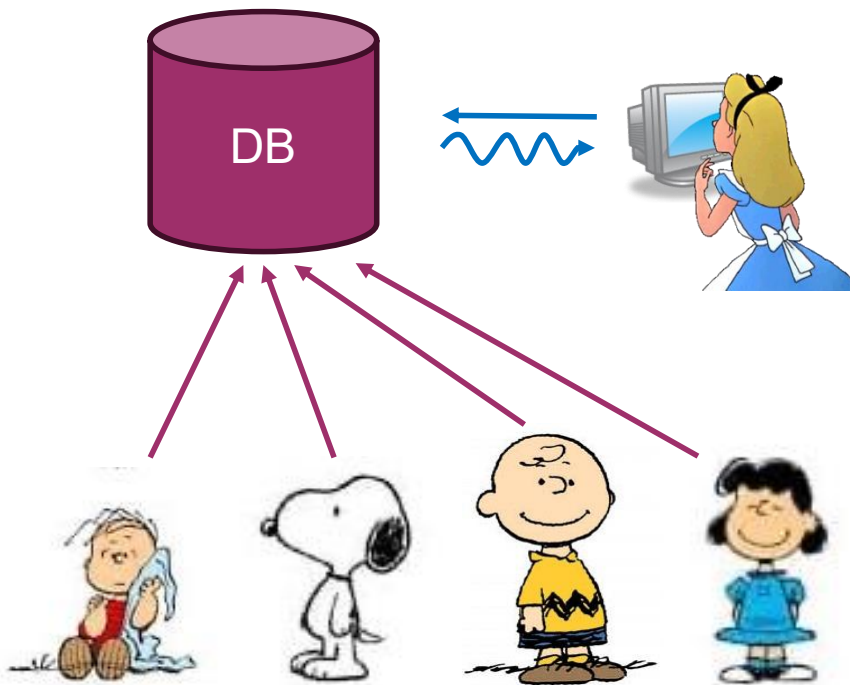
- Given a query Q on a database D that has sensitivity Δ , respond with $f(D) = Q(D) + Y$ where Y is drawn from the distribution $Lap(\frac{\Delta}{\epsilon})$



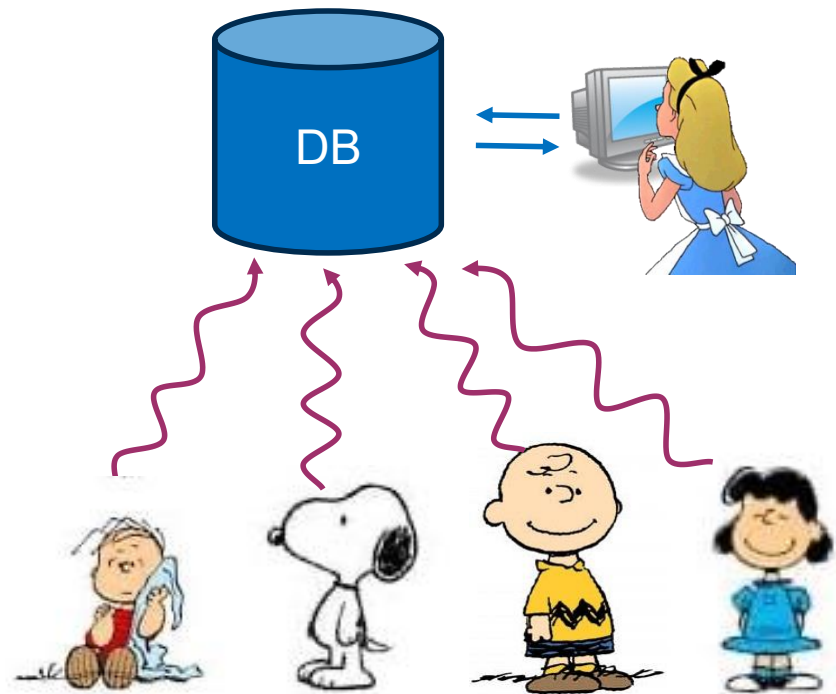
- Theorem:** this mechanism satisfies ϵ -differential privacy
- Limitations:
 - composition/managing privacy budget
 - Have to trust whoever is storing the database
 - Whoever stores database has to be online for analysis

Models of Differential Privacy

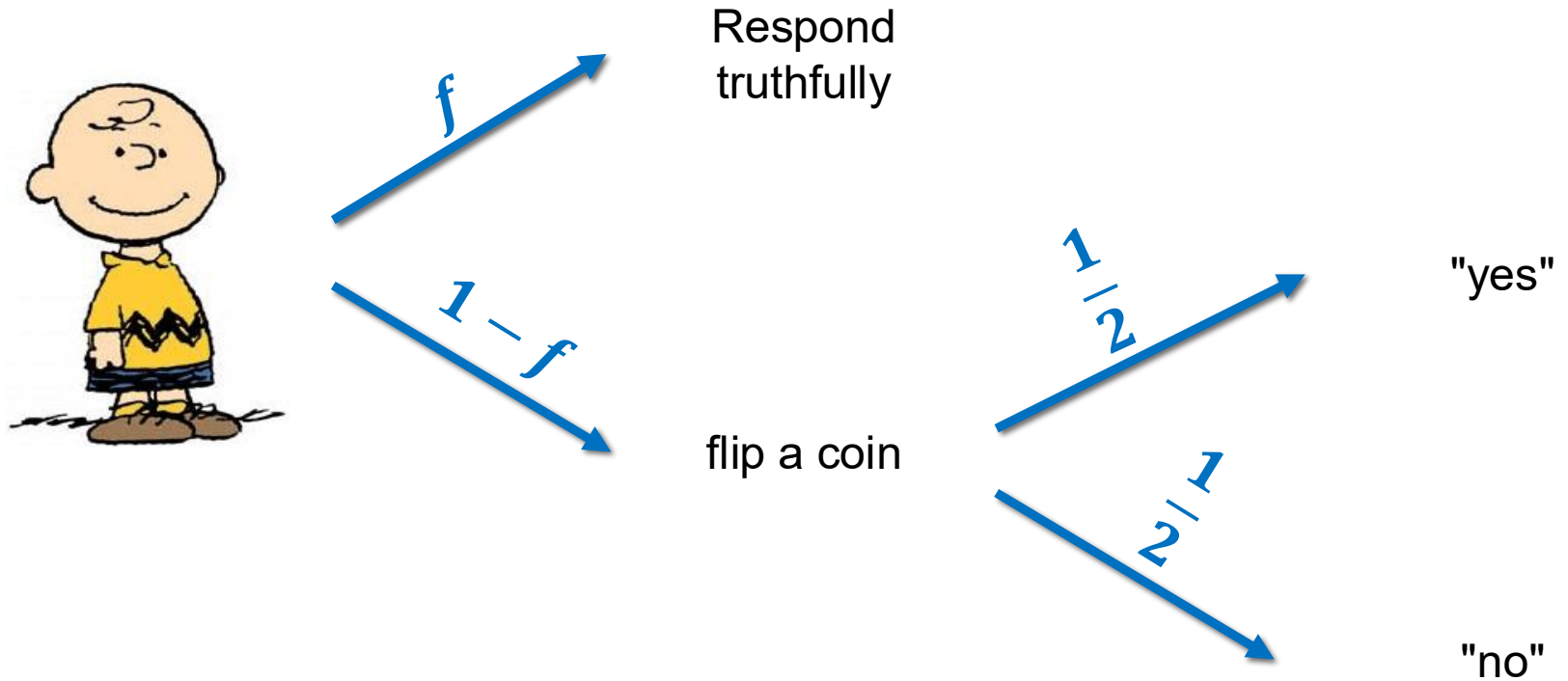
Central DP



Local DP



Randomized Response



$\langle Y, N, \textcolor{blue}{Y}, Y, Y, N, \dots, Y, N \rangle$

Randomized Response

- **Theorem:** If $p = \frac{1}{2}$ then this mechanism satisfies $\ln(3)$ -differential privacy

$$\begin{aligned}
 & \frac{\Pr[\langle Y, N, \mathbf{Y}, Y, Y, N, \dots, Y, N \rangle \mid f(\text{Charlie}) = Y]}{\Pr[\langle Y, N, \mathbf{Y}, Y, Y, N, \dots, Y, N \rangle \mid f(\text{Charlie}) = N]} \\
 &= \frac{\Pr[Y \mid f(P_0)] \cdot \Pr[N \mid f(P_1)] \cdot \Pr[Y \mid f(\text{Charlie}) = Y] \cdot \dots \cdot \Pr[N \mid f(P_{n-1})]}{\Pr[Y \mid f(P_0)] \cdot \Pr[N \mid f(P_1)] \cdot \Pr[Y \mid f(\text{Charlie}) = N] \cdot \dots \cdot \Pr[N \mid f(P_{n-1})]} \\
 &= \frac{\Pr[Y \mid f(\text{Charlie}) = Y]}{\Pr[Y \mid f(\text{Charlie}) = N]} \\
 &= \frac{p \cdot 1 + (1 - p) \cdot \frac{1}{2}}{p \cdot 0 + (1 - p) \cdot \frac{1}{2}} = \frac{(1 + p) \cdot \frac{1}{2}}{(1 - p) \cdot \frac{1}{2}} = \frac{(1 + p)}{(1 - p)} \\
 &= \frac{\left(1 + \frac{1}{2}\right)}{\left(1 - \frac{1}{2}\right)} = \frac{\left(3 \cdot \frac{1}{2}\right)}{\left(\frac{1}{2}\right)} = 3 \\
 &= e^{\ln(3)} \approx e^{1.1} \quad \Rightarrow \text{ln(3)-differentially private}
 \end{aligned}$$

Exercise: Randomized Response

- Assume that we've deployed this randomized response technique (with $p = \frac{1}{2}$) to collect data about how many people have used AI to do their homework. When we look at the dataset, we see that 32% of the responses were "yes". What should we estimate is the true rate of AI use?

Limitations

- Limitation #1: This only works for binary responses

Handling Ordinal Data

- “Hot-one encoding” – store data as an array of k bits, each bit encodes whether one possible attribute applies

Human	Black Hair	Blond Hair	Brown Hair	Female
-------	------------	------------	------------	--------

$B_{Lucy} =$

1	1	0	0	1
---	---	---	---	---



$B_{Charlie} =$

1	0	0	0	0
---	---	---	---	---



$B_{Snoopy} =$

0	0	0	0	0
---	---	---	---	---



Handling Arbitrary Bit-strings

- just use the string
 - not private: length can leak info
- hash the string
 - risk of collisions when deployed at scale
- use some other fancy data structure
 - Bloom filters!

Bloom Filters

- A bloom filter is an array of k bits that can store one or more data values
- A bloom filter has n associated hash functions all with the range $[0, k-1]$
- Initially, all bits are zero (empty filter)
- To add a value x to the filter, compute $y_1 = h_1(x), \dots, y_n = h_n(x)$ and set bits at indices $y_1, \dots, y_n$ to 1
- To add another value x' , compute the n hashes and set bits to 1

0	1	1	0	1
---	---	---	---	---

- To check if a value is in the filter, compute the n hashes. If all those bits are 1, assume value is in filter. Otherwise not.

Example: Bloom Filters

- Assume $k=16$, $n=2$,

- $h_1(x) = x \% 16$
- $h_2(x) = (x/16) \% 16$

- Insert 88:

- $h_1(88) = 8$
- $h_2(88) = 5$

- Insert 106:

- $h_1(106) = 10$
- $h_2(106) = 6$

- Check for 88:

- $h_1(88) = 8$ 
- $h_2(88) = 5$ 



- Check for 100:

- $h_1(100) = 4$ 
- $h_2(100) = 6$ 



- Check for 104:

- $h_1(104) = 8$ 
- $h_2(104) = 6$ 



0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	1	1	0	1	0	1	0	0	0	0	0

No false negatives, but can get false positives!

Exercise: Bloom Filters

- Assume $k=16$, $n=2$,

- $h_1(x) = x \% 16$
- $h_2(x) = (x/16)\% 16$

- You have the following Bloom filter:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0	0	0	1	0	1	1	0	0	0	0	1	1	0	0

- Which of the following values are stored in this filter?
 - 121
 - 124
 - 64
 - 47

Adding Differential Privacy

- Step 1: encode values as fixed-length binary array B
 - hot-one encoding
 - Bloom filter

- Step 2: apply randomized response bitwise

- for each bit b_i create a modified bit b'_i

$$b'_i = \begin{cases} 1, & \text{with probability } \frac{1}{2}f \\ 0, & \text{with probability } \frac{1}{2}f \\ b_i, & \text{with probability } 1 - f \end{cases}$$

- Note: if a signal bit is replaced with 0, B' will no longer contain information about the value in B

Limitations

- ~~Limitation #1: This only works for binary responses~~
- Limitation #2: Repeated answers degrade privacy

Memoization

- Permanently associated a fixed random response with the real value B
- If you needed again, used the stored B'
- Never generate a fresh random B' for the same value
- **Permanent randomized response**

Limitations

- ~~Limitation #1: This only works for binary responses~~
- ~~Limitation #2: Repeated answers degrade privacy~~
- Limitation #3: Memoized repeated responses are linkable

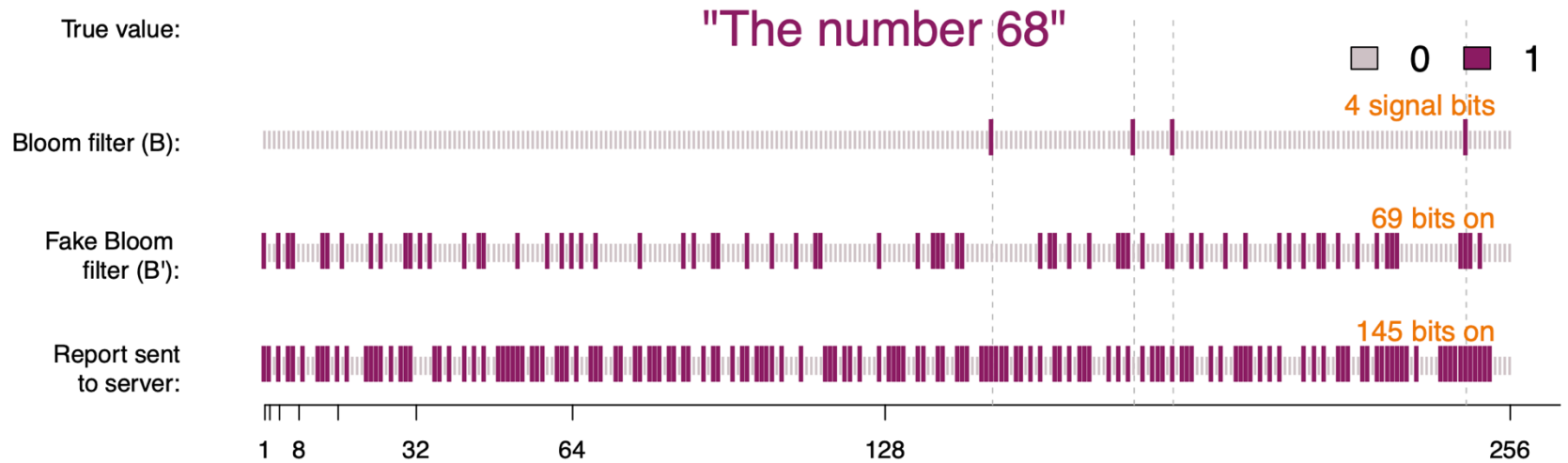
Add a second layer of Privacy

- **Instantaneous randomized response**
- Create new array using bitwise randomized response

$$\Pr[S_i = 1] = \begin{cases} q, & \text{if } B'_i = 1 \\ p, & \text{if } B'_i = 0 \end{cases}$$

- Send this new array S to the server

RAPPOR



$$\begin{aligned} n &= 4 \\ f &= .5 \\ q &= .75 \\ p &= .5 \end{aligned}$$



Privacy of RAPPOR

- $q^* = \Pr[S_i = 1 | b_i = 1]$
 $= \Pr[S_i = 1 \text{ and } b'_i = 1 | b_i = 1] + \Pr[S_i = 0 \text{ and } b'_i = 0 | b_i = 1]$
 $= \Pr[S_i = 1 | b'_i = 1 \text{ and } b_i = 1] \cdot \Pr[b'_i = 1 | b_i = 1]$
 $\quad + \Pr[S_i = 1 | b'_i = 0 \text{ and } b_i = 1] \cdot \Pr[b'_i = 0 | b_i = 1]$
 $= q \cdot (\frac{1}{2}f + (1 - f)) + p \cdot (\frac{1}{2}f)$
 $= \frac{1}{2}f(p + q) + (1 - f)q$
- $p^* = \Pr[S_i = 1 | b_i = 0] = \frac{1}{2}f(p + q) + (1 - f)p$

Privacy of RAPPOR

- $q^* = \Pr[S_i = 1 | b_i = 1] = \frac{1}{2}f(p + q) + (1 - f)q$
- $p^* = \Pr[S_i = 1 | b_i = 0] = \frac{1}{2}f(p + q) + (1 - f)p$

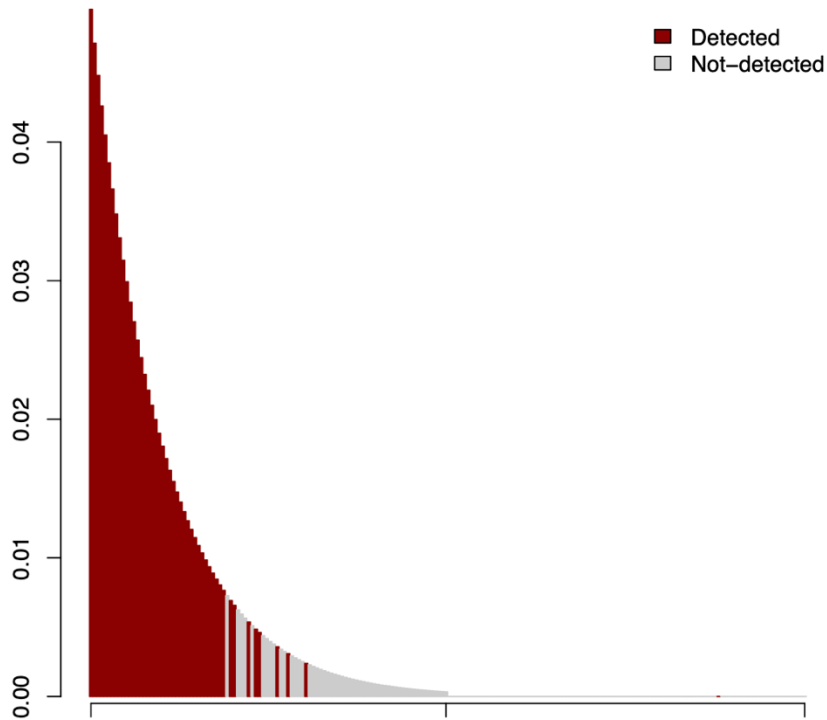
$$\begin{aligned}\frac{\Pr[S \mid f(\text{Charlie}) = B_1]}{\Pr[S \mid f(\text{Charlie}) = B_2]} &\leq \frac{(q^*)^n \cdot (1 - p^*)^n}{(p^*)^n \cdot (1 - q^*)^n} \\ &= \left(\frac{q^*(1 - p^*)}{p^*(1 - q^*)} \right)^n \\ &= e^{n \ln \left(\frac{q^*(1 - p^*)}{p^*(1 - q^*)} \right)}\end{aligned}$$

- RAPPOR is ϵ_1 -differentially private for $\epsilon_1 = n \ln \left(\frac{q^*(1 - p^*)}{p^*(1 - q^*)} \right)$
- Backup if repeated queries negate instantaneous DP is still ϵ_2 -differentially private for $\epsilon_2 = 2n \ln \left(\frac{1 - \frac{1}{2}f}{\frac{1}{2}f} \right)$

Utility of RAPPOR

- Calculate estimate of number of times each bit is truly set in B
- Use regression model to select candidate strings to consider
- Use least-squares regression to estimate counts for candidate strings
- filter out ones with insignificant p-values

Utility of RAPPOR

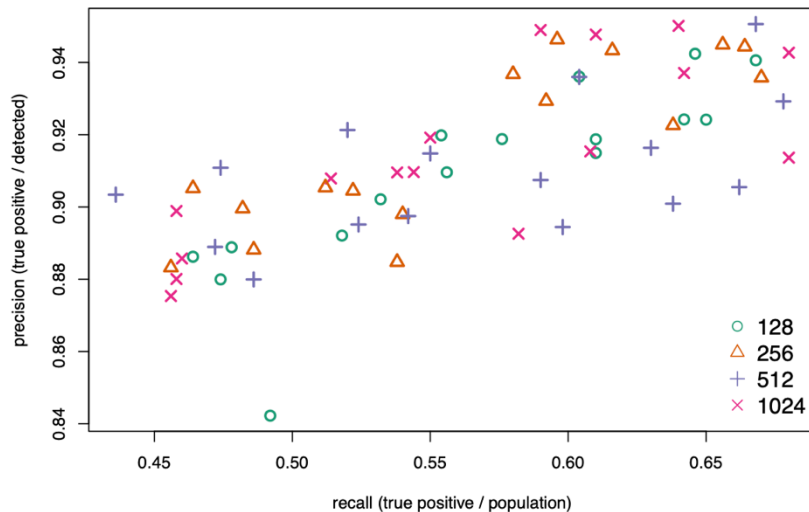


String	Est	Truth
V_1	48803	49884
V_2	47388	47026
V_3	39509	44642
V_4	40420	42747
V_5	41490	40077
V_10	34196	31234
V_19	23860	20057
V_20	20159	19110

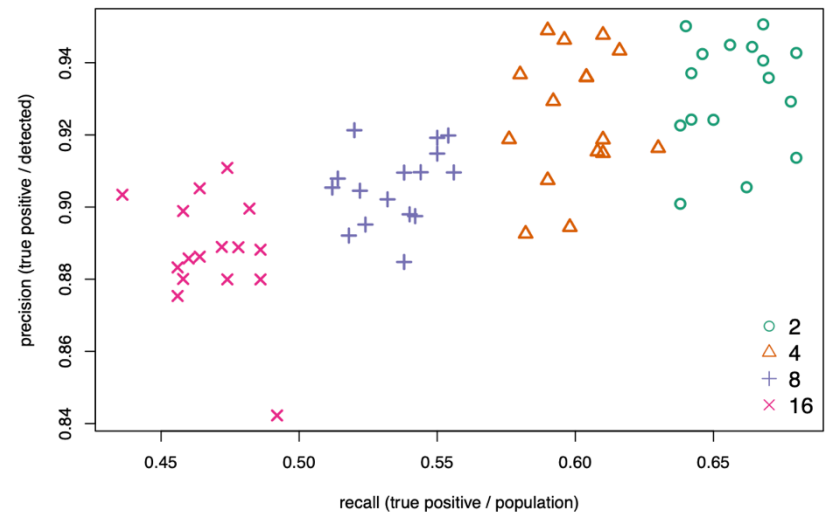
Experiments run with $k=128, n=2, f=.5, q=.75, p=.5$

Choosing Parameters

Bloom Filter Size



Number of Hash functions (n)



f, p, q should be chosen for desired value of ϵ
Experiments run with $\epsilon = \ln(3)$

Estimating Malware Prevalence

- 186,792 reports from 10,133 Windows computers
- sampled actively running processes (~18 process names per machine)
- Goal: estimate frequency of malware “BADAPPLE.COM”

Process Name	Est.	Stdev	P.value	Prop.
RASERVER.EXE	8054	1212	1.56E-11	0.04
RUNDLL32.EXE	7488	1212	3.32E-10	0.04
CONHOST.EXE	7451	1212	4.02E-10	0.04
SPPSVC.EXE	6363	1212	7.74E-08	0.03
AITAGENT.EXE	5579	1212	2.11E-06	0.03
MSIEXEC.EXE	5147	1212	1.10E-05	0.03
SILVERLIGHT.EXE	4915	1212	2.53E-05	0.03
BADAPPLE.COM	4860	1212	3.07E-05	0.03
LPREMOVE.EXE	4787	1212	3.95E-05	0.03
DEFRAG.EXE	4760	1212	4.34E-05	0.03

RAPPOR in the real world

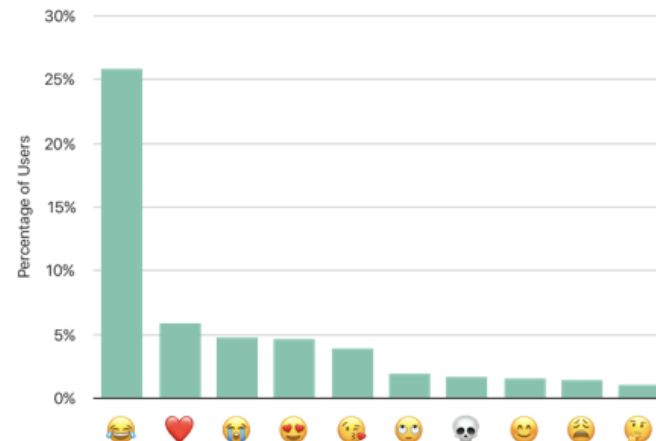


- used in Chromium for error reporting and other data collection for ~10 years
- more recently, mostly replaced by private ML

Local DP on iPhones



- Apple uses a form of local differential privacy known as Private Count Mean Sketch to collect data iOS users
- identifying websites with high memory or energy usage
- discovering new words
- identifying popular emojis



Local DP on Windows Devices



- first deployed in some versions of Windows 10
- collects time user spends on a particular app (6 hour intervals)

Census Data

- U.S. federal government use differential privacy to protect data released about the 2020 census
- cool application of differential privacy at scale
- lots of controversy about this at the policy level

