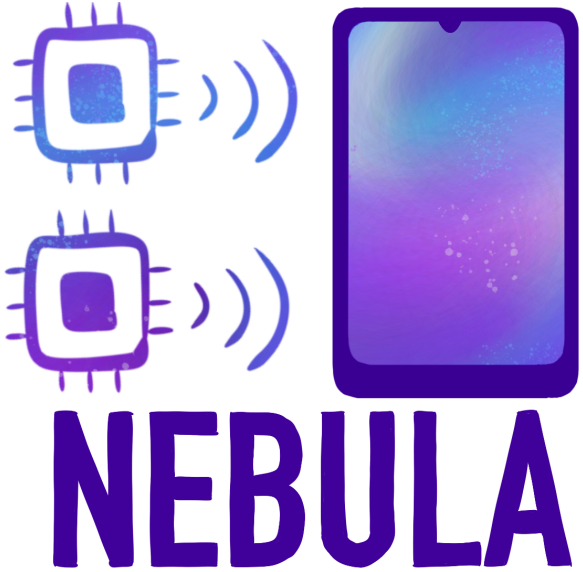
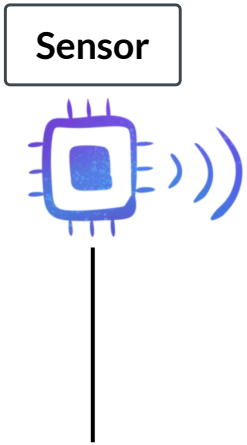


A Privacy-First Platform for Data Backhaul

Jean-Luc Watson, Tess Despres, Alvin Tan, Shishir Patil
Prabal Dutta, Raluca Ada Popa

UC Berkeley

Goal: enable sensor deployments **at large scale**



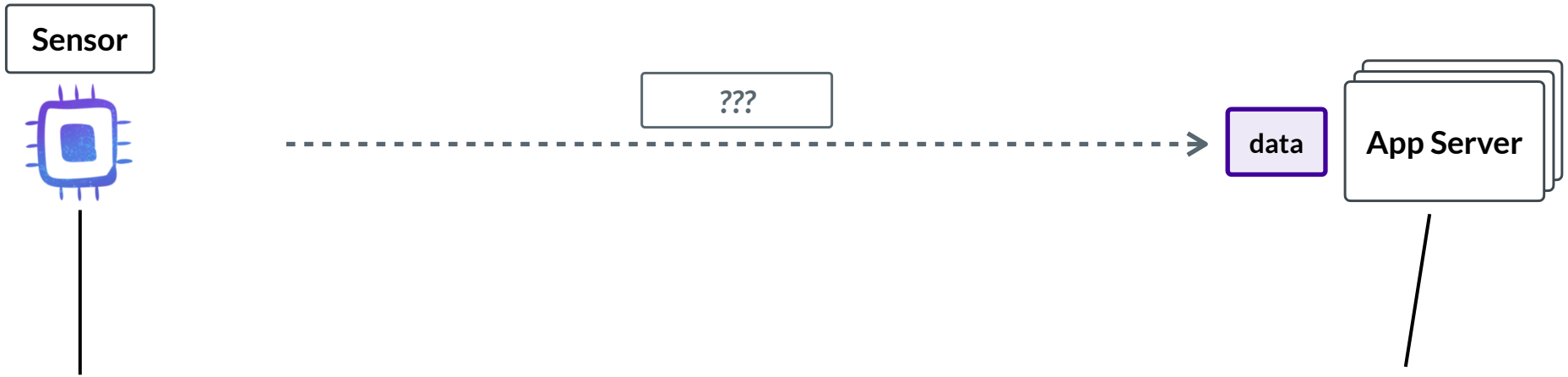
Air quality measurement **on every block**
Public infrastructure use **throughout a city**
Wildlife counting **over square miles**
Package tracking **at every stop**

Problem: data backhaul for useful applications

Air quality measurement **on every block**
Public infrastructure use **throughout a city**
Wildlife counting **over square miles**
Package tracking **at every stop**

Scientific studies
City planners
Conservation agency
Shipping customers

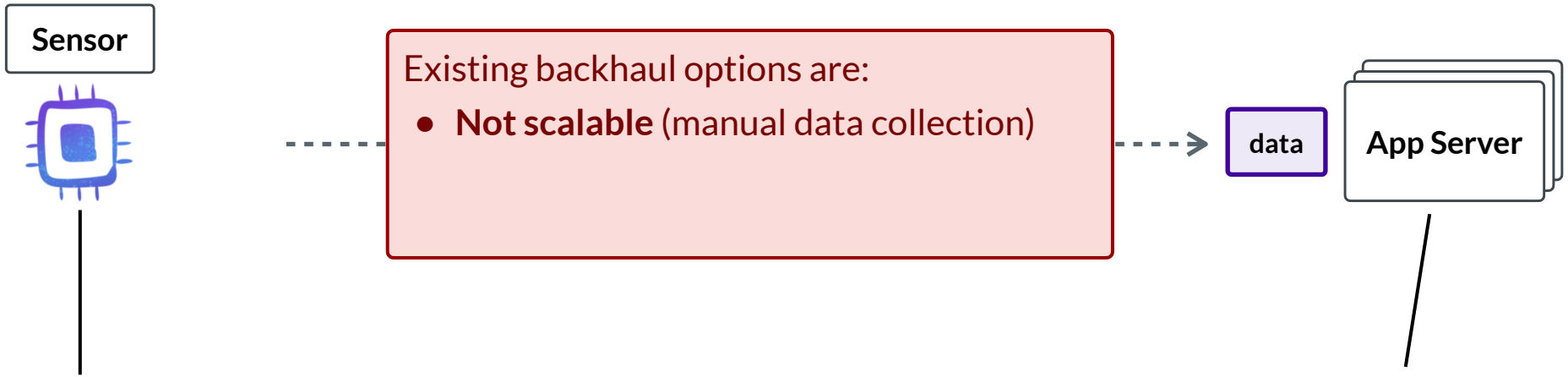
Problem: data backhaul for useful applications



Air quality measurement **on every block**
Public infrastructure use **throughout a city**
Wildlife counting **over square miles**
Package tracking **at every stop**

Scientific studies
City planners
Conservation agency
Shipping customers

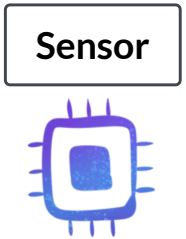
Problem: data backhaul for useful applications



Air quality measurement **on every block**
Public infrastructure use **throughout a city**
Wildlife counting **over square miles**
Package tracking **at every stop**

Scientific studies
City planners
Conservation agency
Shipping customers

Problem: data backhaul for useful applications



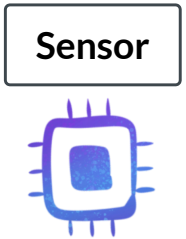
Existing backhaul options are:

- **Not scalable** (manual data collection)
- **Expensive** (app-specific base stations)

Air quality measurement **on every block**
Public infrastructure use **throughout a city**
Wildlife counting **over square miles**
Package tracking **at every stop**

Scientific studies
City planners
Conservation agency
Shipping customers

Problem: data backhaul for useful applications



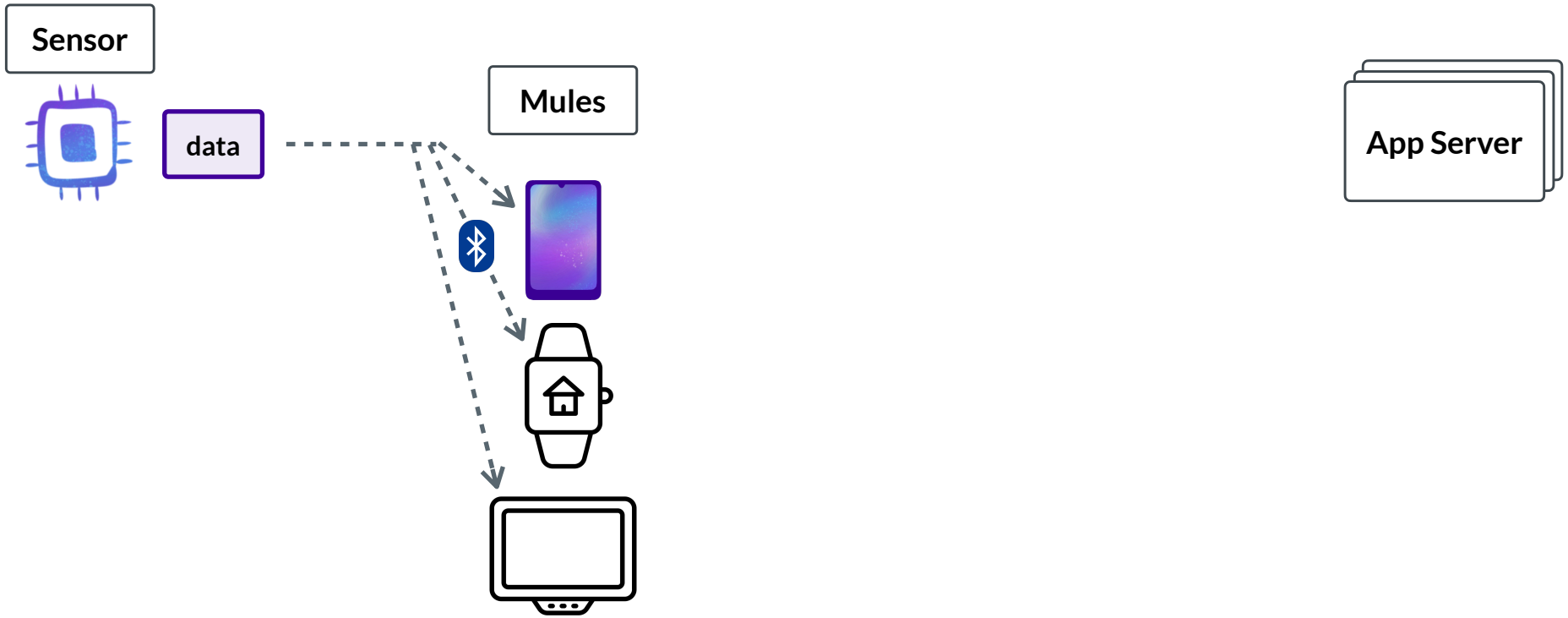
Existing backhaul options are:

- **Not scalable** (manual data collection)
- **Expensive** (app-specific base stations)
- **Power-hungry** (cellular radios)

Air quality measurement **on every block**
Public infrastructure use **throughout a city**
Wildlife counting **over square miles**
Package tracking **at every stop**

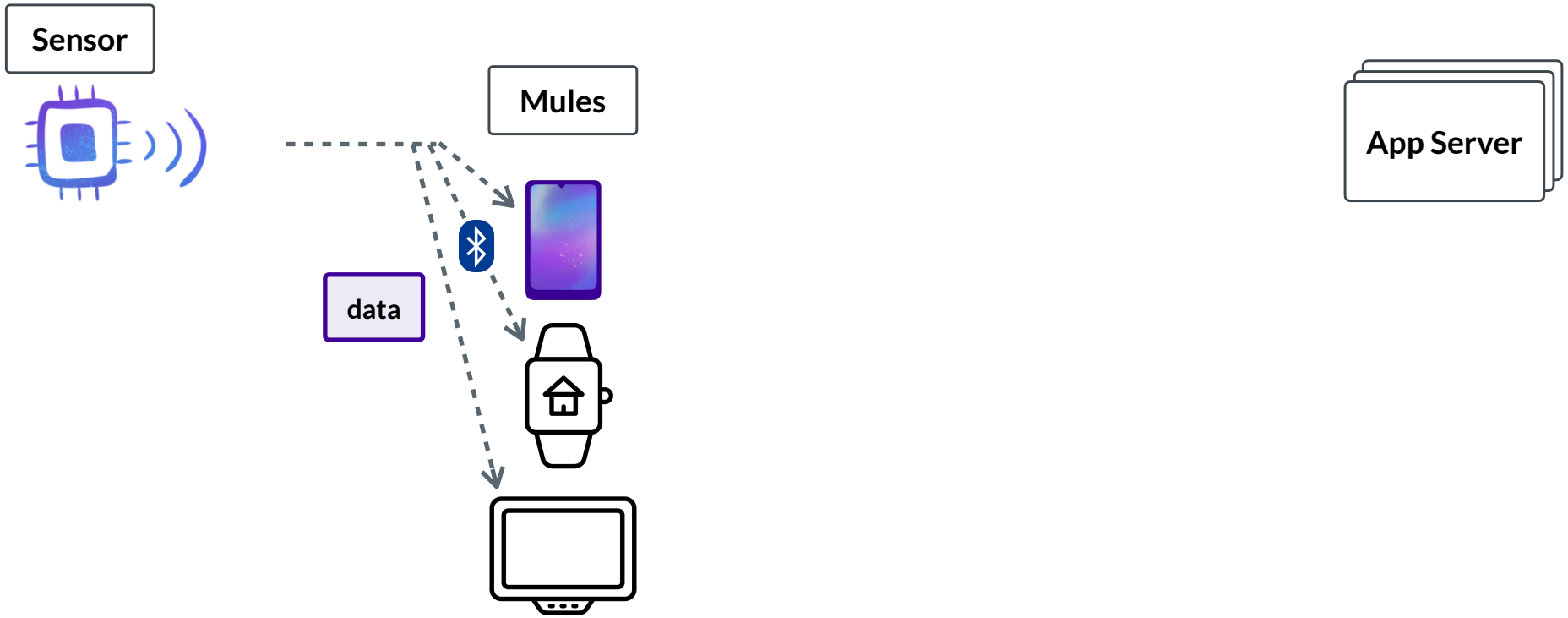
Scientific studies
City planners
Conservation agency
Shipping customers

Instead, 'mules' in close proximity could gather data



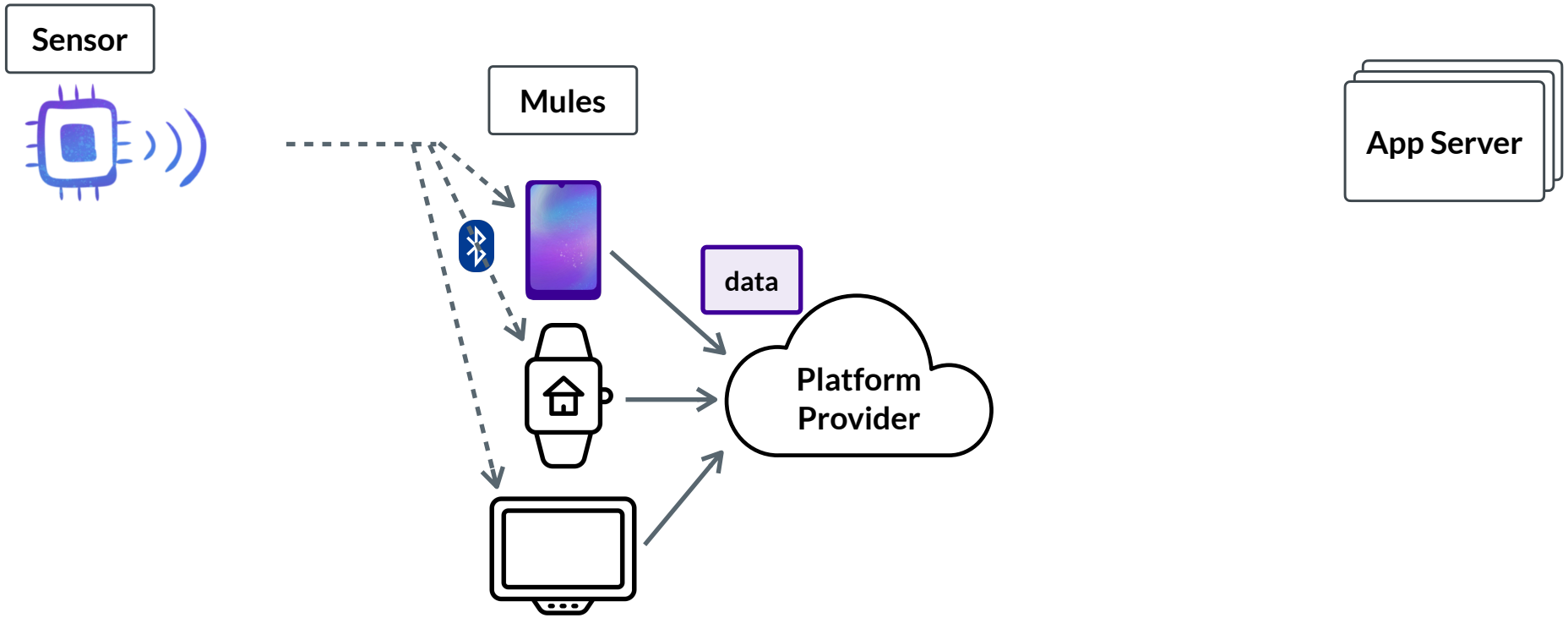
[icons by Icons Field from Noun Project]

Instead, 'mules' in close proximity could gather data



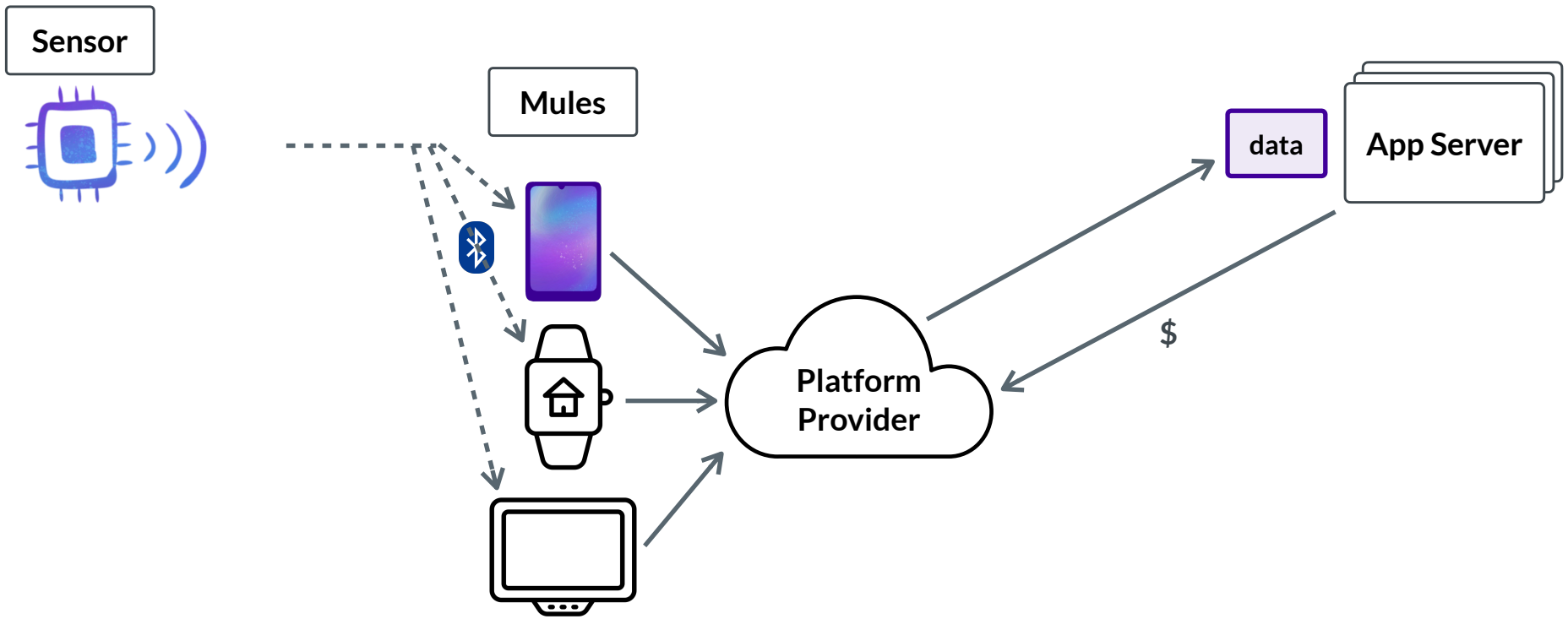
[icons by Icons Field from Noun Project]

Mules send data to a backhaul provider



[icons by Icons Field from Noun Project]

The provider forwards to paying servers

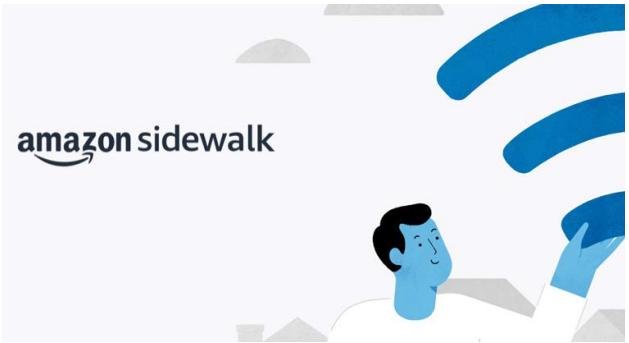


[icons by Icons Field from Noun Project]

The provider forwards to paying servers

Sensor

Backhaul systems are *already* being deployed

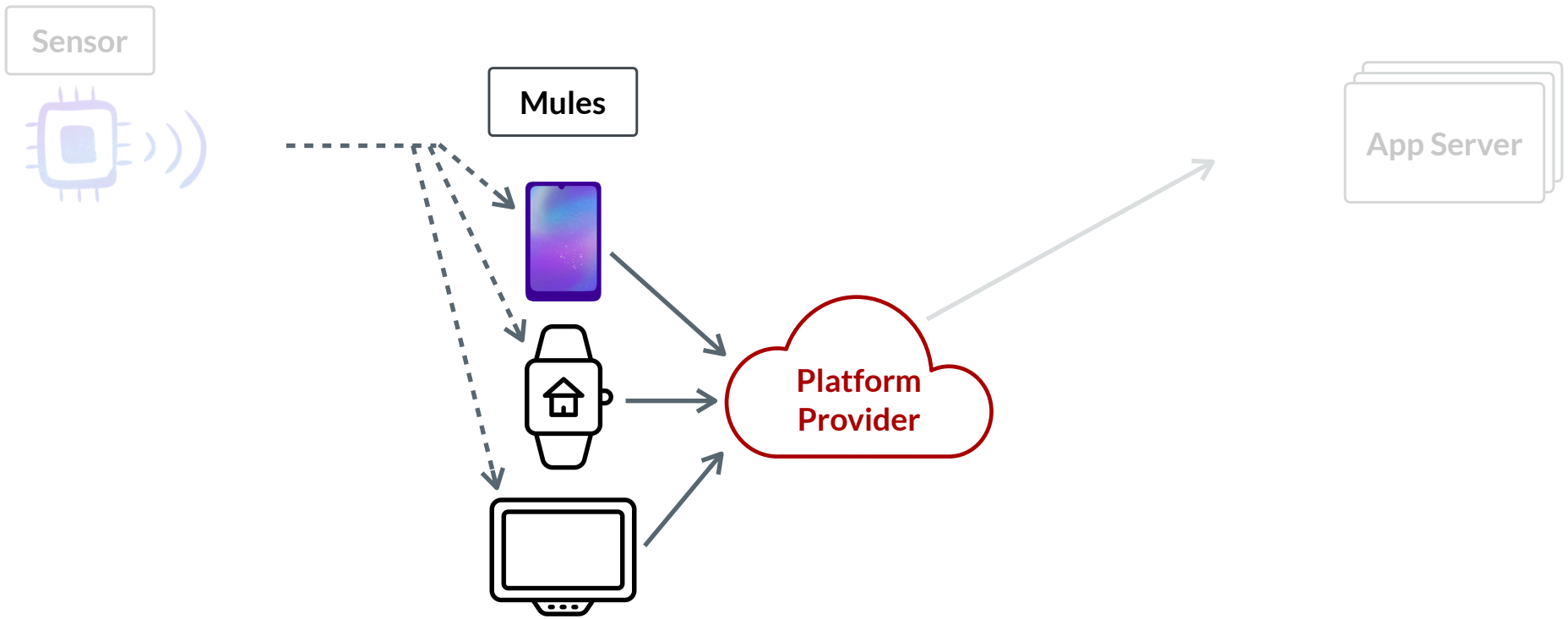


Find My

Exposure Notification

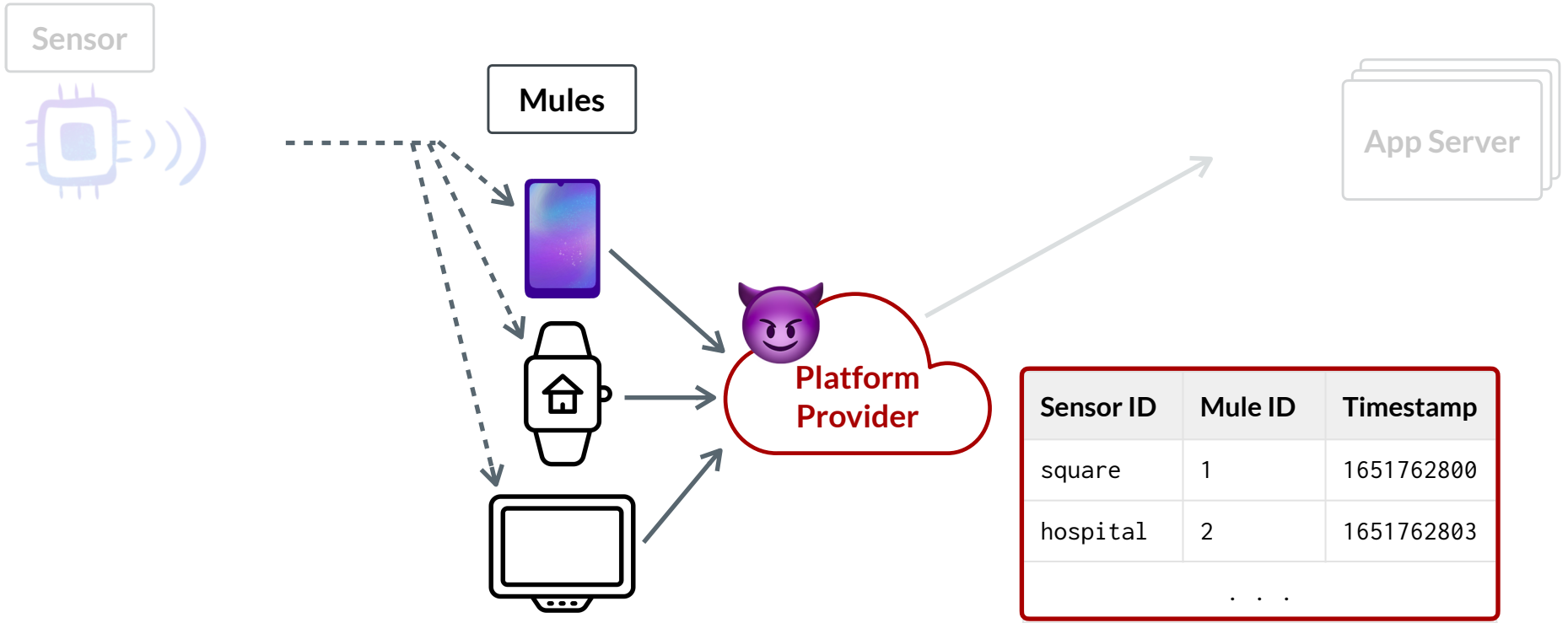
[icons by Icons Field from Noun Project]

Centralized backhaul threatens privacy



[icons by Icons Field from Noun Project]

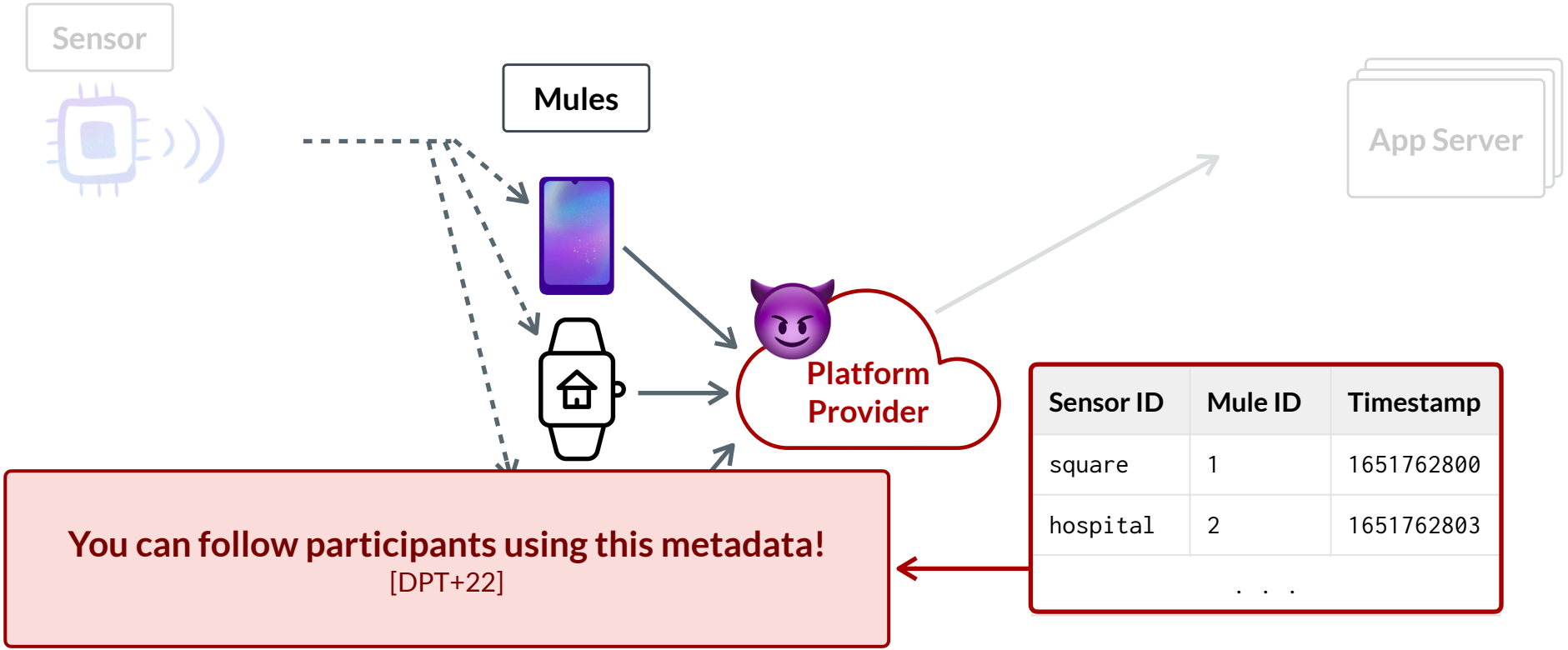
Centralized backhaul threatens privacy



Sensor ID	Mule ID	Timestamp
square	1	1651762800
hospital	2	1651762803
. . .		

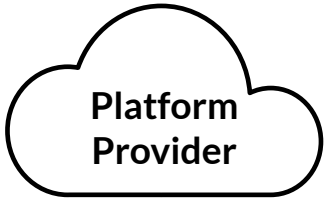
[icons by Icons Field from Noun Project]

Centralized backhaul threatens privacy



[icons by Icons Field from Noun Project]

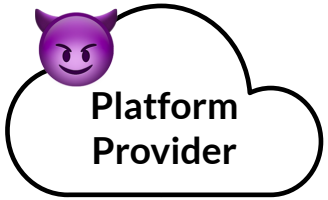
Challenge: maintaining usability



General-purpose
Scalability
Preventing misuse
Payment
Privacy

Challenge: maintaining usability

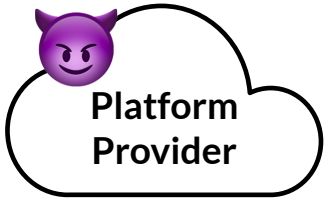
Provider can run a **general-purpose** service that is not private



	Sidewalk
General-purpose	✅
Scalability	✅
Preventing misuse	✅
Payment	✅
Privacy	❌

Challenge: maintaining usability

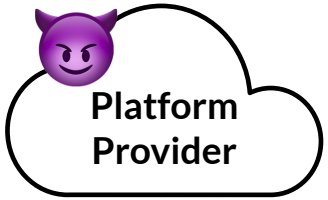
Provider can run a **private** service if it limits application scope



	Sidewalk	FindMy
General-purpose	✅	❌
Scalability	✅	✅
Preventing misuse	✅	❌
Payment	✅	❌
Privacy	❌	✅*

Challenge: maintaining usability

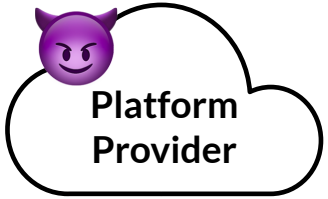
Provider can run a **private and general-purpose** service but not at scale



	Sidewalk	FindMy	Private communication [Chaum88, HLZ+15, ECG+21] Anonymous sensors [CKD+08, KTC+08]
General-purpose	✅	❌	✅
Scalability	✅	✅	❌
Preventing misuse	✅	❌	✅
Payment	✅	❌	❌
Privacy	❌	✅*	✅

Challenge: maintaining usability

Nebula accomplishes each of these at the same time



	Sidewalk	FindMy	Private communication [Chaum88, HLZ+15, ECG+21] Anonymous sensors [CKD+08, KTC+08]	**Nebula**
General-purpose	✅	❌	✅	✅
Scalability	✅	✅	❌	✅
Preventing misuse	✅	❌	✅	✅
Payment	✅	❌	❌	✅
Privacy	❌	✅*	✅	✅

NEBULA

A backhaul system that *preserves mule privacy* while:

1. Charging for backhaul use and incentivizing mule participation

NEBULA

A backhaul system that *preserves mule privacy* while:

1. Charging for backhaul use and incentivizing mule participation

> Out-of-band accounting: app servers buy untraceable *tokens* from provider and exchange them for data.

NEBULA

A backhaul system that *preserves mule privacy* while:

1. Charging for backhaul use and incentivizing mule participation

   Out-of-band accounting: app servers buy untraceable *tokens* from provider and exchange them for data.

2. Detecting mule and application server misbehavior

NEBULA

A backhaul system that *preserves mule privacy* while:

1. Charging for backhaul use and incentivizing mule participation

   Out-of-band accounting: app servers buy untraceable *tokens* from provider and exchange them for data.

2. Detecting mule and application server misbehavior

   Allow mules to submit complaints with *proof* of misbehavior to the provider.

NEBULA

A backhaul system that *preserves mule privacy* while:

1. Charging for backhaul use and incentivizing mule participation

   Out-of-band accounting: app servers buy untraceable *tokens* from provider and exchange them for data.

2. Detecting mule and application server misbehavior

   Allow mules to submit anonymous complaints with *proof* of misbehavior to the provider.

3. Supporting millions of mules

   Nebula can process ~445k tokens/second, or **250M tokens/dollar**

NEBULA

A backhaul system that *preserves mule privacy* while:

1. Charging for backhaul use and incentivizing mule participation

   Out-of-band accounting: app servers buy untraceable *tokens* from provider and exchange them for data.

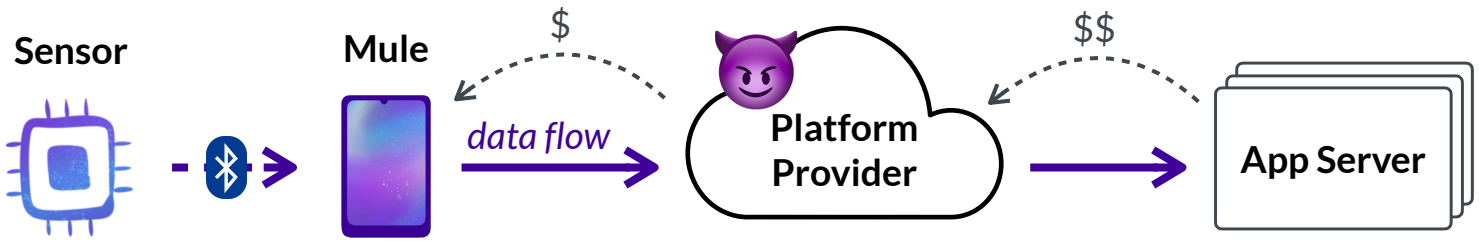
2. Detecting mule and application server misbehavior

   Allow mules to submit anonymous complaints with *proof* of misbehavior to the provider.

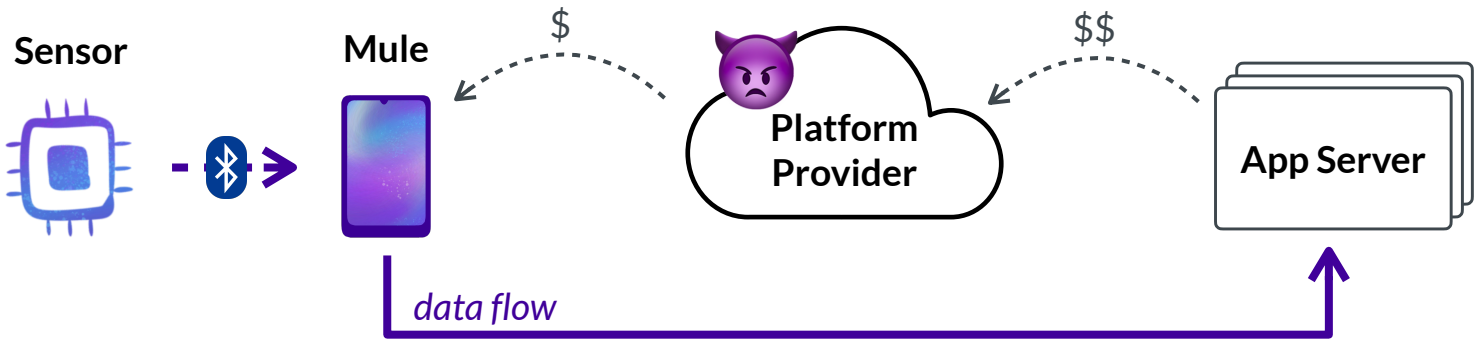
3. Supporting millions of mules

   Nebula can process ~445k tokens/second, or **250M tokens/dollar**

Nebula’s approach: decentralize protocol onto mules



Nebula’s approach: decentralize protocol onto mules



Core Design: Tokens

- System usage is tracked with PrivacyPass [DGS+18] **tokens** 
  - *Unlinkable*: when tokens are redeemed to a malicious server, cannot be linked to the client(s) that generated them

[icons by Chunk Icons from Noun Project]

Core Design: Tokens

- System usage is tracked with PrivacyPass [DGS+18] **tokens** 
  - *Unlinkable*: when tokens are redeemed to a malicious server, cannot be linked to the client(s) that generated them

Provider controls access to system resources without knowing the token owner

[icons by Chunk Icons from Noun Project]

Core Design: Tokens

- System usage is tracked with PrivacyPass [DGS+18] **tokens** 
  - *Unlinkable*: when tokens are redeemed to a malicious server, cannot be linked to the client(s) that generated them

    Provider controls access to system resources without knowing the token owner

  - *One-more-token security*: even knowing many valid tokens, a malicious client cannot forge more

[icons by Chunk Icons from Noun Project]

Core Design: Tokens

- System usage is tracked with PrivacyPass [DGS+18] **tokens** 
  - *Unlinkable*: when tokens are redeemed to a malicious server, cannot be linked to the client(s) that generated them

    Provider controls access to system resources without knowing the token owner

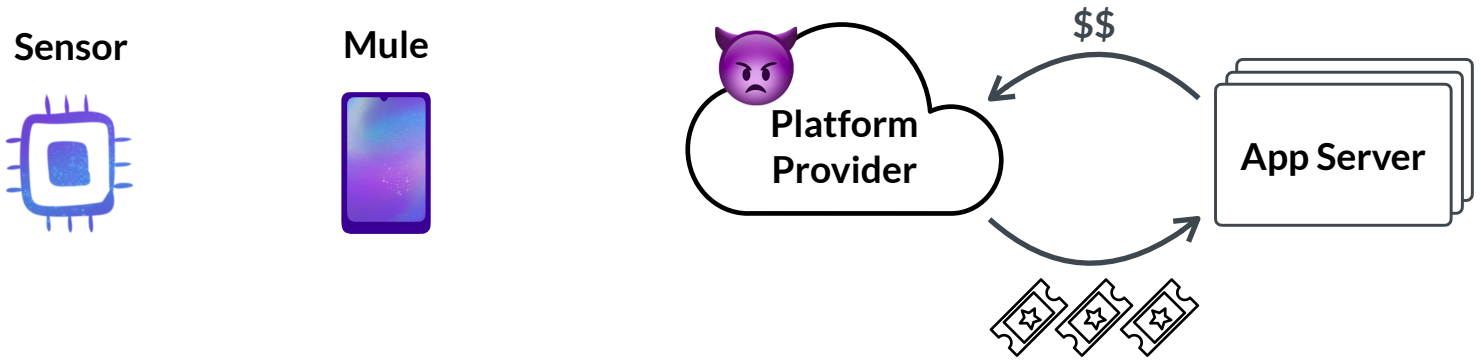
  - *One-more-token security*: even knowing many valid tokens, a malicious client cannot forge more

    Only way to acquire more tokens is to participate in backhaul

[icons by Chunk Icons from Noun Project]

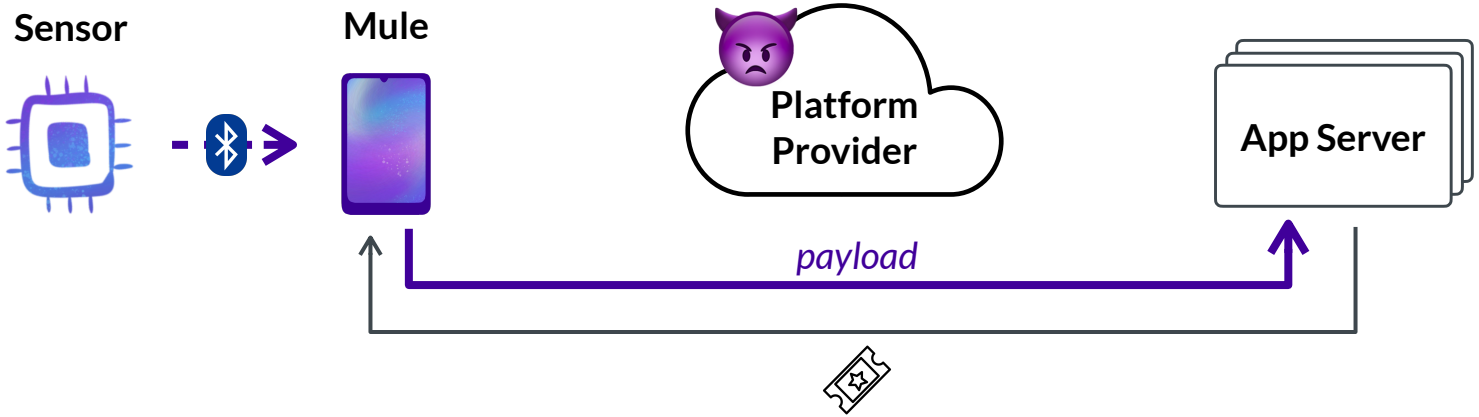
(1) Token Pre-purchase

- At the beginning of an epoch, app servers *pre-purchase* tokens from the provider



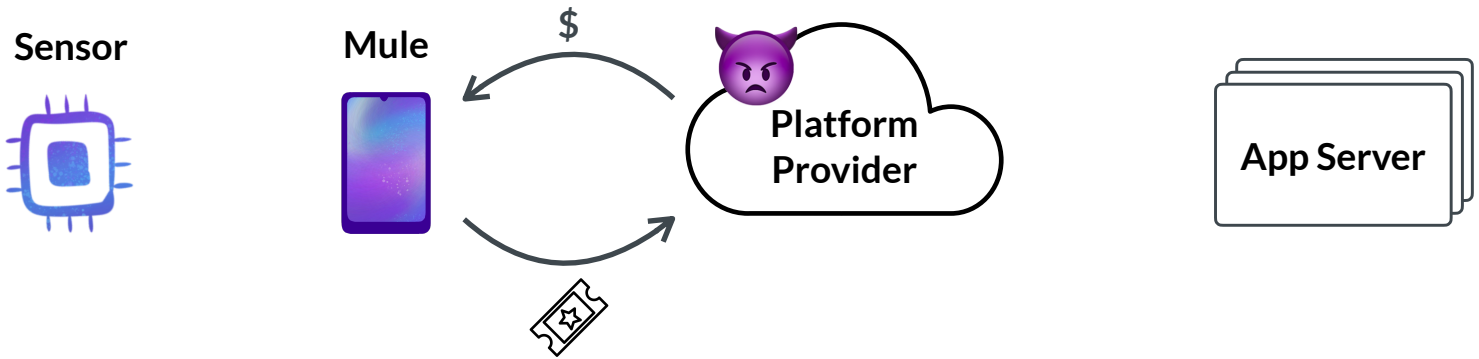
(2) Payload Delivery

- Mules send data directly to application servers over anonymous connections and receive tokens in exchange



(3) Token Redemption

- At the end of an epoch, mules redeem tokens with the platform provider in exchange for compensation



NEBULA

A backhaul system that *preserves mule privacy* while:

1. Charging for backhaul use and incentivizing mule participation

   Out-of-band accounting: app servers buy untraceable *tokens* from provider and exchange them for data.

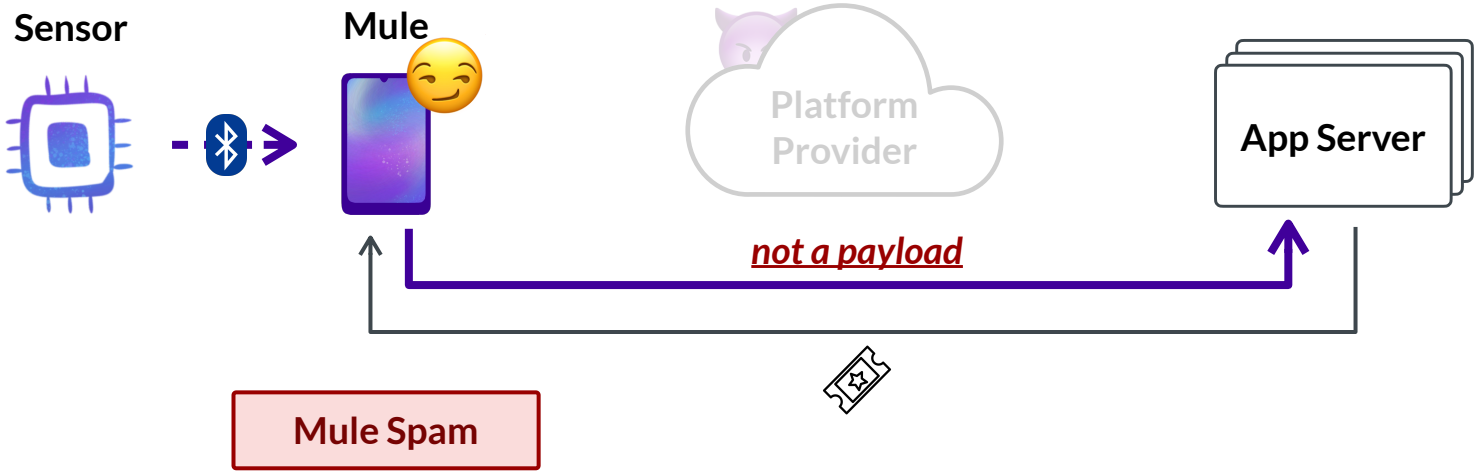
2. Detecting mule and application server misbehavior

   Allow mules to submit complaints with *proof* of misbehavior to the provider.

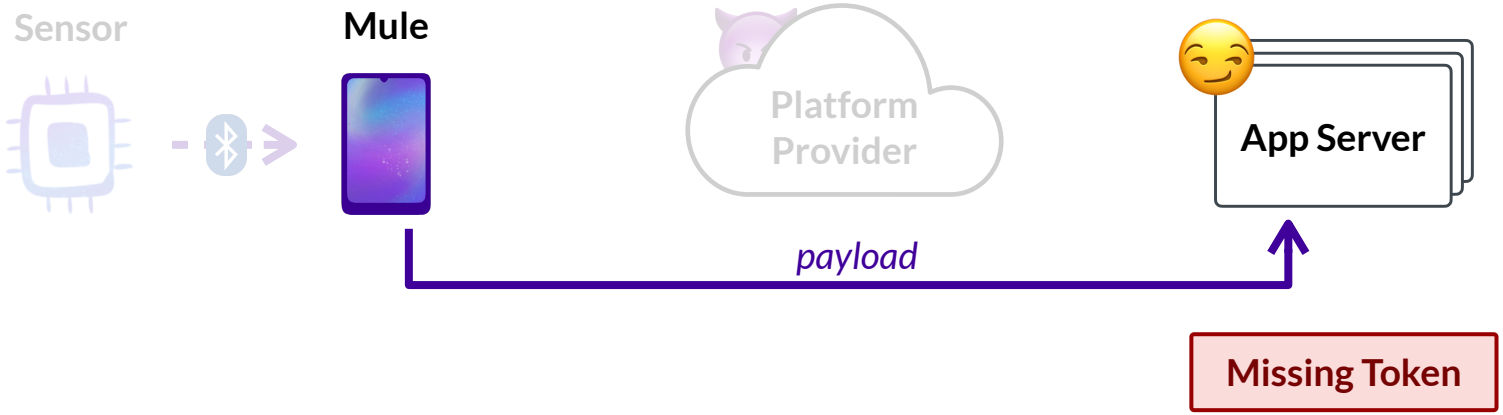
3. Supporting millions of mules

   Nebula can process ~445k tokens/second, or **250M tokens/dollar**

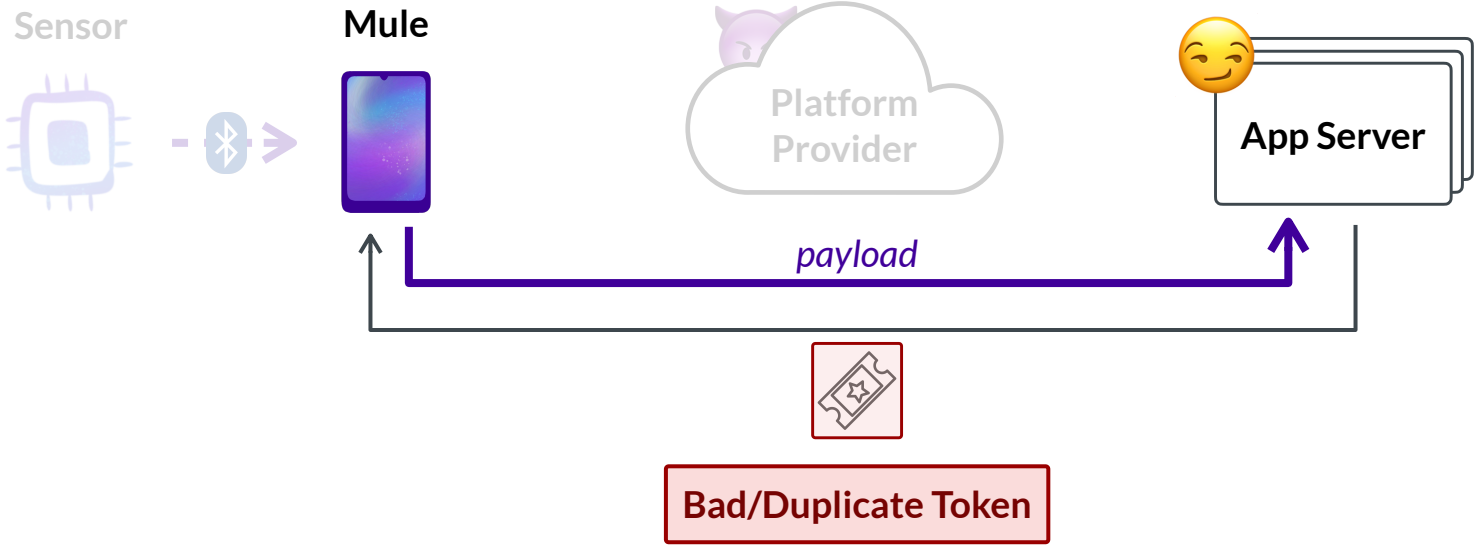
Big problem: **delivery misbehavior**



Big problem: **delivery misbehavior**

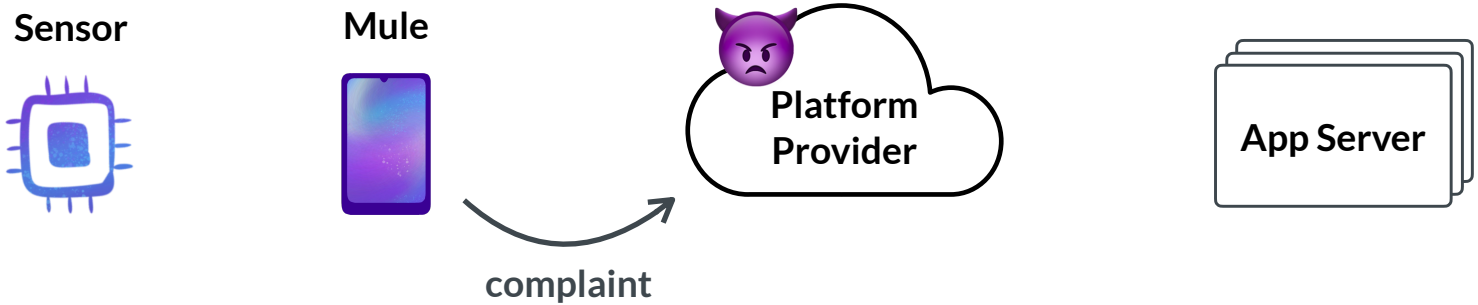


Big problem: **delivery misbehavior**



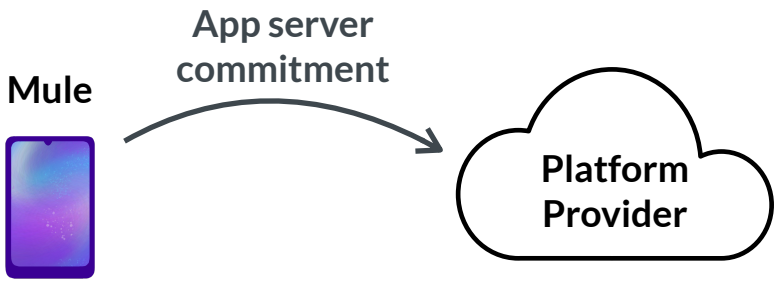
(4) Complaint

- After an epoch, if a mule notices misbehavior (e.g. invalid token), it can *complain* to the provider for a new one



Complaints are based on app server commitments

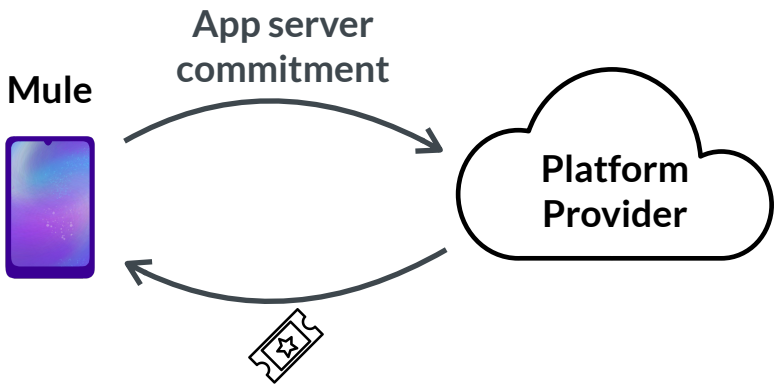
Before receiving data, app servers commit to token they will use if payload is uploaded



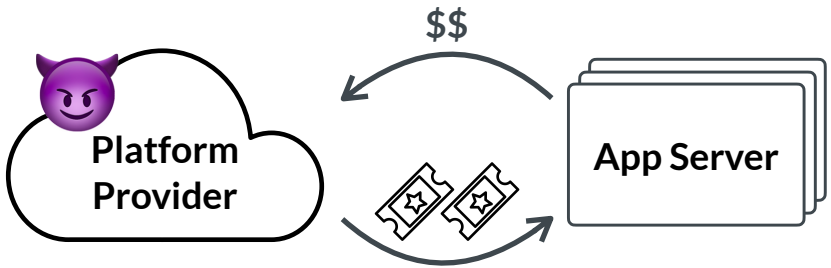
Complaints are based on app server commitments

Before receiving data, app servers commit to token they will use if payload is uploaded

A *new* token is granted for *proof of misbehavior*



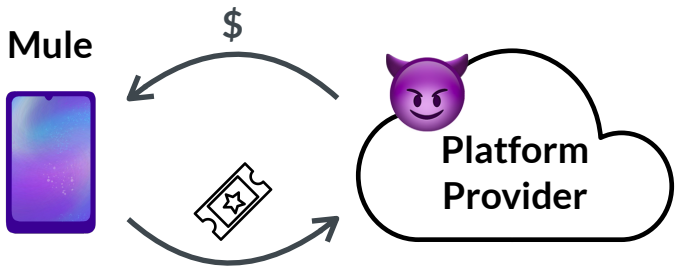
Privacy guarantees



Overall, the provider learns:

1. Count of prepaid tokens purchased by each app server

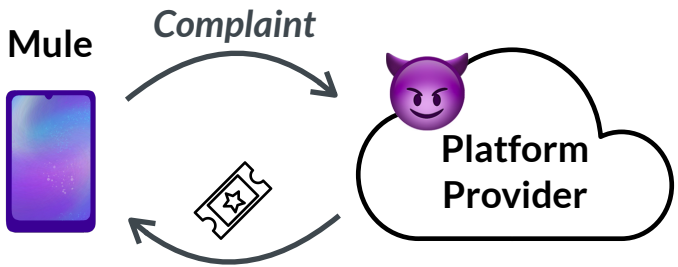
Privacy guarantees



Overall, the provider learns:

1. Count of prepaid tokens purchased by each app server
2. Count of payloads a mule uploaded, across all app servers

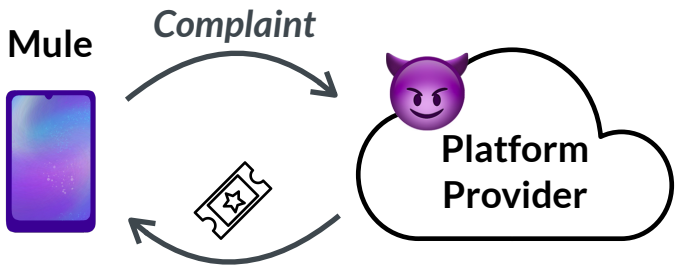
Privacy guarantees



Overall, the provider learns:

1. Count of prepaid tokens purchased by each app server
2. Count of payloads a mule uploaded, across all app servers
3. Set of anonymous complaints against app server misbehavior

Privacy guarantees



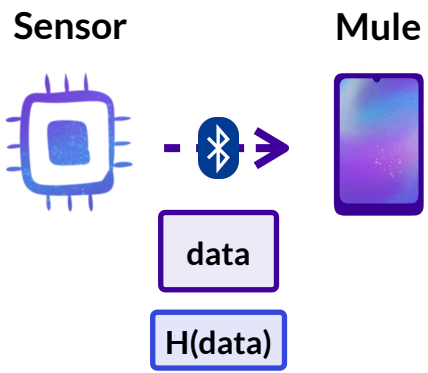
Overall, the provider learns:

Formal Proof Sketch in Paper

1. Count of prepaid tokens purchased by each app server
2. Count of payloads a mule uploaded, across all app servers
3. Set of anonymous complaints against app server misbehavior

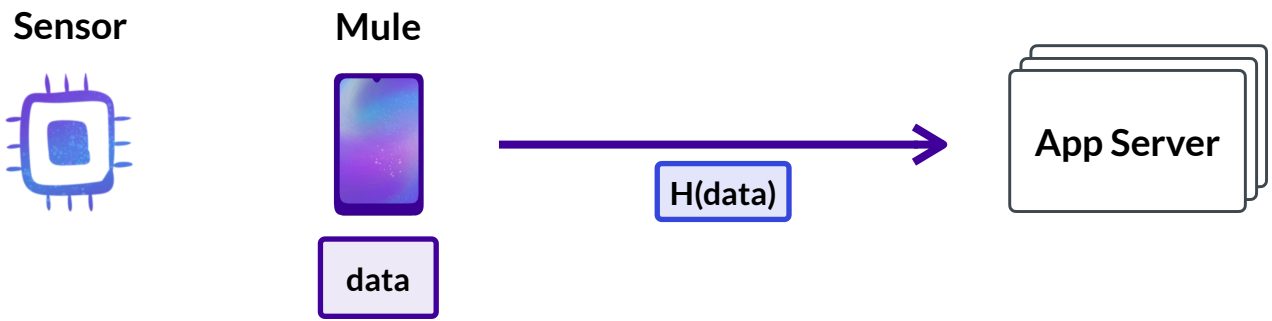
Delivery in Detail

(1) Mule encounters and authenticates a sensor, receiving a E2E-encrypted payload and a *sensor-signed payload hash*



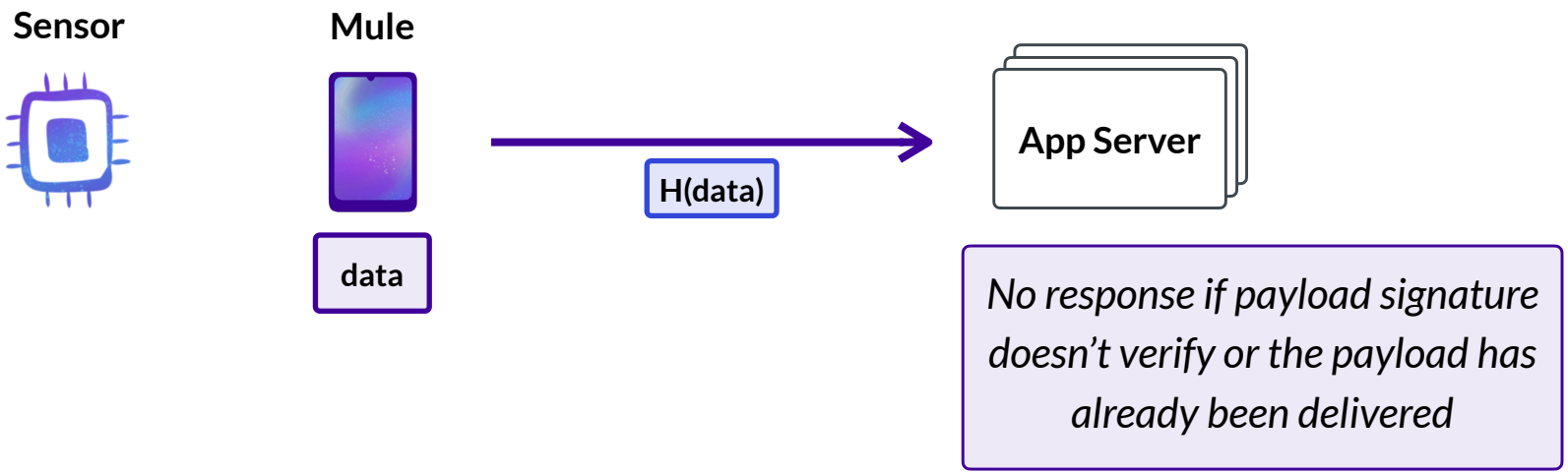
Delivery in Detail

(2) Mule sends signed hash to desired application server. Server checks hash and decides whether they want the payload.



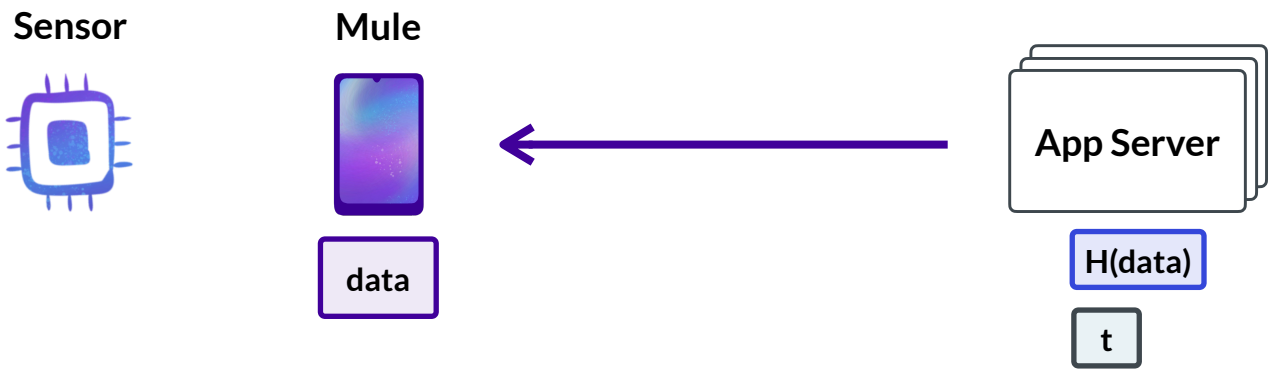
Delivery in Detail

(2) Mule sends signed hash to desired application server. Server checks hash and decides whether they want the payload.



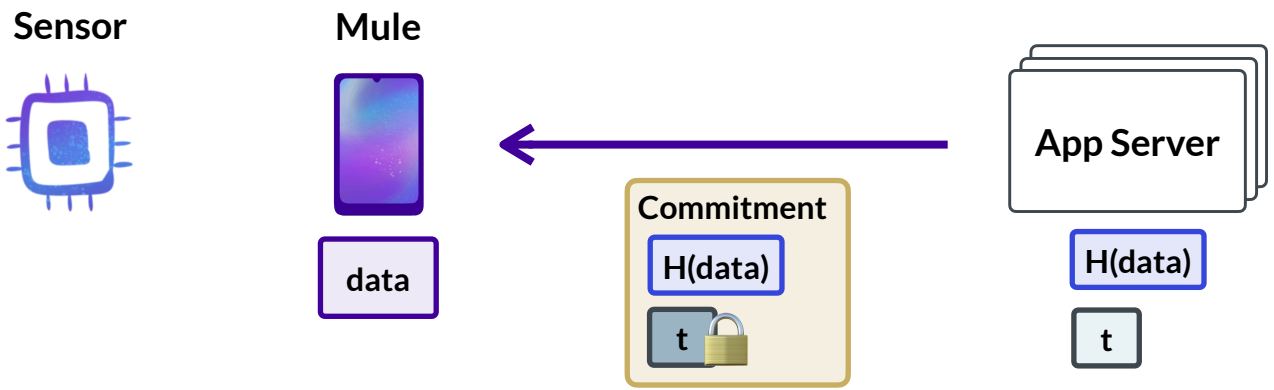
Delivery in Detail

(3) If the server would like the payload, they sign a **commitment** to the hash and an encrypted token they are willing to spend in exchange



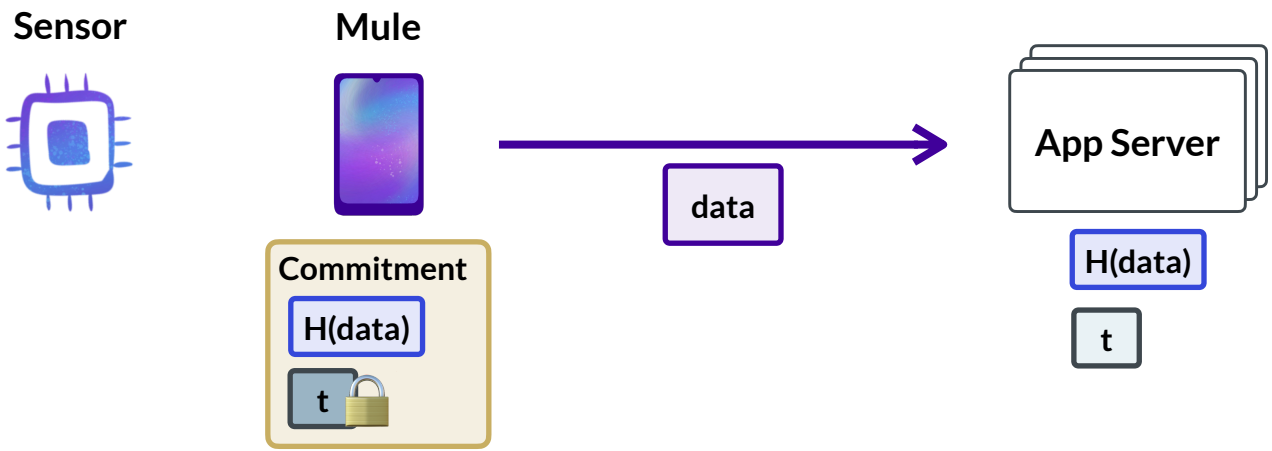
Delivery in Detail

(3) If the server would like the payload, they sign a **commitment** to the hash and an encrypted token they are willing to spend in exchange



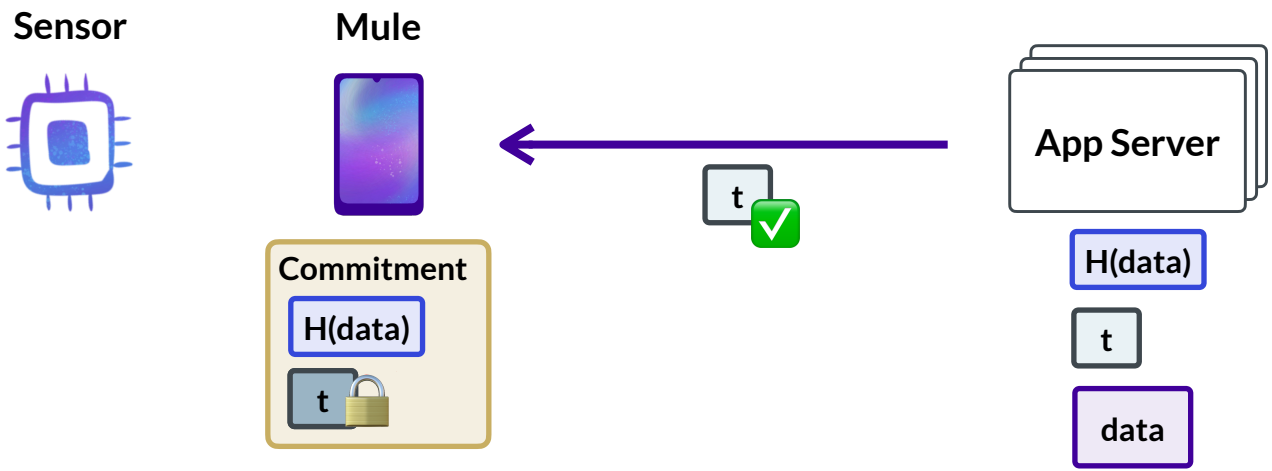
Delivery in Detail

(4) Commitment in hand, the mule sends the actual payload to the app server.



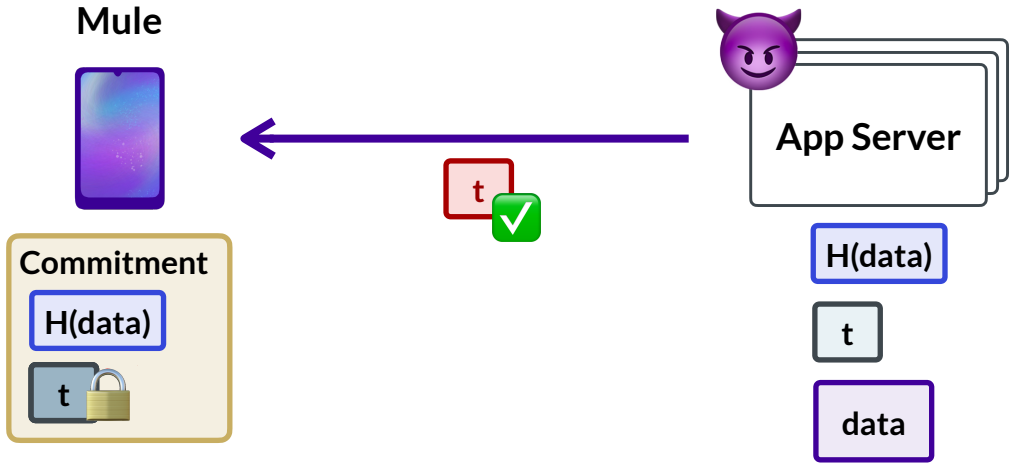
Delivery in Detail

(5) The server checks the data against the hash it previously got; if successful, it signs a response containing the unencrypted token



Complaints in Detail

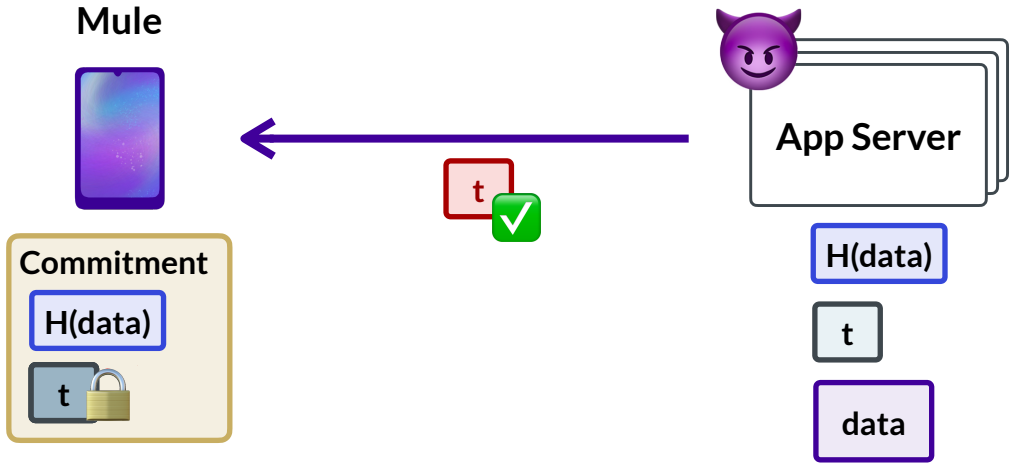
What happens if the mule gets a bad token, or no token at all?



Complaints in Detail

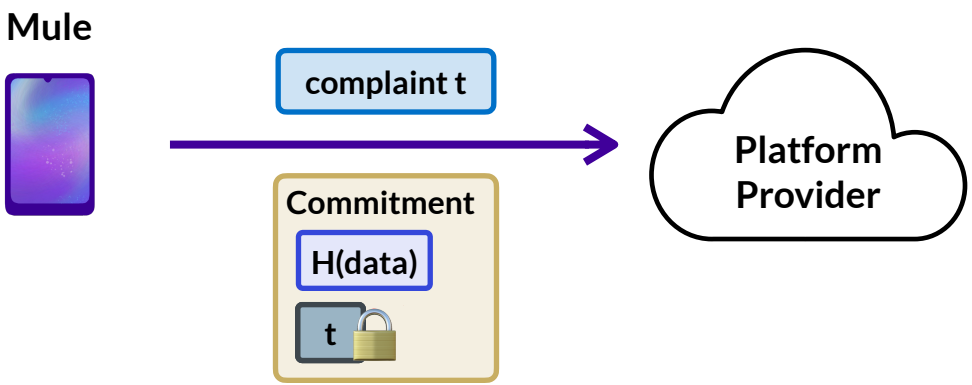
Provider will notify mule of duplicate/invalid tokens it redeems at end of epoch

What happens if the mule gets a bad token, or no token at all?



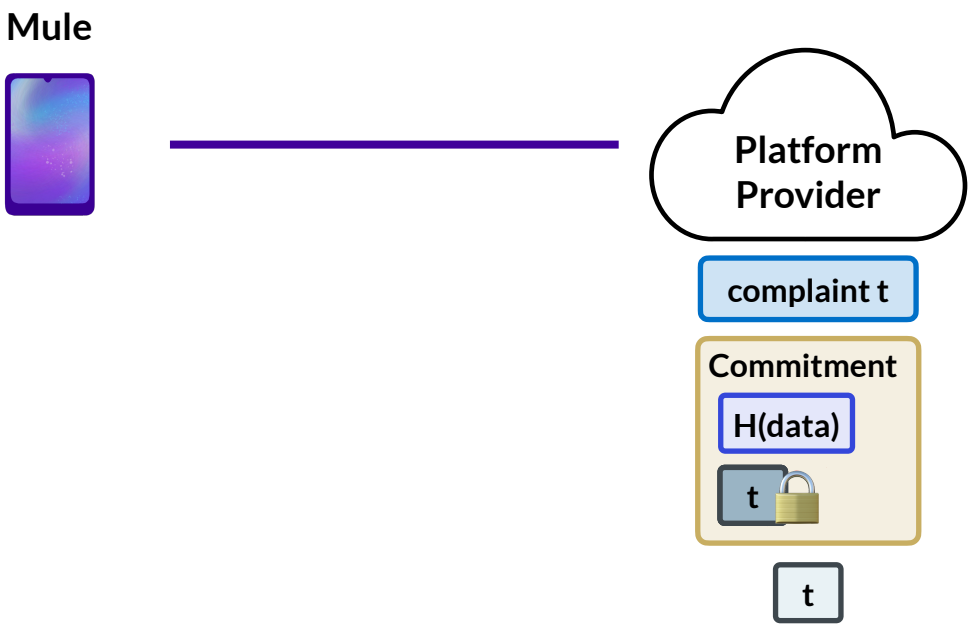
Complaints in Detail

(1) At end of epoch, mule will send a complaint token and commitment to provider



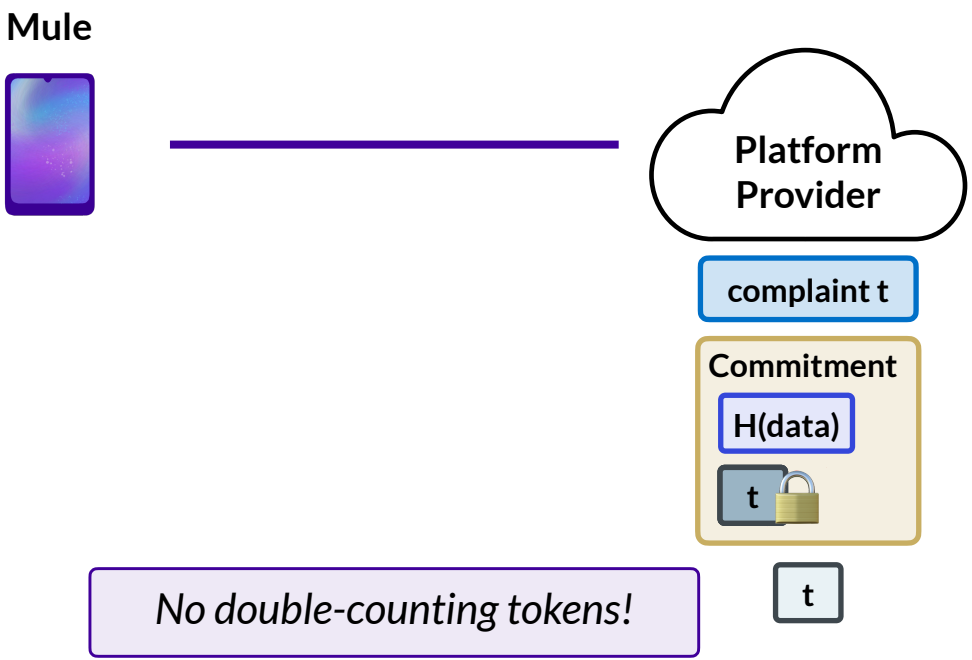
Complaints in Detail

(2) Provider will check complaint token, then decrypt and verify the delivery token, invalidating it



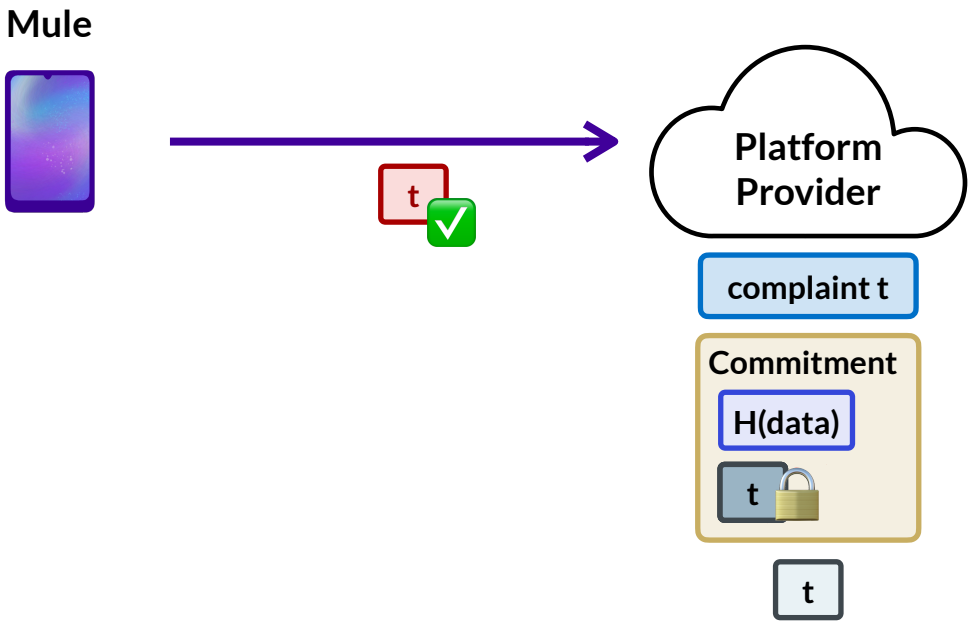
Complaints in Detail

(2) Provider will check complaint token, then decrypt and verify the delivery token, invalidating it



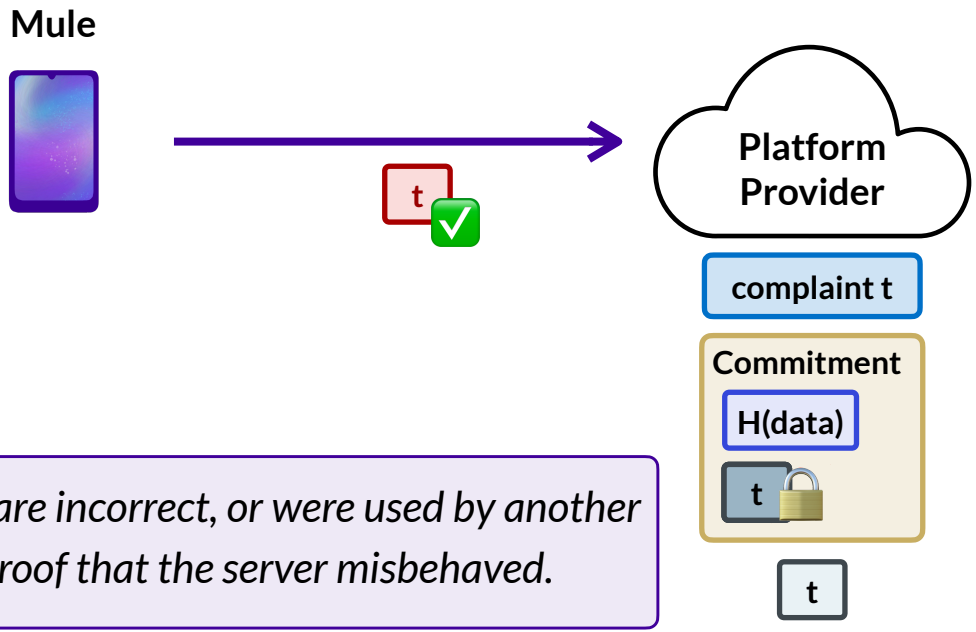
Complaints in Detail

(3) If the token was rejected by the platform provider (wrong or a duplicate), the mule sends over the signed token it got from the app server



Complaints in Detail

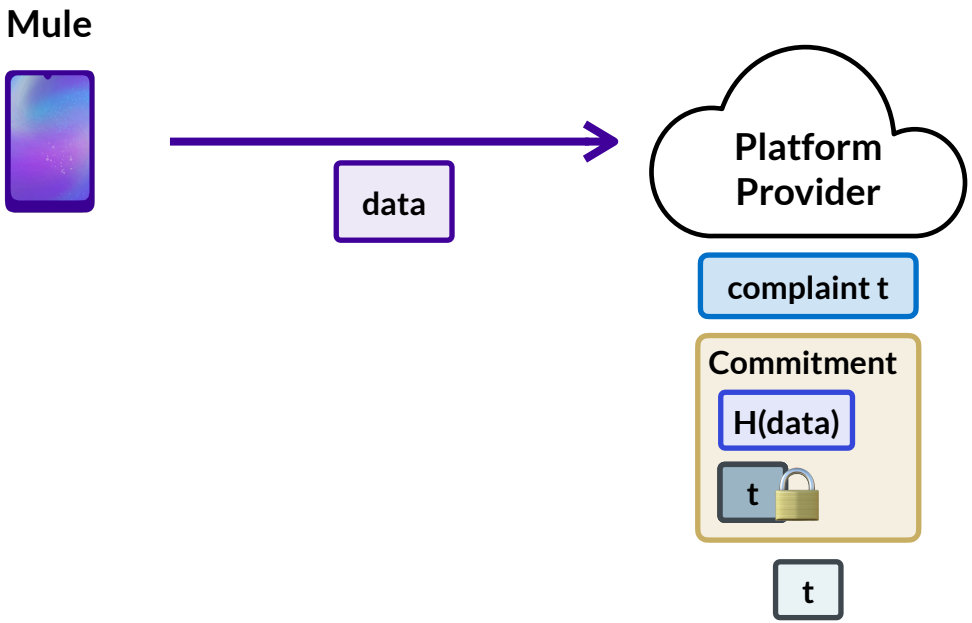
(3) If the token was rejected by the platform provider (wrong or a duplicate), the mule sends over the signed token it got from the app server



If the tokens mismatch, are incorrect, or were used by another payload, we have proof that the server misbehaved.

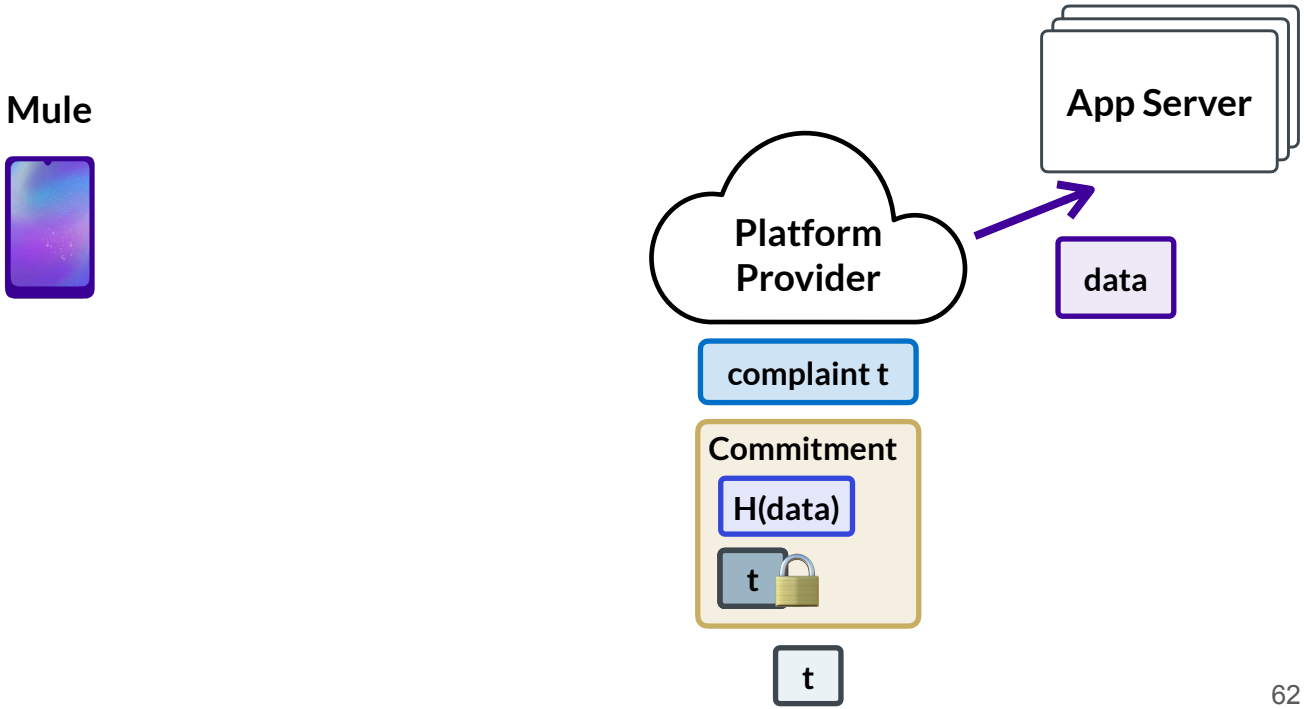
Complaints in Detail

(3b) If no token was received by the mule, it has the option of sending the original data payload matching the committed hash



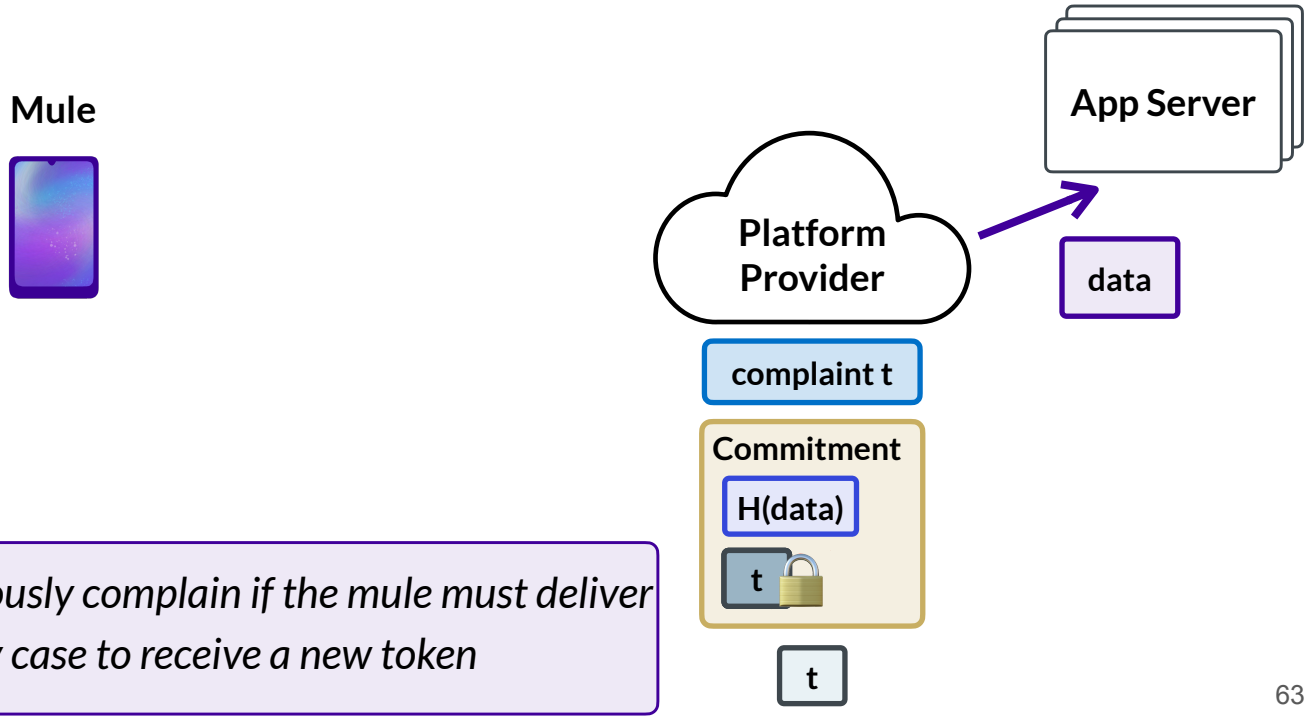
Complaints in Detail

(3b) If the data matches the committed hash, it's forwarded to the app server



Complaints in Detail

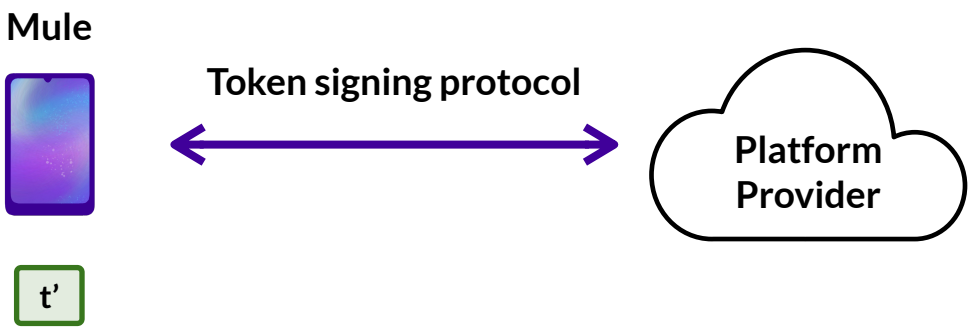
(3b) If the data matches the committed hash, it's forwarded to the app server



Little incentive to maliciously complain if the mule must deliver the data in any case to receive a new token

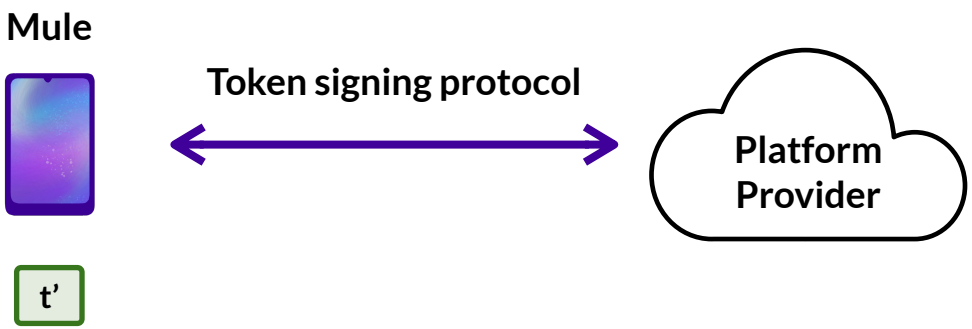
Complaints in Detail

(4) Platform provider and mule work together to blindly sign a new token!



Complaints in Detail

(4) Platform provider and mule work together to blindly sign a new token!



For duplicate tokens: first-come first-served to avoid generating more tokens than were used during delivery while incentivizing complaints

NEBULA

A backhaul system that *preserves mule privacy* while:

1. Charging for backhaul use and incentivizing mule participation

   Out-of-band accounting: app servers buy untraceable *tokens* from provider and exchange them for data.

2. Detecting mule and application server misbehavior

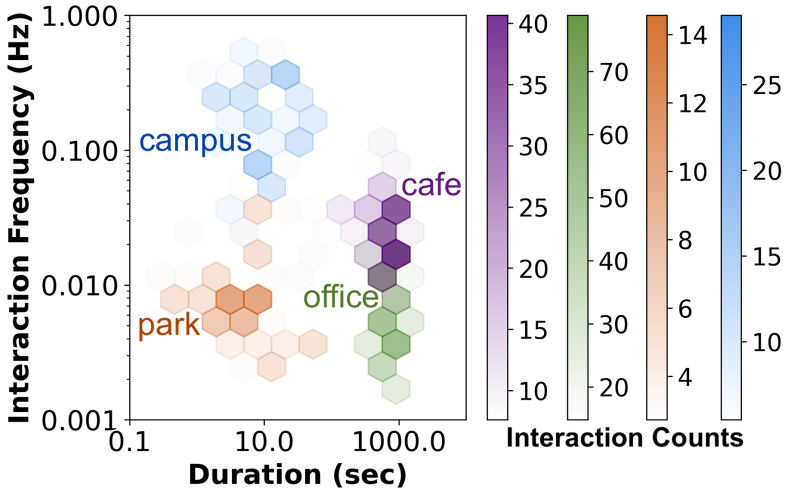
   Allow mules to submit anonymous complaints with *proof* of misbehavior to the provider.

3. Supporting millions of mules

   Nebula can process ~445k tokens/second, or **250M tokens/dollar**

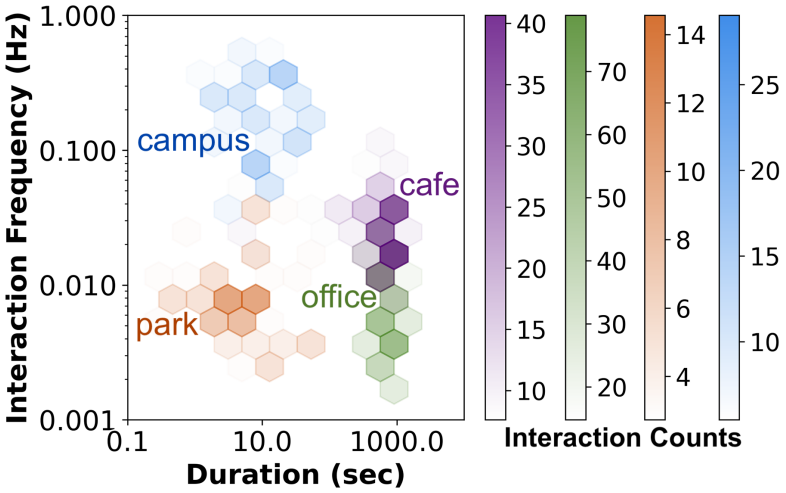
Supporting upload in varied environments

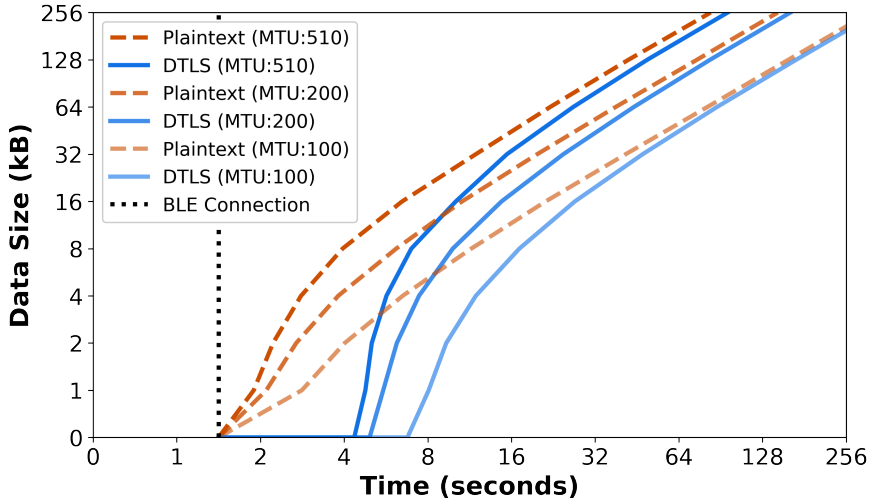
We measured likely *frequency* of sensor-mule interaction



Supporting upload in varied environments

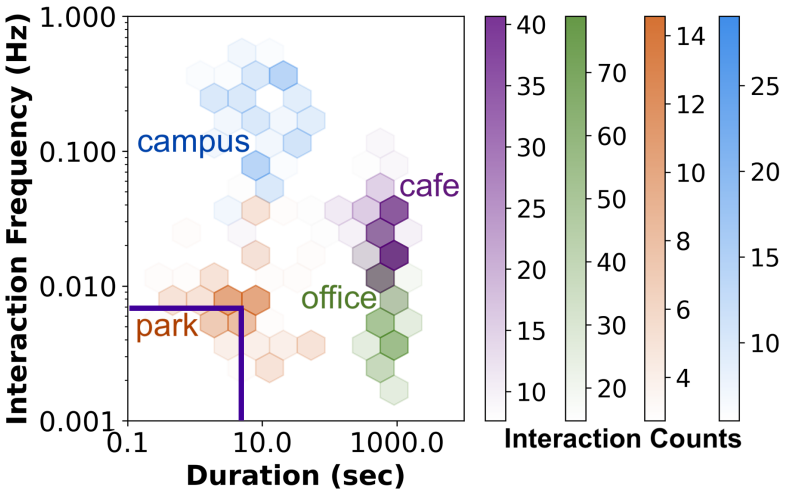
We measured likely *frequency* of sensor-mule interaction, and *duration* of data transfer

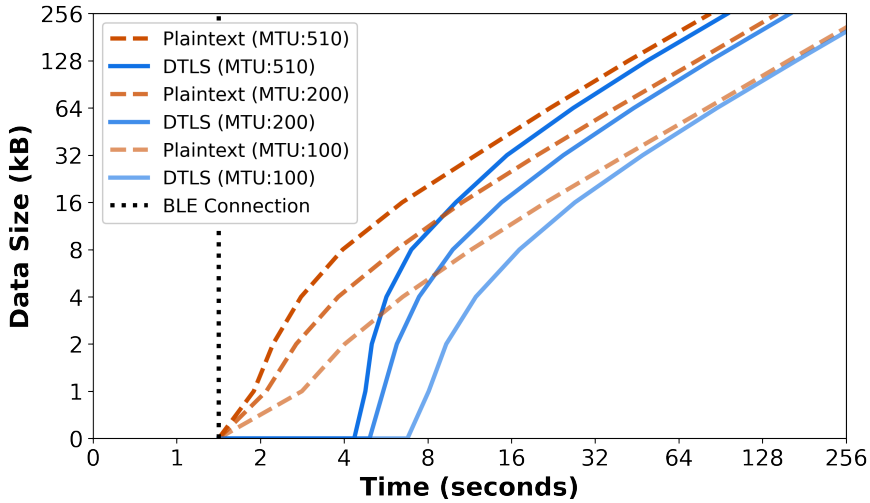




Supporting upload in varied environments

We measured likely *frequency* of sensor-mule interaction, and *duration* of data transfer

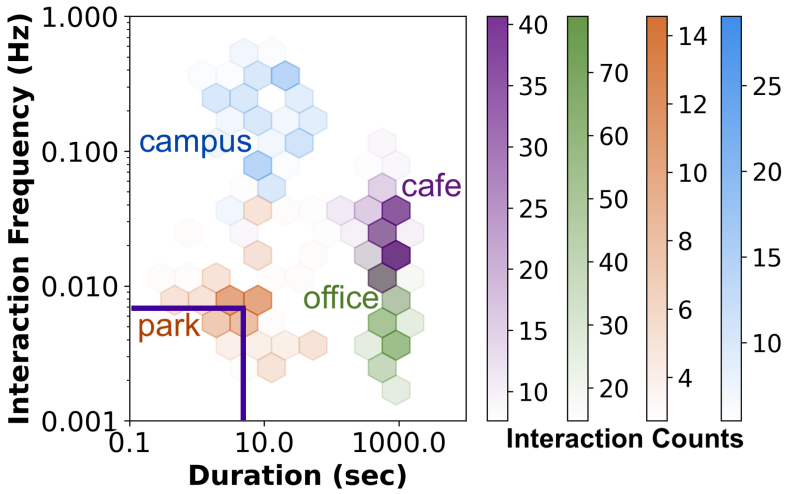


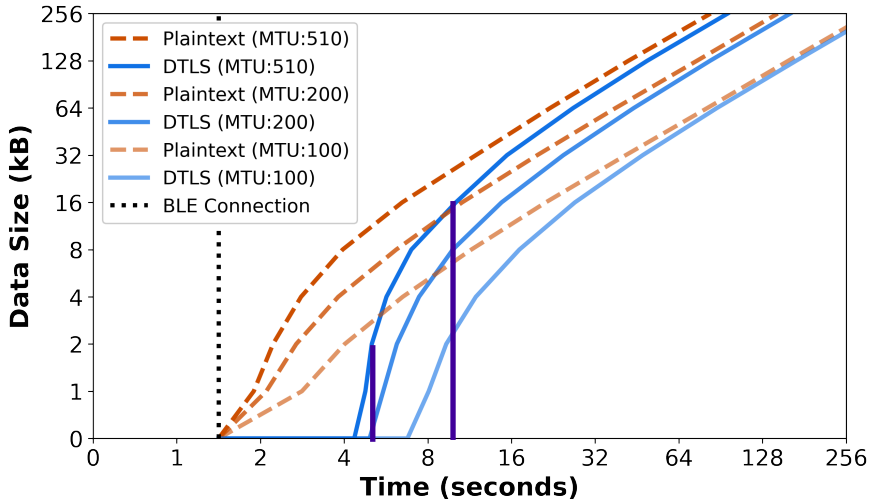


Example: in a park, we can expect a mule every few minutes

Supporting upload in varied environments

We measured likely *frequency* of sensor-mule interaction, and *duration* of data transfer

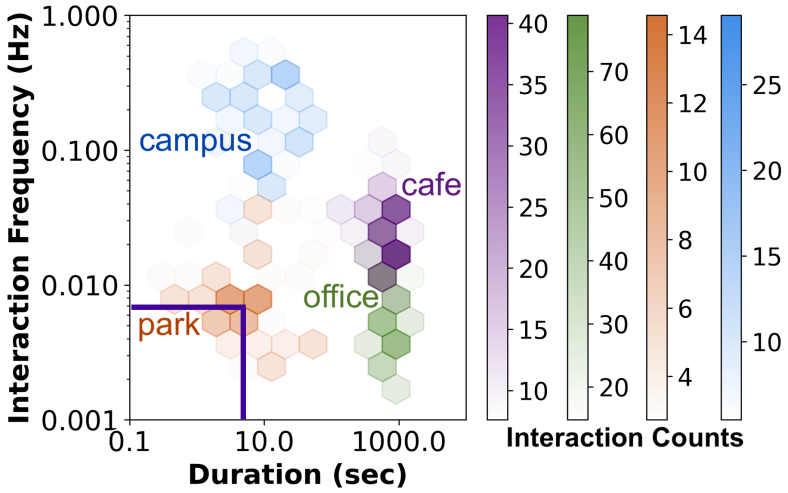


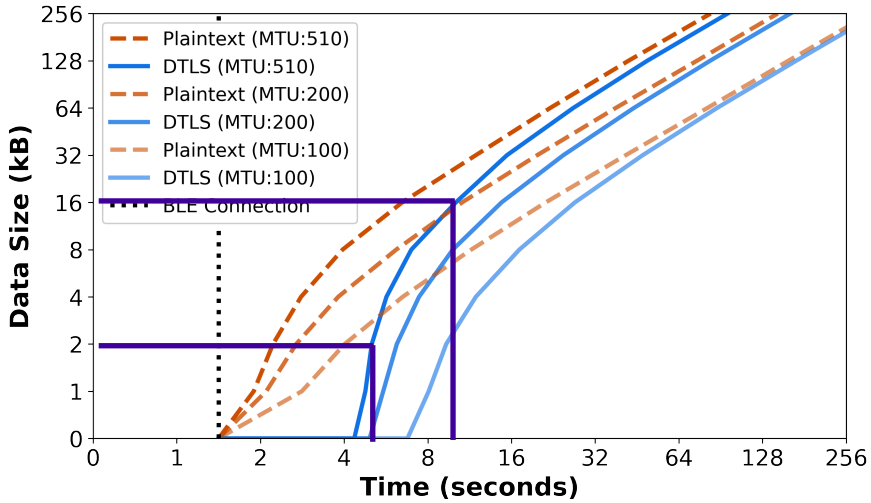


Example: in a park, we can expect a mule every few minutes,
each with a transmission window of 5 to 10 seconds,

Supporting upload in varied environments

We measured likely *frequency* of sensor-mule interaction, and *duration* of data transfer

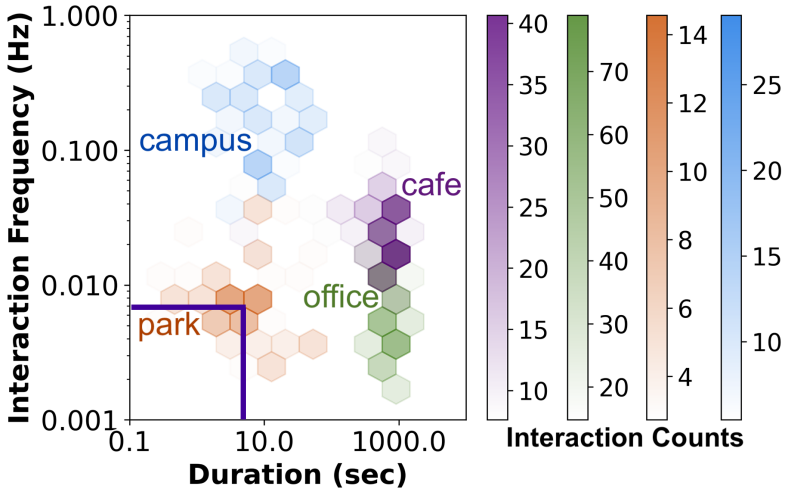


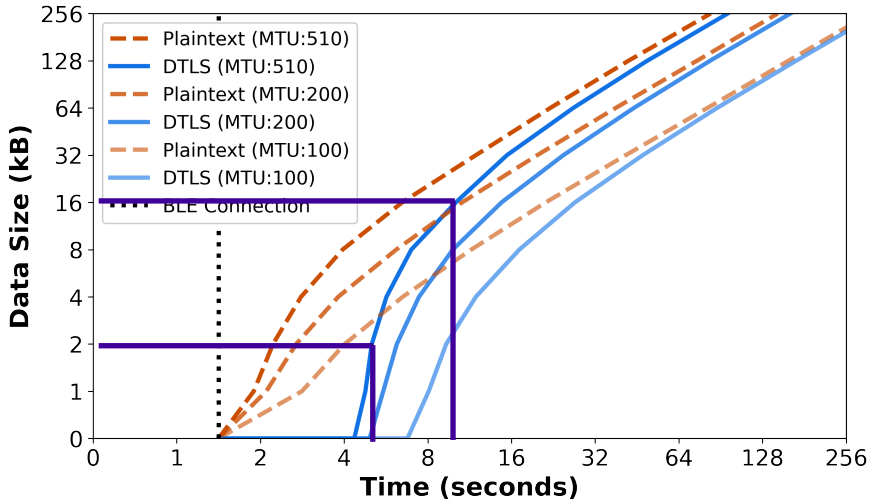


Example: in a park, we can expect a mule every few minutes,
each with a transmission window of 5 to 10 seconds, supporting 2 to 16kB payloads

Supporting upload in varied environments

We measured likely *frequency* of sensor-mule interaction, and *duration* of data transfer



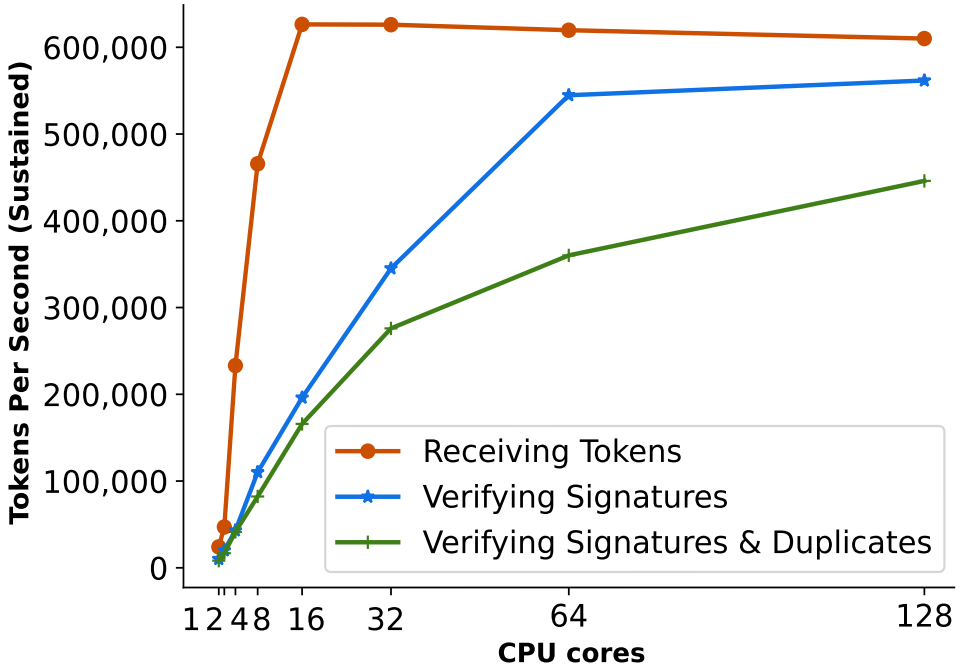


Example: in a park, we can expect a mule every few minutes,
each with a transmission window of 5 to 10 seconds, supporting 2 to 16kB payloads

Analytical Energy/Memory models in paper

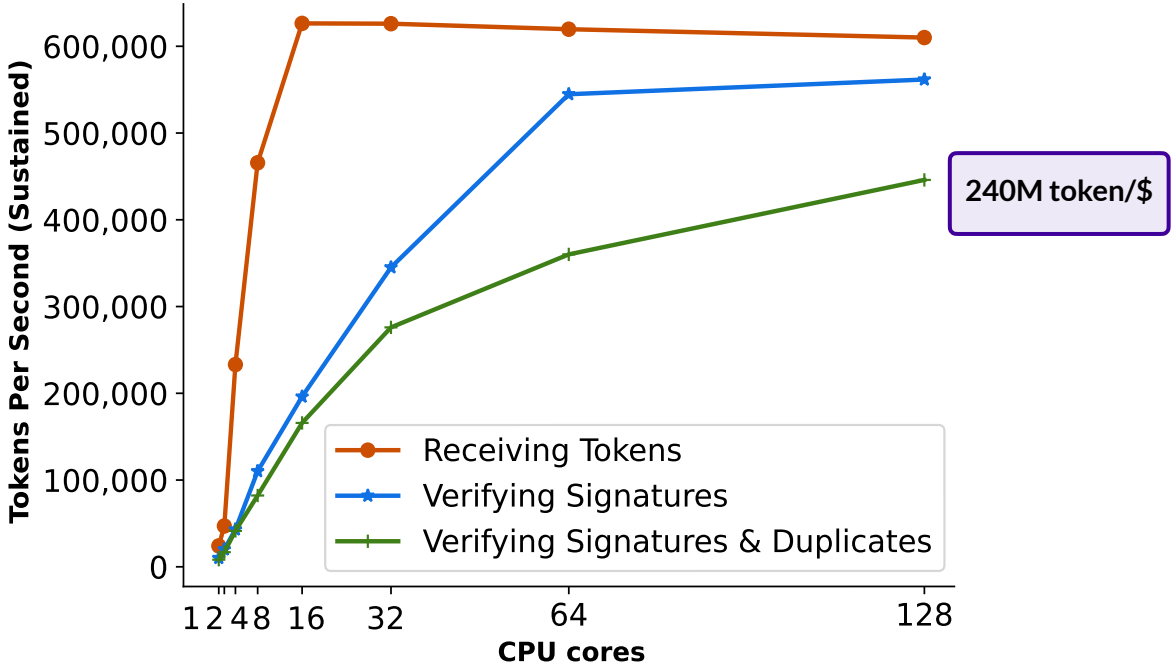
Provider performance

High-throughput database to check token validity and detect duplicates.



Provider performance

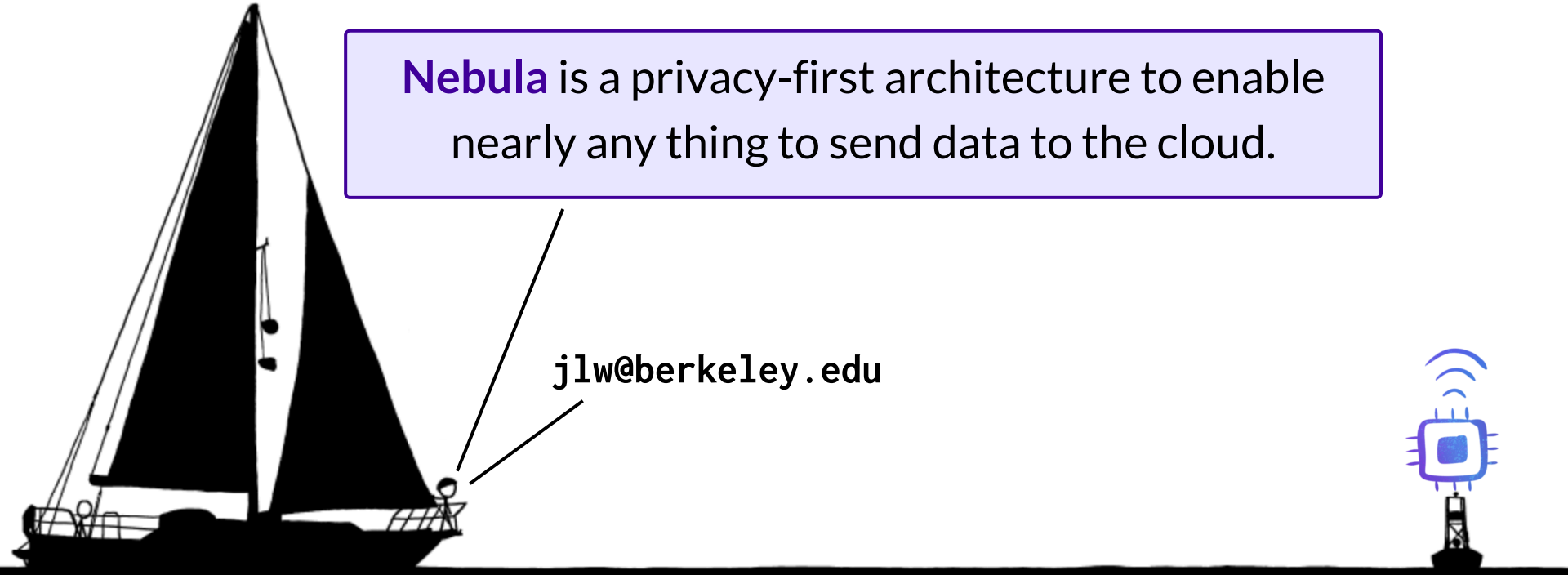
High-throughput database to check token validity and detect duplicates.



Fast enough for every person in the US to redeem each month

Find us at our poster!

Nebula is a privacy-first architecture to enable nearly any thing to send data to the cloud.



`jlw@berkeley.edu`

[xkcd 1100]