# From Federated Learning to Federated Analytics

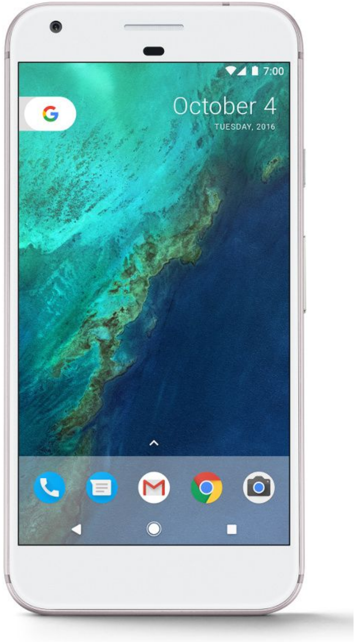

Peter Kairouz
kairouz@

Building on the work of many

# Data is born at the edge

Billions of phones & IoT devices constantly generate data

Data enables better products and smarter models

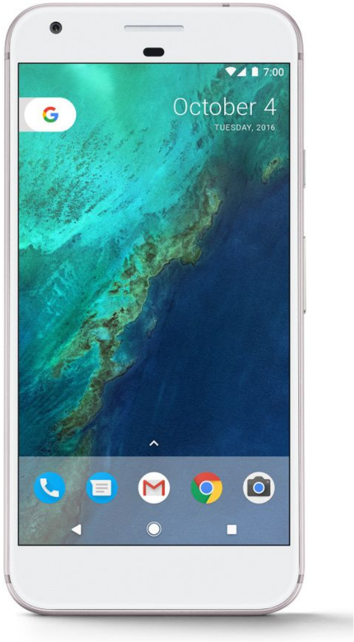



# Can data live at the edge?

Data processing is moving on device:

- Improved latency
- Works offline
- Better battery life
- Privacy advantages

E.g., on-device inference for mobile keyboards and cameras.

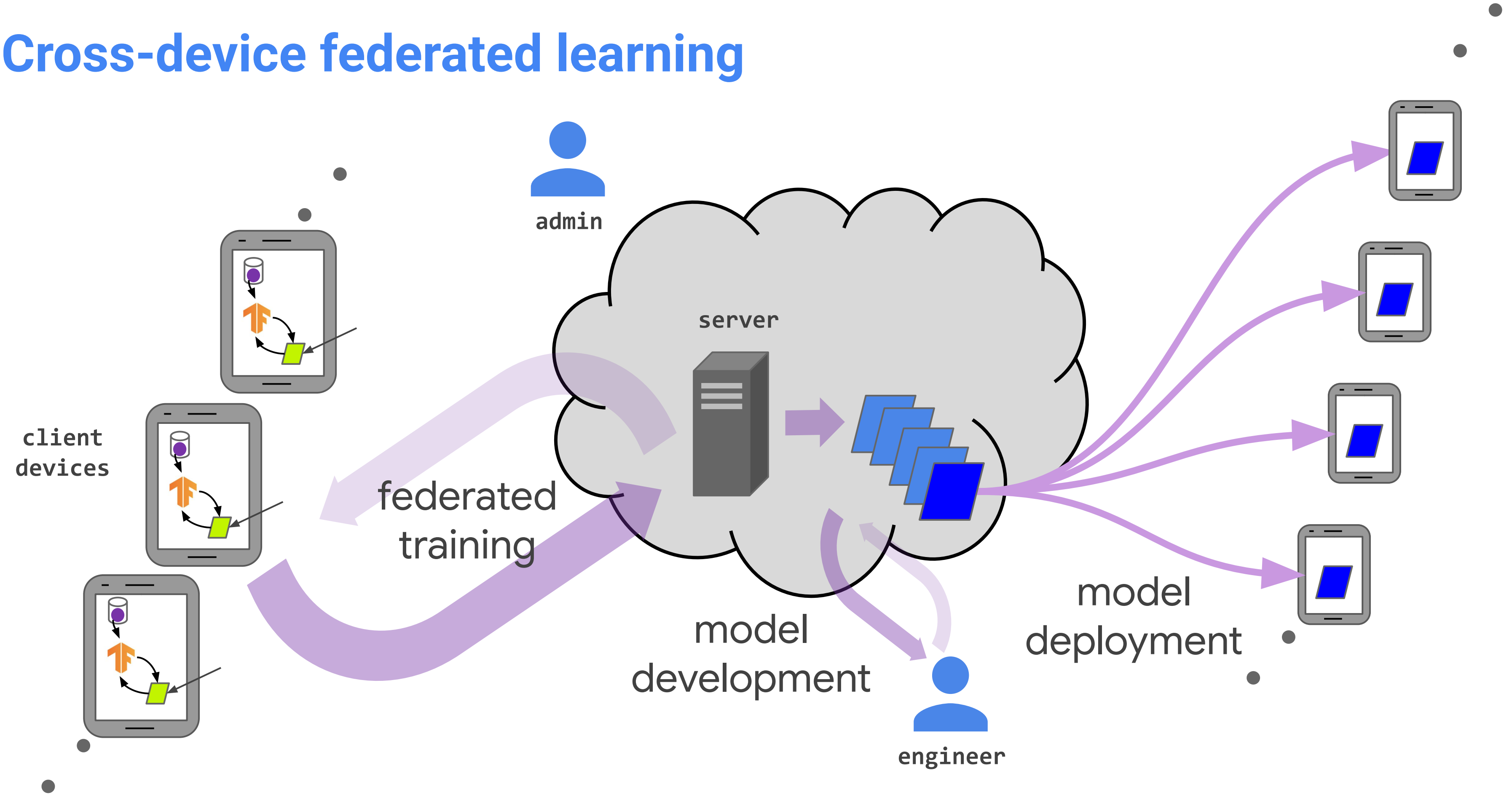
# What is federated learning?

# Federated learning

**Federated learning** is a machine learning setting where multiple entities (clients) collaborate in solving a machine learning problem, under the coordination of a central server or service provider. Each client's raw data is stored locally and not exchanged or transferred; instead, focused updates intended for immediate aggregation are used to achieve the learning objective.

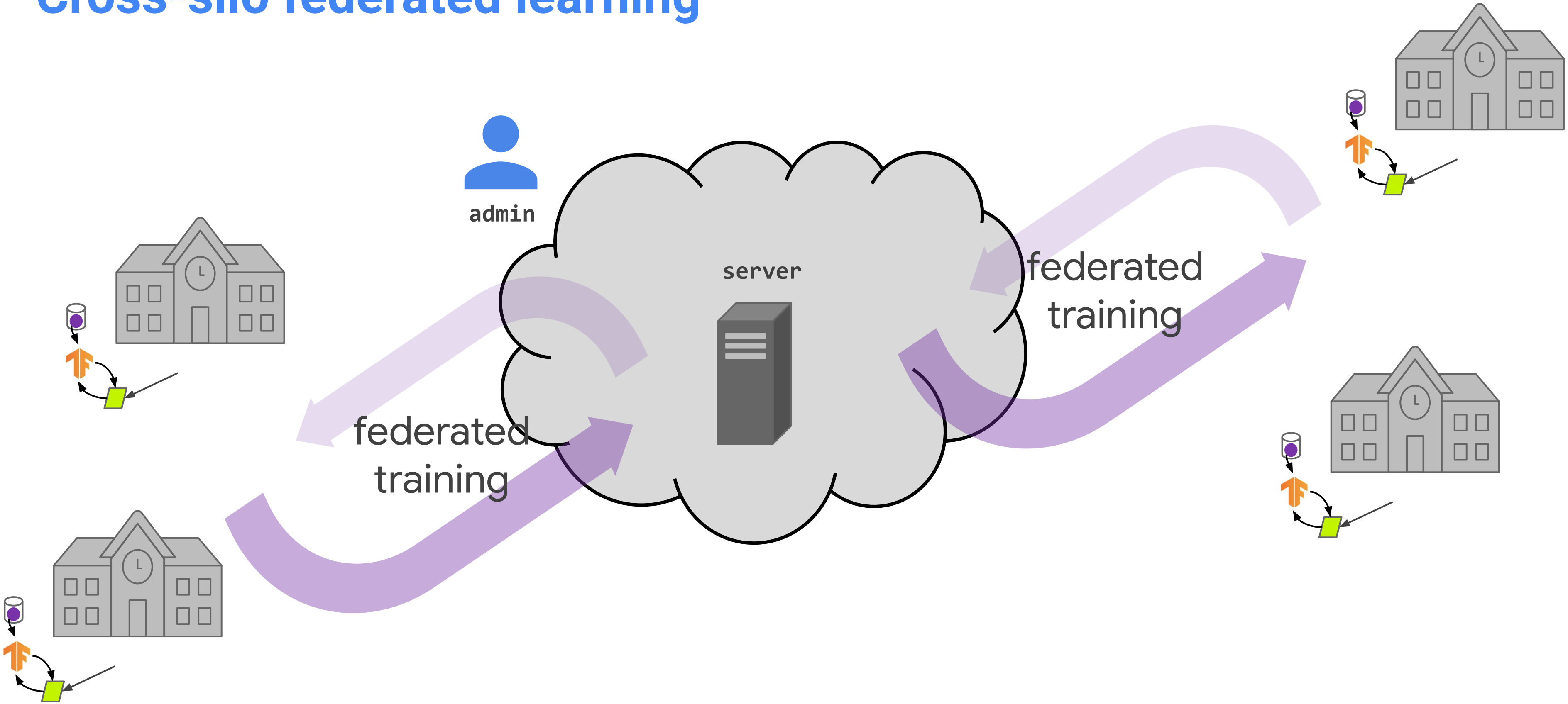
working definition proposed in
*Advances and Open Problems in Federated Learning* (arxiv/1912.04977)

# Cross-device federated learning




Google

# Cross-silo federated learning




Google

# Characteristics of the federated learning setting

| | Datacenter distributed learning | Cross-silo federated learning | Cross-device federated learning |
|---|---|---|---|
| **Setting** | Training a model on a large but "flat" dataset. Clients are compute nodes in a single cluster or datacenter. | Training a model on siloed data. Clients are different organizations (e.g., medical or financial) or datacenters in different geographical regions. | The clients are a very large number of mobile or IoT devices. |
| **Data distribution** | Data is centrally stored, so it can be shuffled and balanced across clients. Any client can read any part of the dataset. | **Data is generated locally and remains decentralized.** Each client stores its own data and cannot read the data of other clients. Data is not independently or identically distributed. | |
| **Orchestration** | Centrally orchestrated. | A central orchestration server/service organizes the training, but never sees raw data. | |
| **Wide-area communication** | None (fully connected clients in one datacenter/cluster). | Hub-and-spoke topology, with the hub representing a coordinating service provider (typically without data) and the spokes connecting to clients. | |
| **Data availability** | All clients are almost always available. | | Only a fraction of clients are available at any one time, often with diurnal and other variations. |
| **Distribution scale** | Typically 1 - 1000 clients. | Typically 2 - 100 clients. | Massively parallel, up to $10^{10}$ clients. |

Google

# Characteristics of the federated learning setting

| | Datacenter distributed learning | Cross-silo federated learning | Cross-device federated learning |
|---|---|---|---|
| **Addressability** | Each client has an identity or name that allows the system to access it specifically. | | Clients cannot be indexed directly (i.e., no use of client identifiers) |
| **Client statefulness** | Stateful --- each client may participate in each round of the computation, carrying state from round to round. | | Generally stateless --- each client will likely participate only once in a task, so generally we assume a fresh sample of never before seen clients in each round of computation. |
| **Primary bottleneck** | Computation is more often the bottleneck in the datacenter, where very fast networks can be assumed. | Might be computation or communication. | Communication is often the primary bottleneck, though it depends on the task. Generally, federated computations uses wi-fi or slower connections. |
| **Reliability of clients** | Relatively few failures. | | Highly unreliable --- 5% or more of the clients participating in a round of computation are expected to fail or drop out (e.g., because the device becomes ineligible when battery, network, or idleness requirements for training/computation are violated). |
| **Data partition axis** | Data can be partitioned / re-partitioned arbitrarily across clients. | Partition is fixed. Could be example-partitioned (horizontal) or feature-partitioned (vertical). | Fixed partitioning by example (horizontal). |

# Federated learning vs fully decentralized learning

| | Federated learning | Fully decentralized (peer-to-peer) learning |
|---|---|---|
| **Orchestration** | A central orchestration server/service organizes the training, but never sees raw data. | No centralized orchestration. |
| **Wide-area communication pattern** | Hub-and-spoke topology, with the hub representing a coordinating service provider (typically without data) and the spokes connecting to clients. | Peer-to-peer topology. |

# Federated beyond learning

Google

# Beyond learning: federated analytics

**Federated analytics is** the practice of applying data science methods to the analysis of raw data that is stored locally on users' devices. Like federated learning, it works by running local computations over each device's data, and only making the aggregated results — and never any data from a particular device — available to product engineers. Unlike federated learning, however, federated analytics aims to support basic data science needs.

*definition proposed in*
https://ai.googleblog.com/2020/05/federated-analytics-collaborative-data.html

# Federated analytics

- Federated histograms over closed sets
- Federated heavy hitters discovery over open sets
- Federated density of vector spaces
- Federated selection of random data subsets
- Federated SQL?
- etc...

# Federated heavy hitters (frequent item) discovery

= “The **moon** is full, the sky full of stars.”

= “The full **moon** is two days before Halloween this month.”

= “You see the **moon** instead of how dark the night is.”

We are going to focus on words and assume each device has a single word - both assumptions can be relaxed

# The TrieHH Algorithm

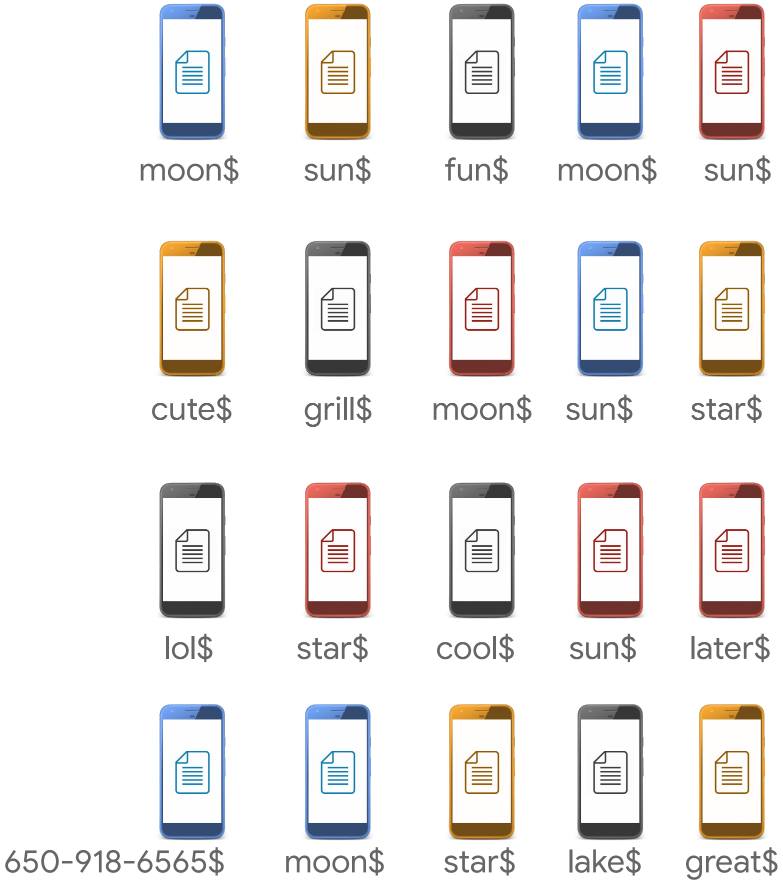
# Algorithm via example

n = 20, each user has a single word

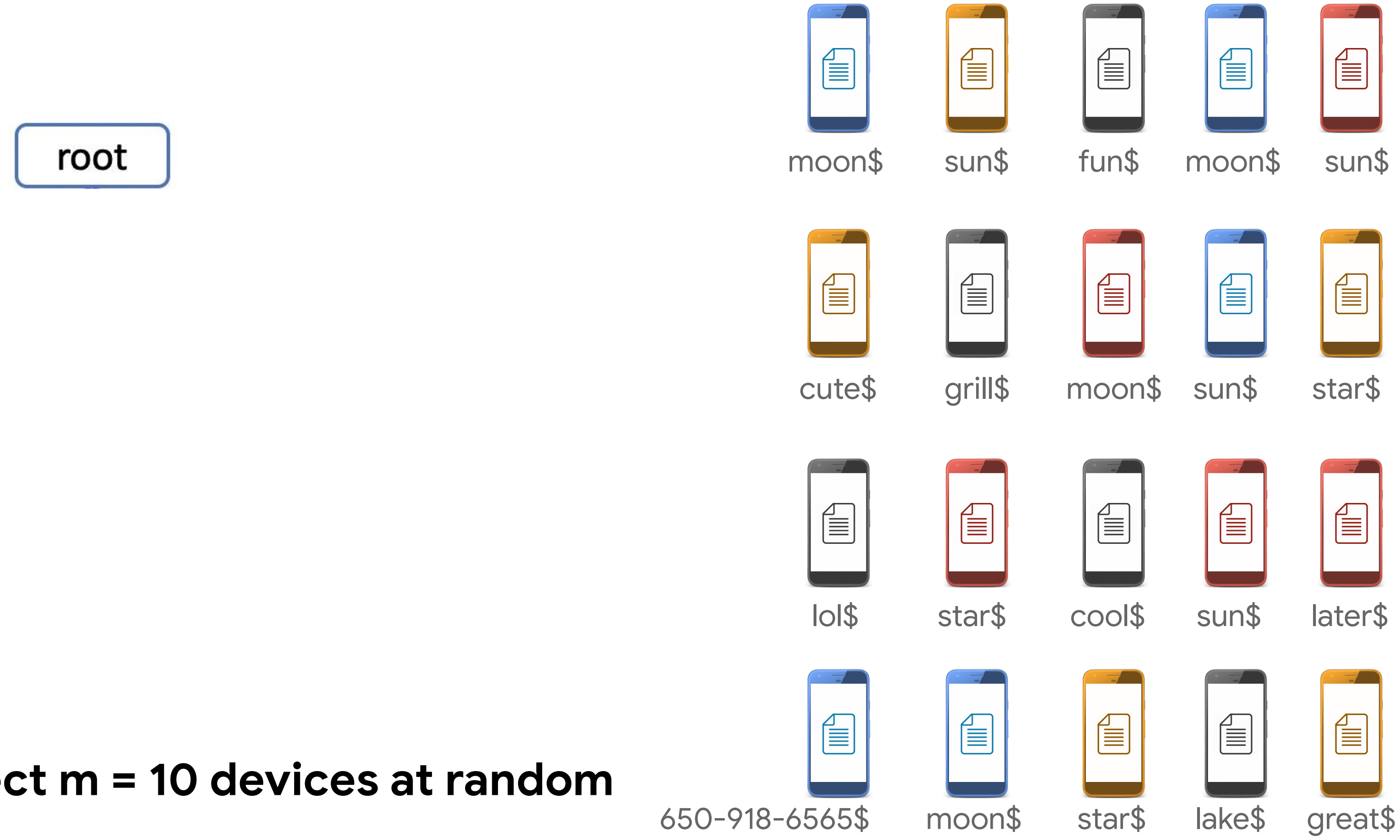
“moon” and “sun” appear 4 times

“star” appears 3 times

“$” denotes end of word

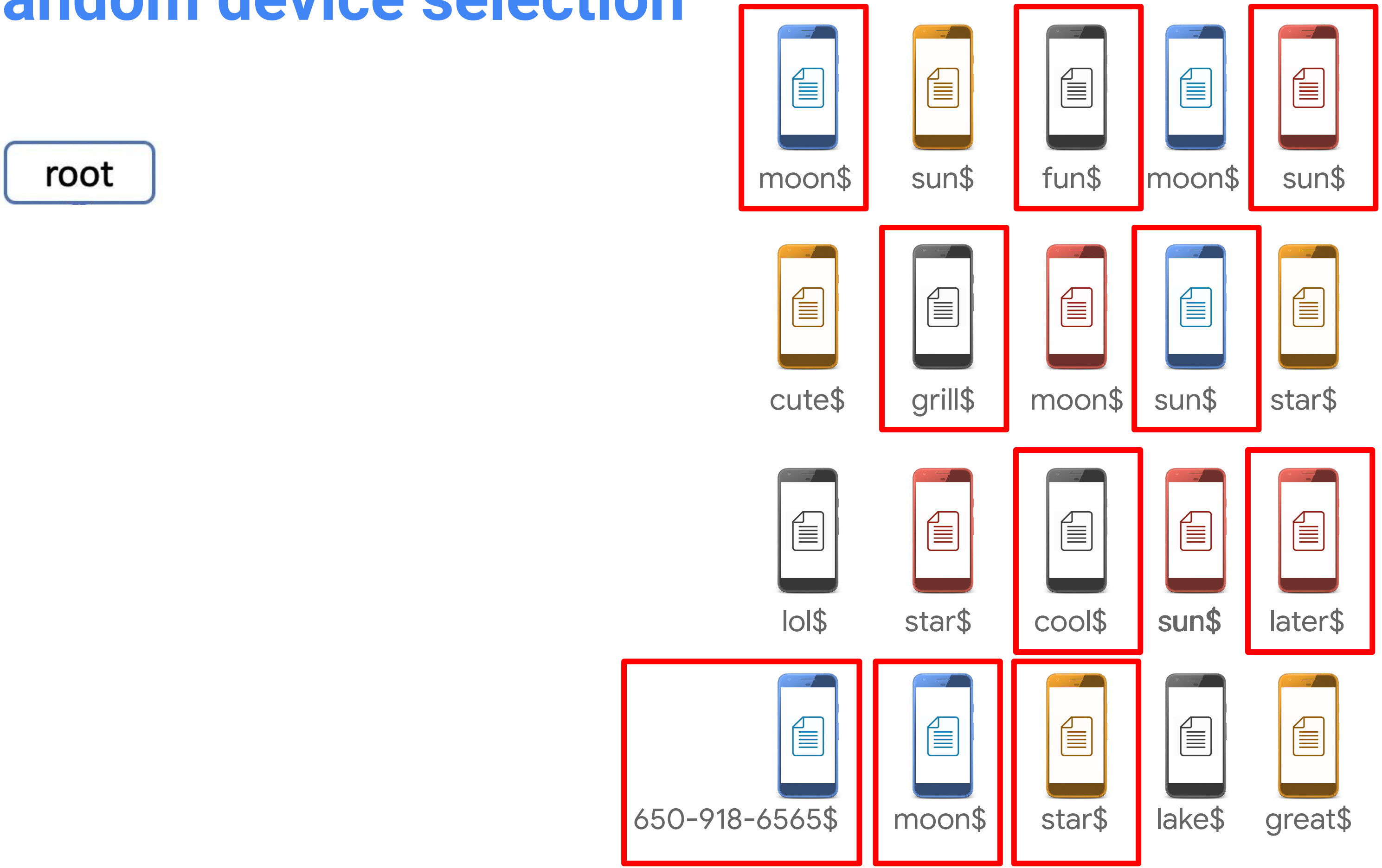


# Round 1: random device selection




**Randomly select m = 10 devices at random**

# Round 1: random device selection



# Round 1: voting stage

root

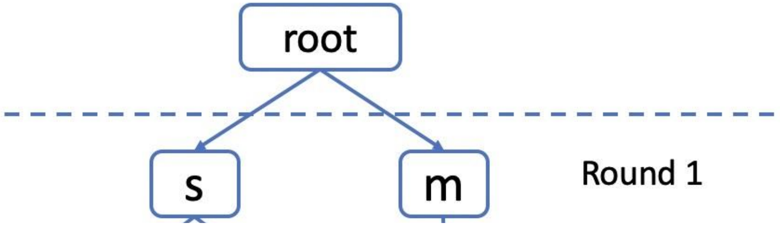

‘s’: 3 votes

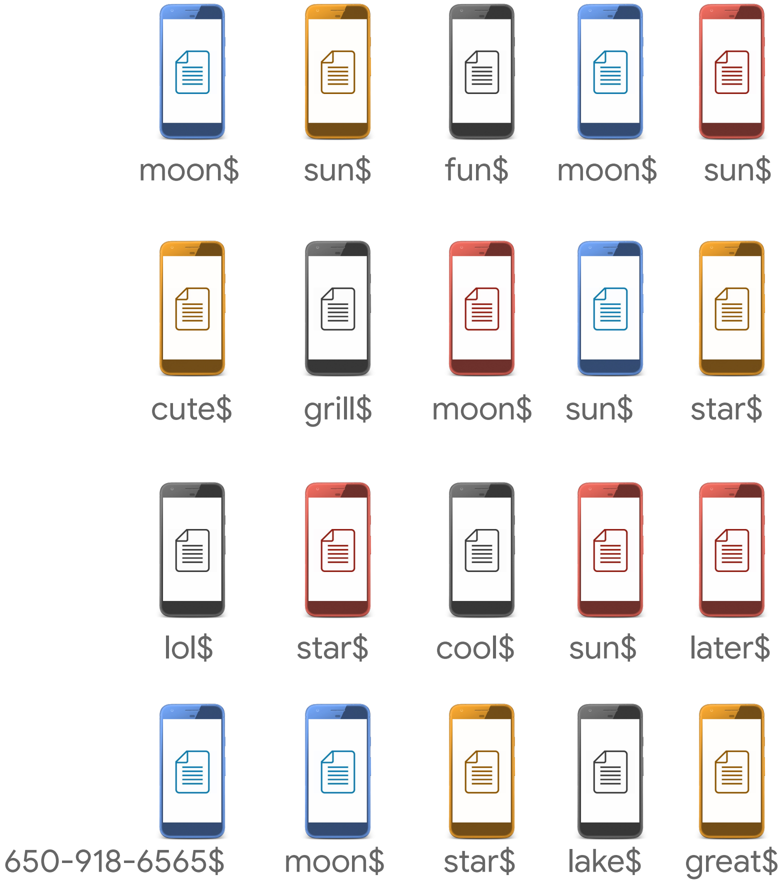
‘m’: 2 votes

‘f’, ‘g’, ‘c’, ‘l’, ‘6’: 1 vote each

**Each device votes on one character**

# Round 1: vote thresholding stage



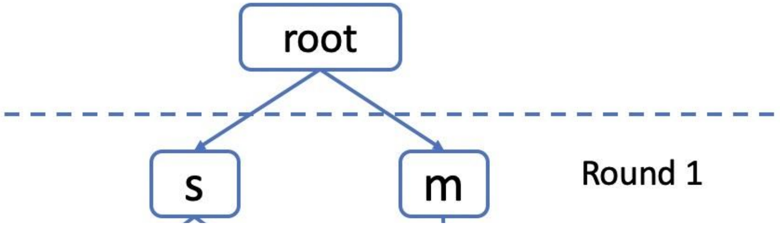

‘s’: 3 votes

‘m’: 2 votes

‘f’, ‘g’, ‘c’, ‘l’, ‘6’: 1 vote each

**Chars with votes ≥ $\theta$ = 2 are added to trie**

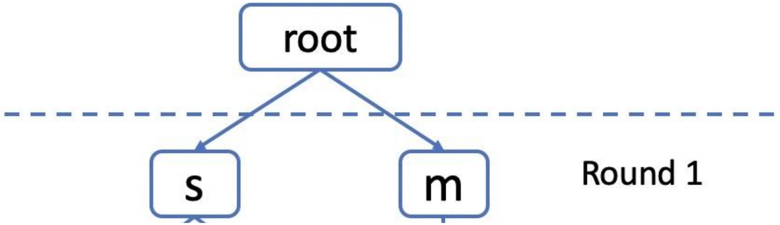
# Round 2: random device selection





**Randomly select m = 10 devices at random**

# Round 2: voting stage



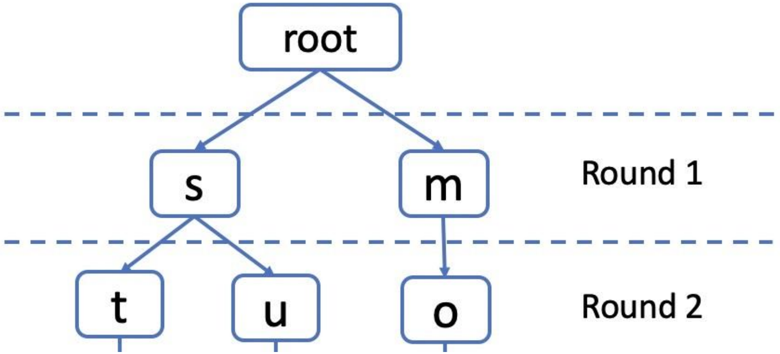

‘su’: 3 votes

‘st’: 2 votes

‘mo’: 2 votes


**Devices with words having a prefix in trie vote**

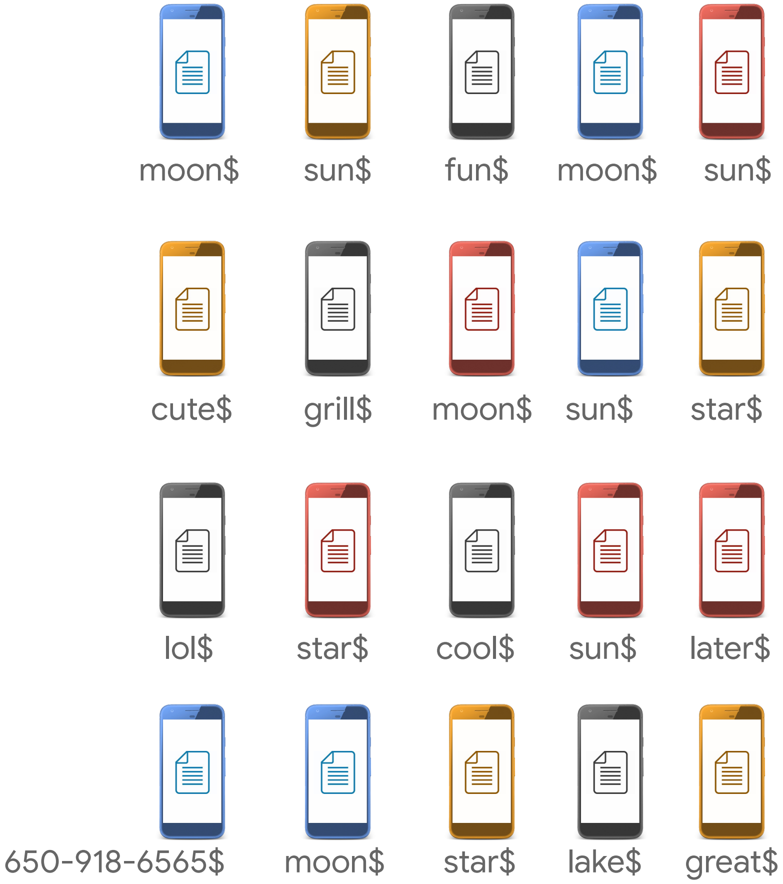
# Round 2: vote thresholding stage



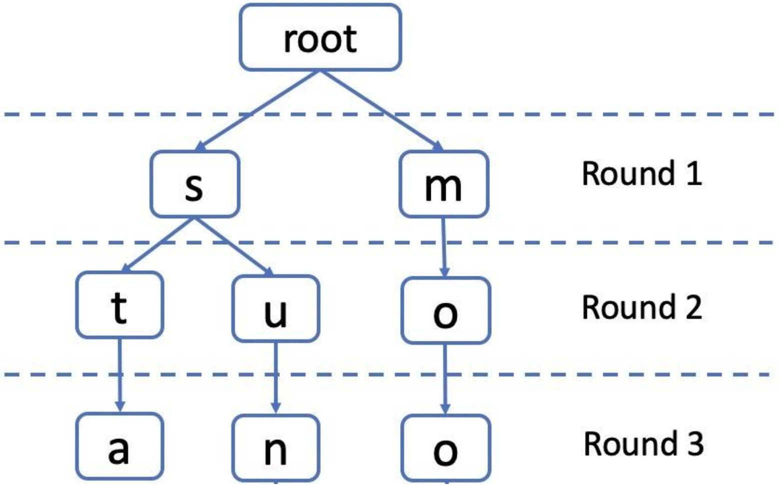

'su': 3 votes

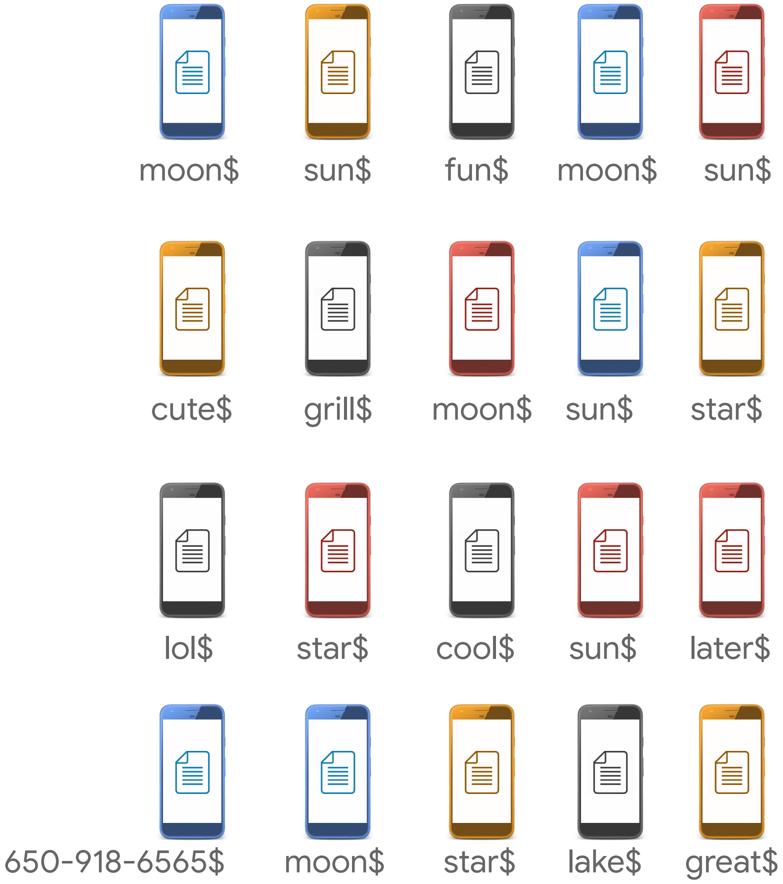
'st': 2 votes

'mo': 2 votes



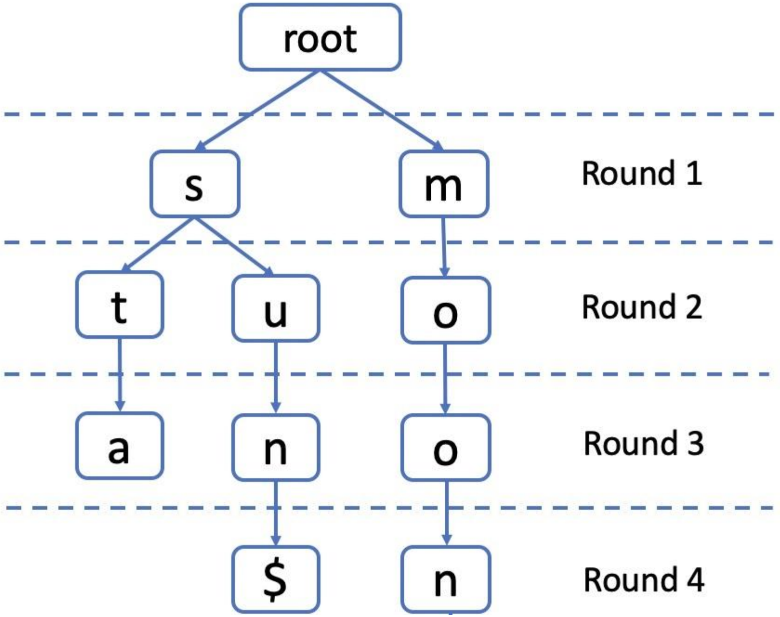
# At the end of round 3

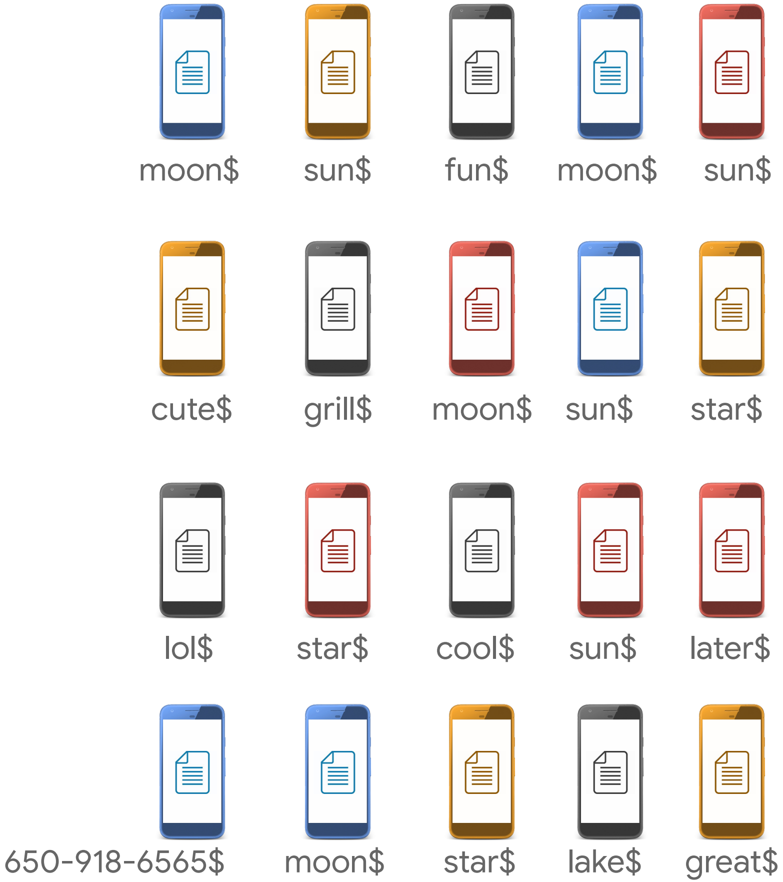


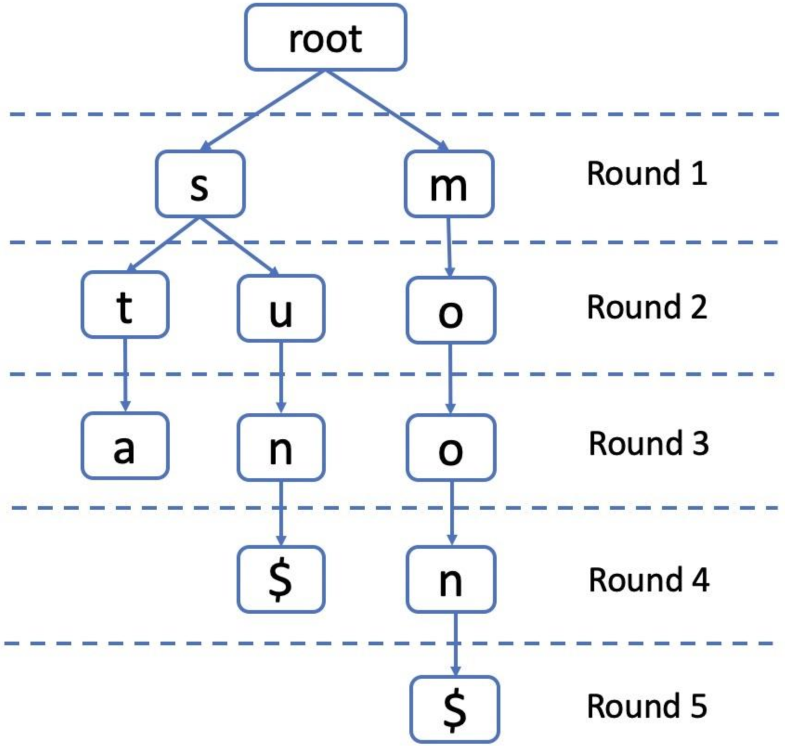

Google

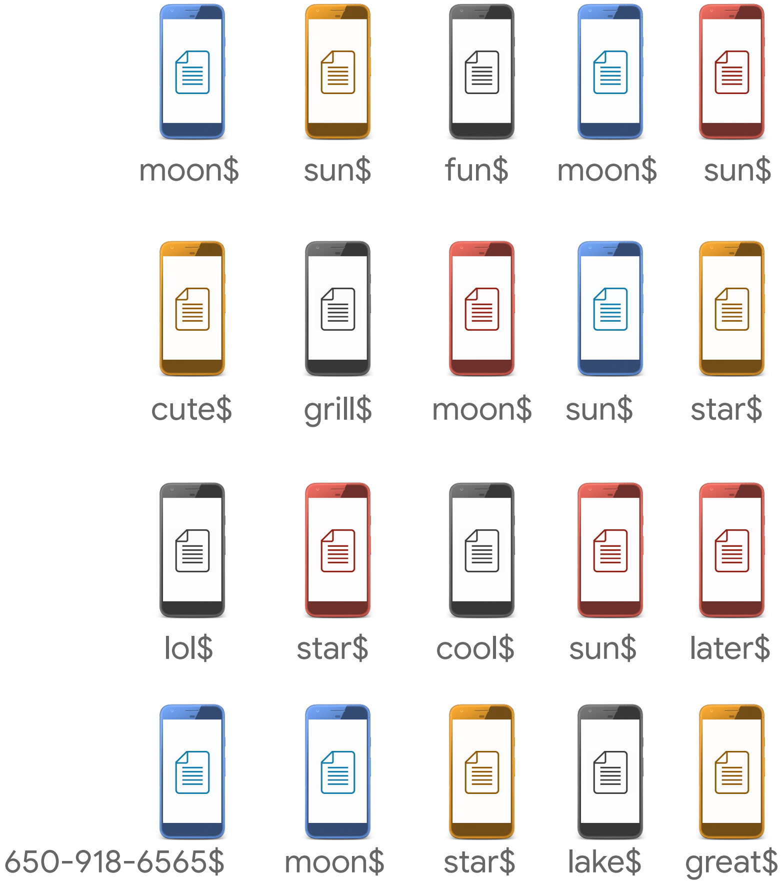
# At the end of round 4



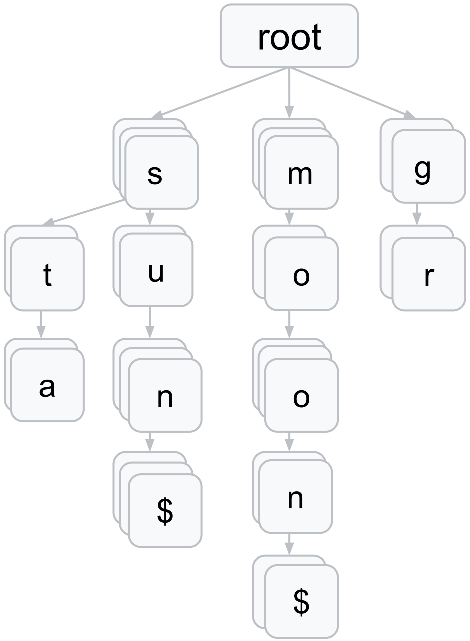
# At the end of round 5

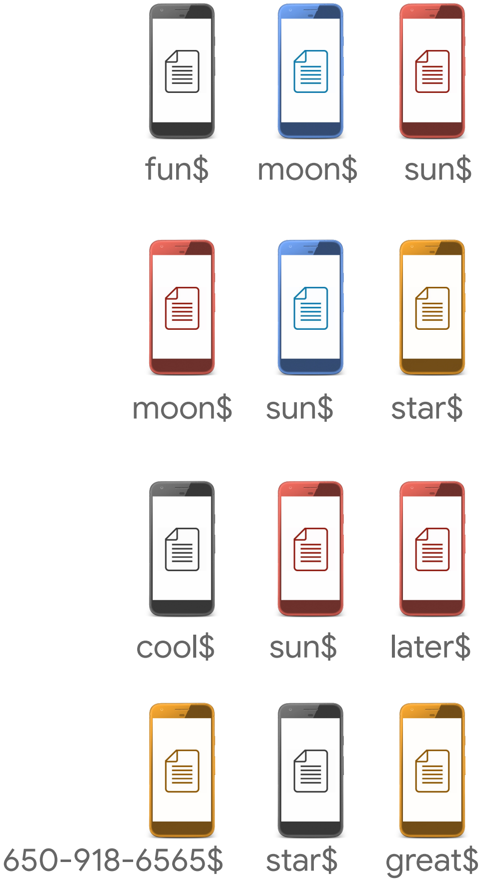



Google

# TrieHH




**Algorithm**
*Input*: List of strings
*Output*: A trie containing popular subsequences

**Key Idea**
Interactively build a trie data structure that keeps track of popular prefixes. Aggregate votes on single character extensions to existing prefixes in the trie. Threshold the counts to ensure that you are only keeping track of popular prefixes.

**Paper:** Federated Heavy Hitters Discovery with Differential Privacy (TPDP19, AISTATS2020)

# Inherent strong privacy guarantees

TrieHH algorithm is differentially private!

- Structural k-anonymity
- User-level (epsilon, delta) central DP
- Great privacy-utility trade-offs
- Limited information exposed to server

Table 1: Choices of $\theta$ and $\gamma$ to achieve $\varepsilon = 2$ in two cases: $\delta \leq \frac{1}{300n}$ and $\delta \leq \frac{1}{n^2}$.

| $n$ | $L = 10$ | | | |
|---|---|---|---|---|
| | $\delta \leq \frac{1}{300n}$ | | $\delta \leq \frac{1}{n^2}$ | |
| | $\theta$ | $\gamma$ | $\theta$ | $\gamma$ |
| $10^4$ | 10 | 1.81 | 12 | 1.51 |
| $10^5$ | 11 | 5.21 | 14 | 4.09 |
| $10^6$ | 12 | 15.10 | 15 | 12.08 |
| $10^7$ | 13 | 44.09 | 17 | 33.71 |

Zhu, Kairouz *et al*. **Federated Heavy Hitter Discovery with Differential Privacy**. *TPDP19, AISTATS2020.*

# Out-of-vocab[1] experiments on Sentiment140

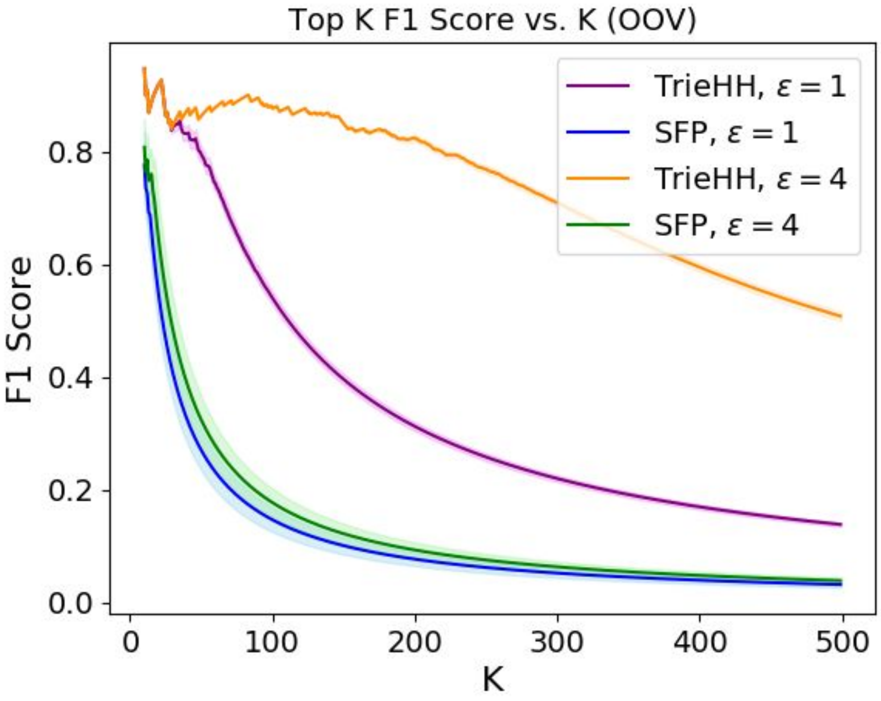



Table 2. Comparison between SFP and TrieHH of recall at K =50 and $\delta = 1/n^2$.

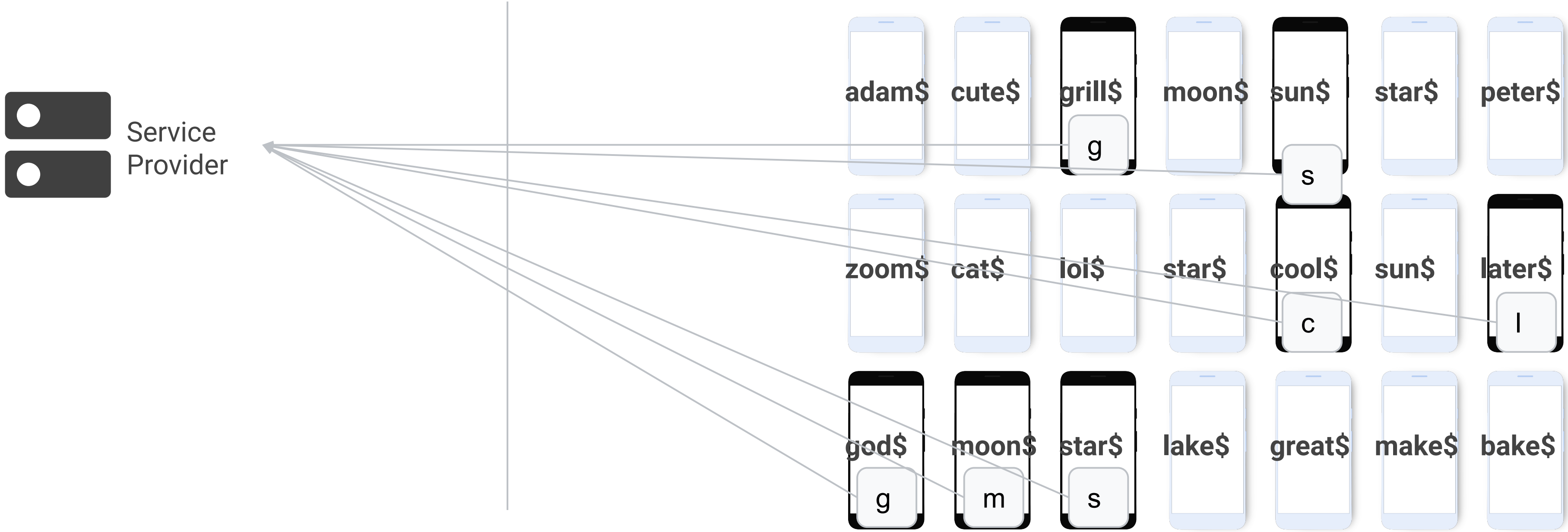
| | [3] $\varepsilon = 1$ | | $\varepsilon = 4$ | |
|---|---|---|---|---|
| | Recall | Prec | Recall | Prec |
| TrieHH | 0.65 | 1 | 0.76 | 1 |
| SFP[2] (20) | 0.17 | 0.853 | 0.19 | 0.867 |
| SFP (80) | 0.25 | 0.494 | 0.325 | 0.456 |

(1) Dictionary contains over 260k words. After removing dictionary words, we lose over 160k users
(2) SFP is an algorithm by Apple for heavy hitter discovery with local DP
(3) SFP's local epsilon is amplified to a central (epsilon, delta) for a fair comparison

# Weaknesses of TrieHH

Google

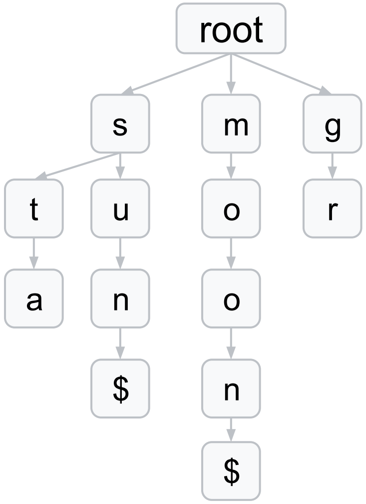
# Linking devices to per-round character extensions




The server can link contributions (character extensions) to devices

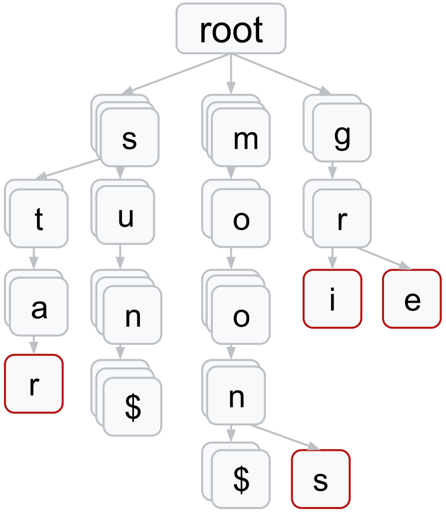
# Learning votes on unpruned edges

**Trie seen by analyst**



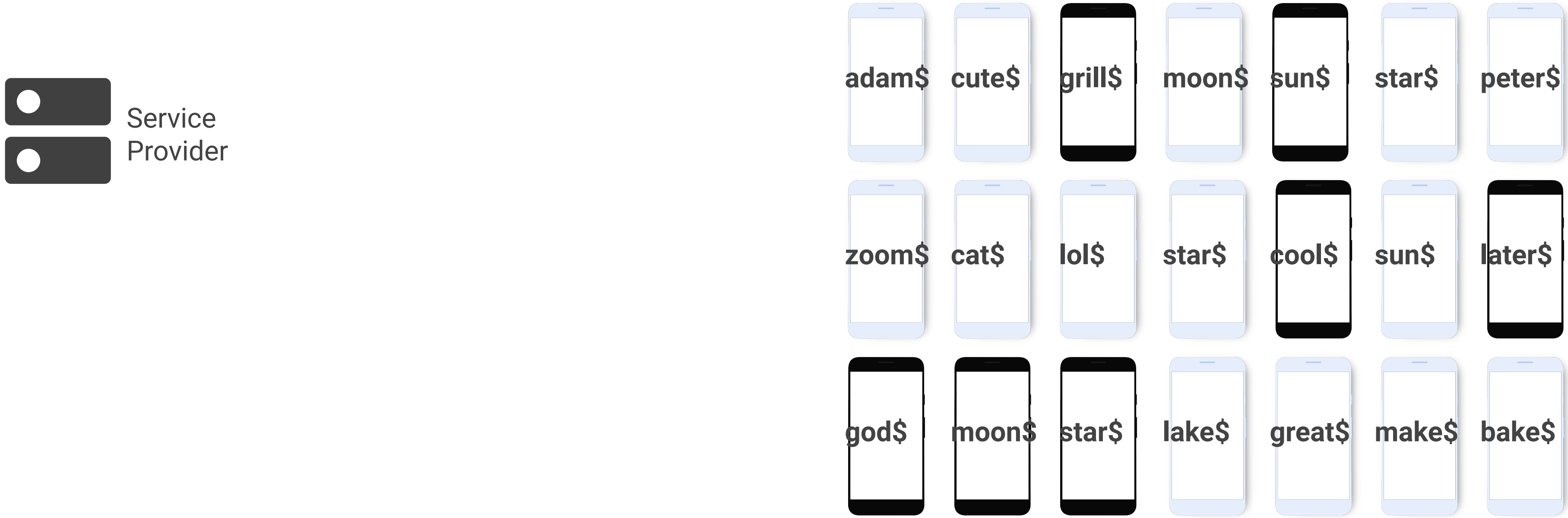

versus

**Trie seen by server**




DP & k-anonymity properties hold with respect to the analyst (not the server!)

# Uniform random device selection

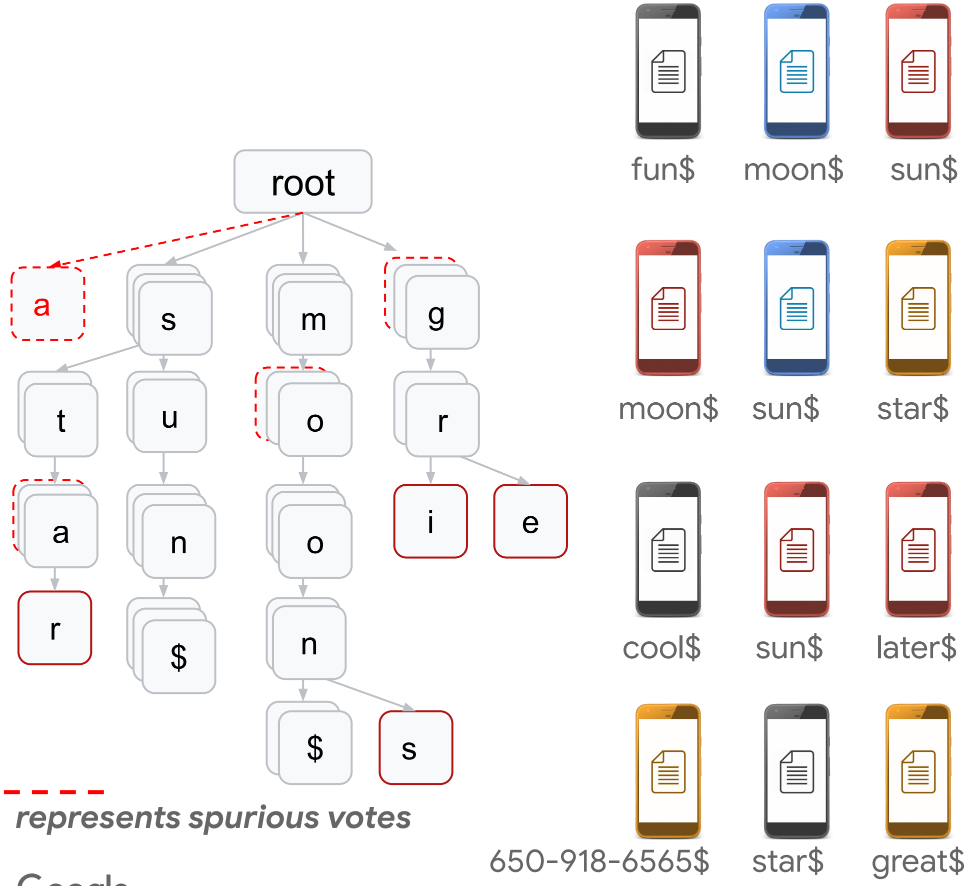



DP holds only when server can sample uniformly at random from the entire population

# Hardened TrieHH

Google

# Hardened TrieHH




**Algorithm**
*Input*: List of strings
*Output:* A trie containing popular subsequences

**Key Idea**
Interactively build a trie data structure that keeps track of popular prefixes. Use ***SecAgg*** to aggregate ***noisy votes*** on single character extensions to existing prefixes in the trie. Threshold the counts to ensure that you are only keeping track of “popular” prefixes.

**Privacy**
Per-round securely aggregated noisy votes are ***automatically differentially private***.

# Advances and Open Problems in Federated Learning


Peter Kairouz[7*] H. Brendan McMahan[7*] Brendan Avent[21] Aurélien Bellet[9]
Mehdi Bennis[19] Arjun Nitin Bhagoji[13] Keith Bonawitz[7] Zachary Charles[7]
Graham Cormode[23] Rachel Cummings[6] Rafael G.L. D'Oliveira[14]
Salim El Rouayheb[14] David Evans[22] Josh Gardner[24] Zachary Garrett[7]
Adrià Gascón[7] Badih Ghazi[7] Phillip B. Gibbons[2] Marco Gruteser[7,14]
Zaid Harchaoui[24] Chaoyang He[21] Lie He [4] Zhouyuan Huo [20]
Ben Hutchinson[7] Justin Hsu[25] Martin Jaggi[4] Tara Javidi[17] Gauri Joshi[2]
Mikhail Khodak[2] Jakub Konečný[7] Aleksandra Korolova[21] Farinaz Koushanfar[17]
Sanmi Koyejo[7,18] Tancrède Lepoint[7] Yang Liu[12] Prateek Mittal[13]
Mehryar Mohri[7] Richard Nock[1] Ayfer Özgür[15] Rasmus Pagh[7,10]
Mariana Raykova[7] Hang Qi[7] Daniel Ramage[7] Ramesh Raskar[11]
Dawn Song[16] Weikang Song[7] Sebastian U. Stich[4] Ziteng Sun[3]
Ananda Theertha Suresh[7] Florian Tramèr[15] Praneeth Vepakomma[11] Jianyu Wang[2]
Li Xiong[5] Zheng Xu[7] Qiang Yang[8] Felix X. Yu[7] Han Yu[12] Sen Zhao[7]

[1]Australian National University, [2]Carnegie Mellon University, [3]Cornell University,
[4]École Polytechnique Fédérale de Lausanne, [5]Emory University, [6]Georgia Institute of Technology,
[7]Google Research, [8]Hong Kong University of Science and Technology, [9]INRIA, [10]IT University of Copenhagen,
[11]Massachusetts Institute of Technology, [12]Nanyang Technological University, [13]Princeton University,
[14]Rutgers University, [15]Stanford University, [16]University of California Berkeley,
[17] University of California San Diego, [18]University of Illinois Urbana-Champaign, [19]University of Oulu,
[20]University of Pittsburgh, [21]University of Southern California, [22]University of Virginia,
[23]University of Warwick, [24]University of Washington, [25]University of Wisconsin–Madison



**Abstract**

Federated learning (FL) is a machine learning setting where many clients (e.g. mobile devices or whole organizations) collaboratively train a model under the orchestration of a central server (e.g. service provider), while keeping the training data decentralized. FL embodies the principles of focused data collection and minimization, and can mitigate many of the systemic privacy risks and costs resulting from traditional, centralized machine learning and data science approaches. Motivated by the explosive growth in FL research, this paper discusses recent advances and presents an extensive collection of open problems and challenges.


## Advances and Open Problems in FL

58 authors from 25 institutions

[arxiv.org/abs/1912.04977](https://arxiv.org/abs/1912.04977)