

Cloud Provider Security



CS 1660: Introduction to
Computer Systems Security

"The Cloud"

- Various features to outsource components of applications
- Faster/cheaper to build apps, easy to sync across devices ...
- In modern times, many different types of services, depending on what you want to outsource



"Types" of services

User-facing applications

- Gmail, Dropbox, OneDrive, ...

"Types" of services

User-facing applications

- Gmail, Dropbox, OneDrive, ...

Developer APIs

- Google Cloud Platform, Microsoft Azure, Amazon Web services, ...
- Various types of services, depending on what developer needs
- Eg. Block storage, databases, whole VMs, cloud functions, ...

=> {Container, VM, Database, Function call, ...} as-a-service

Featured products

Google Cloud products

[Overview](#)[Featured products](#)[AI and Machine Learning](#)[API Management](#)[Compute](#)[Containers](#)[Data Analytics](#)[Databases](#)[Developer Tools](#)[Financial Services](#)[Healthcare and Life Sciences](#)[Hybrid and Multicloud](#)[Internet of Things \(IoT\)](#)[Management Tools](#)

Compute Engine

Virtual machines running in Google's data center.

Cloud Storage

Object storage that's secure, durable, and scalable.

Cloud SDK

Command-line tools and libraries for Google Cloud.

Cloud SQL

Relational database services for MySQL, PostgreSQL, and SQL Server.

Google Kubernetes Engine

Managed environment for running containerized apps.

BigQuery

Data warehouse for business agility and insights.

Cloud CDN

Content delivery network for delivering web and video.

Dataflow

Streaming analytics for stream and batch processing.

Operations

Monitoring, logging, and application performance suite.

Cloud Run

Fully managed environment for running containerized apps.

Anthos

Platform for modernizing existing apps and building new

Cloud Functions

Event-driven compute platform for cloud services and apps.

Cloud Threats

Provider has some service/security guarantees, but not universal...

Need to consider:

- Security at the edge: defenses against outside attackers?
- Any guarantees on data protection?
 - Lost data
 - Corrupt data
 - Stolen data

Example: AWS (Amazon Web Services) “Shared responsibility model”

AWS responsibility “Security of the Cloud” - AWS is responsible for protecting the infrastructure that runs all of the services offered in the AWS Cloud. This infrastructure is composed of the hardware, software, networking, and facilities that run AWS Cloud services.

Customer responsibility “Security in the Cloud” – Customer responsibility will be determined by the AWS Cloud services that a customer selects. This determines the amount of configuration work the customer must perform as part of their security responsibilities. For example, a service such as Amazon Elastic Compute Cloud (Amazon EC2) is categorized as Infrastructure as a Service (IaaS) and, as such, requires the customer to perform all of the necessary security configuration and management tasks.

=> Different levels of security or data protection,
depending on what you pay for!

Threats from Other Cloud Tenants

Other cloud customers ("tenants") might..

- Steal your data
- Tamper with your data

Vulnerabilities with sharing of hardware, software, and network resources among clients

=> "Side channel attacks"



How is data secured in the cloud?

Cloud security fundamentals

- Encryption-at-rest:
- Encryption-in-transit:
- Encryption-in-use:

Definitions by Google, but ideas are common to all providers:
<https://cloud.google.com/compute/confidential-vm/docs/about-cvm>

Cloud security fundamentals

- Encryption-at-rest: data is encrypted when it is stored on disk
- Encryption-in-transit: data is encrypted when it is moving between two points
- Encryption-in-use: data is encrypted *while it's being processed*

Definitions by Google, but ideas are common to all providers:
<https://cloud.google.com/compute/confidential-vm/docs/about-cvm>

Cloud security fundamentals

- Encryption-at-rest: data is encrypted when it is stored on disk
=> Most cloud providers do this
- Encryption-in-transit: data is encrypted when it is moving between two points
=> Most cloud providers do this
- Encryption-in-use: data is encrypted *while it's being processed*
=> *Requires trusted execution environment*
(harder, specialized applications only)

Example: cloud storage

```
from google.cloud import storage

def write_read(bucket_name, blob_name):
    """Write and read a blob from GCS using file-like IO"""

    storage_client = storage.Client()
    bucket = storage_client.bucket(bucket_name)
    blob = bucket.blob(blob_name)

    with blob.open("w") as f:
        f.write("Hello world")

    with blob.open("r") as f:
        print(f.read())
```

Points of encryption

Encryption at rest: Uses some form of file and/or encryption (whole disk, database object, both, ...)

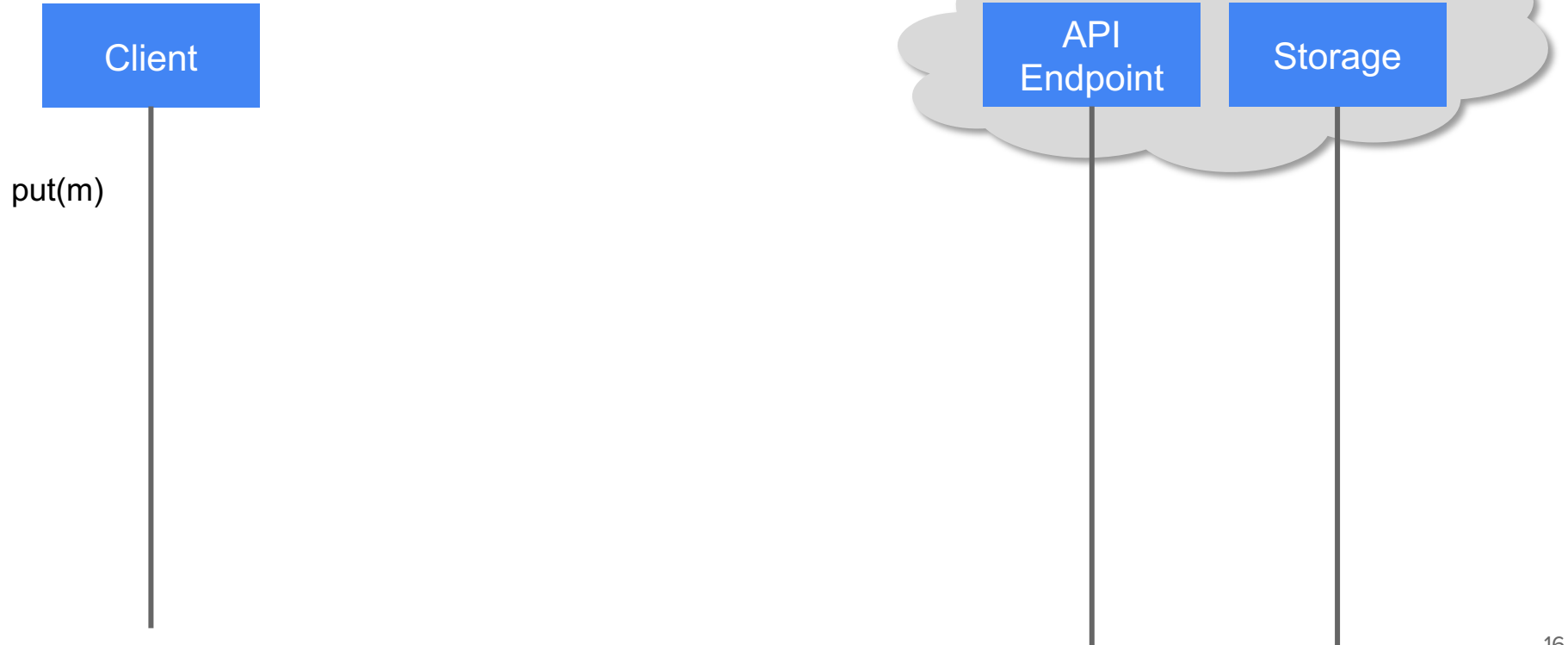
Who holds the keys?

Who holds the keys?

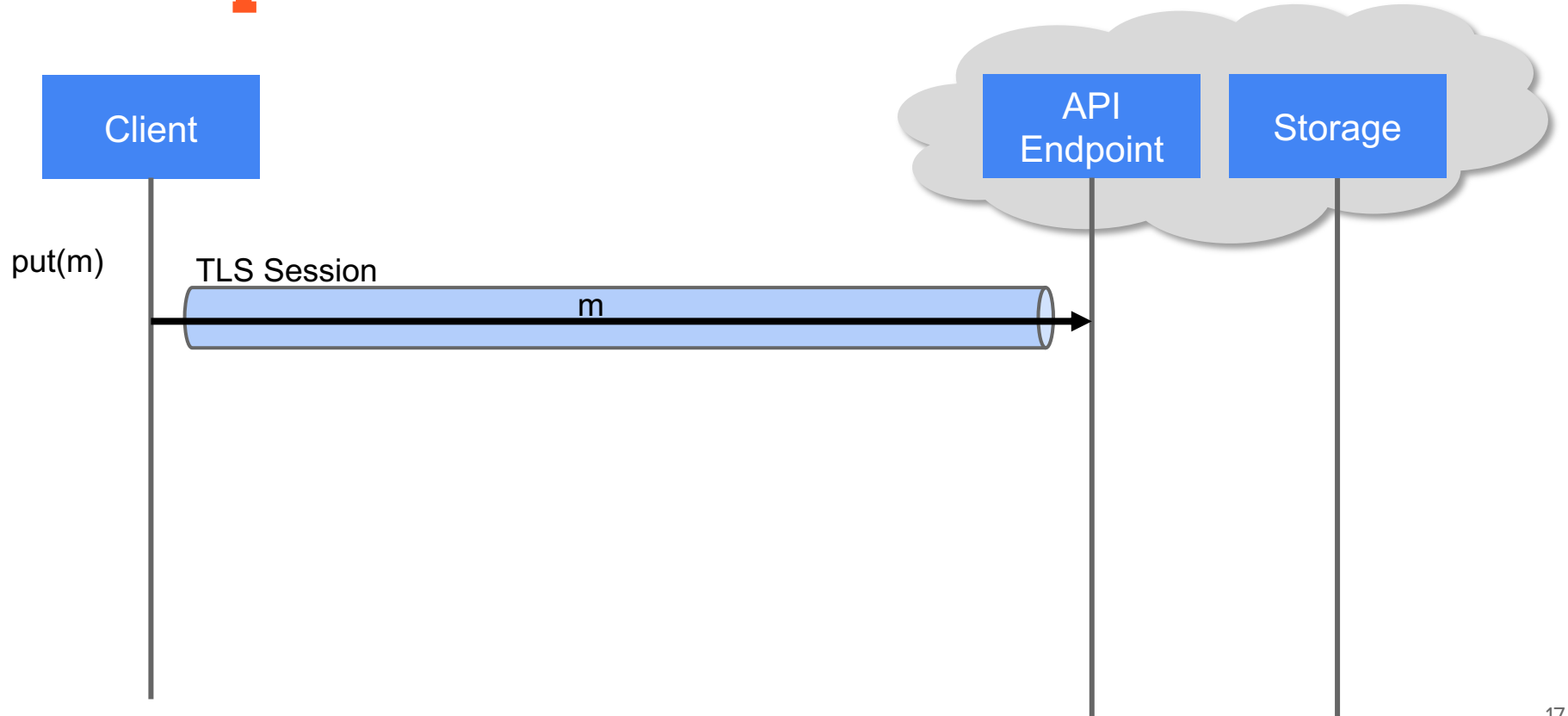
Can be configurable by customer:

- Default keys: "single" key used by provider, transparent to user
- Customer-managed keys: provider has key generation service, customer decides which objects are encrypted with keys
- Client-side keys: client application generates the keys, encrypts data before sending to provider
=> End to end encryption

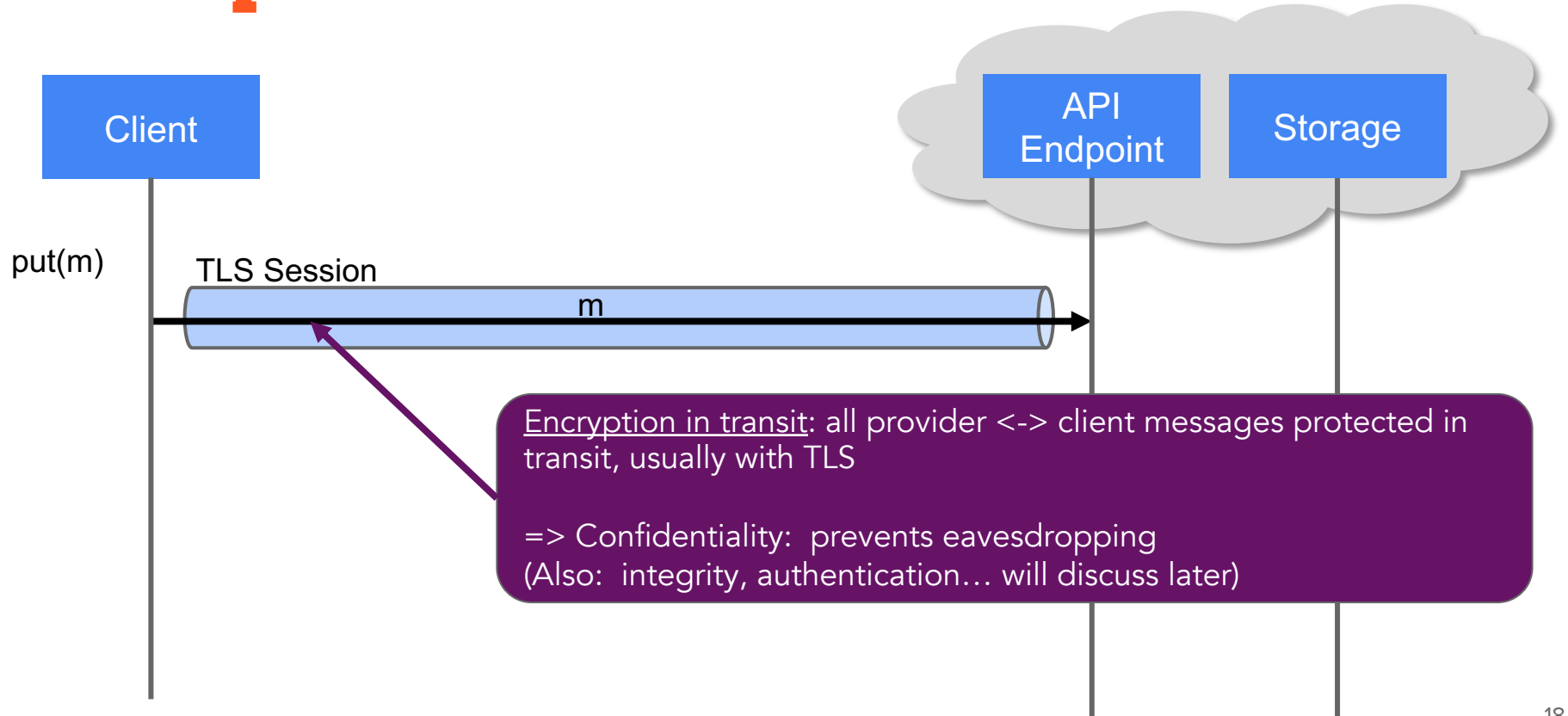
Example: Default method



Example: Default method

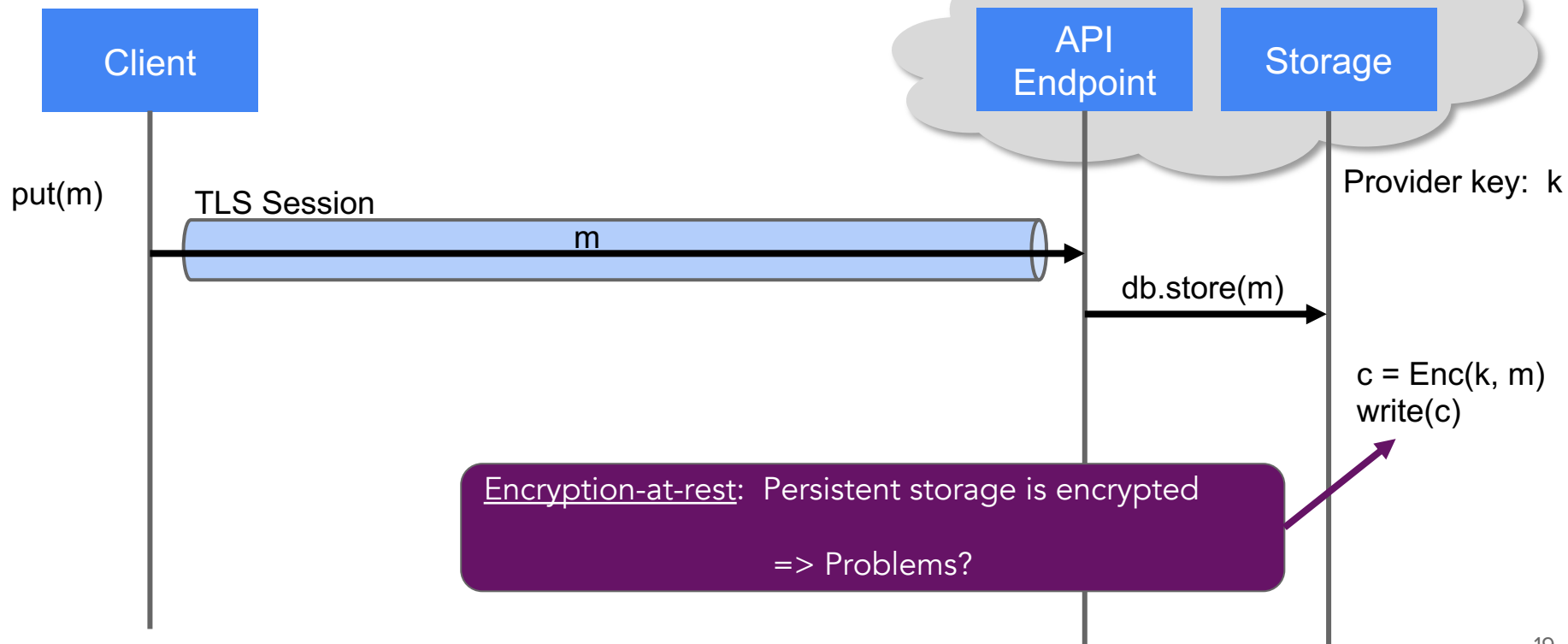


Example: Default method



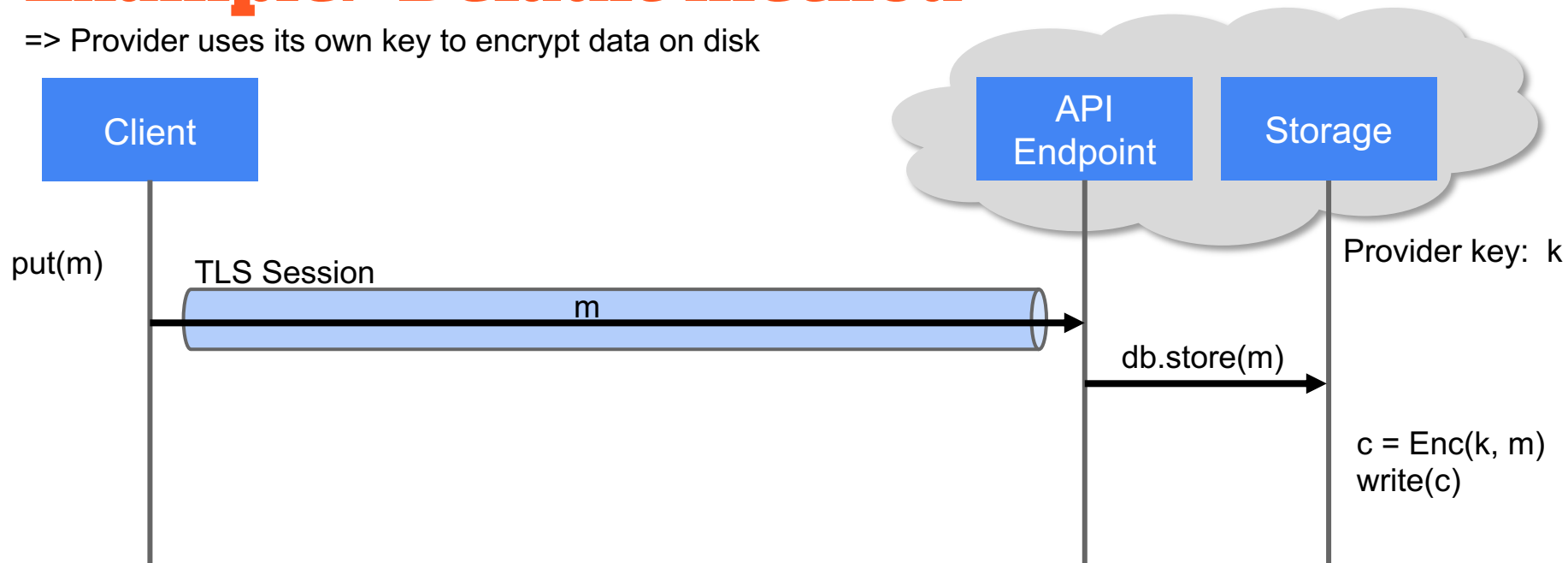
Example: Default method

=> Provider uses its own key to encrypt data on disk



Example: Default method

=> Provider uses its own key to encrypt data on disk

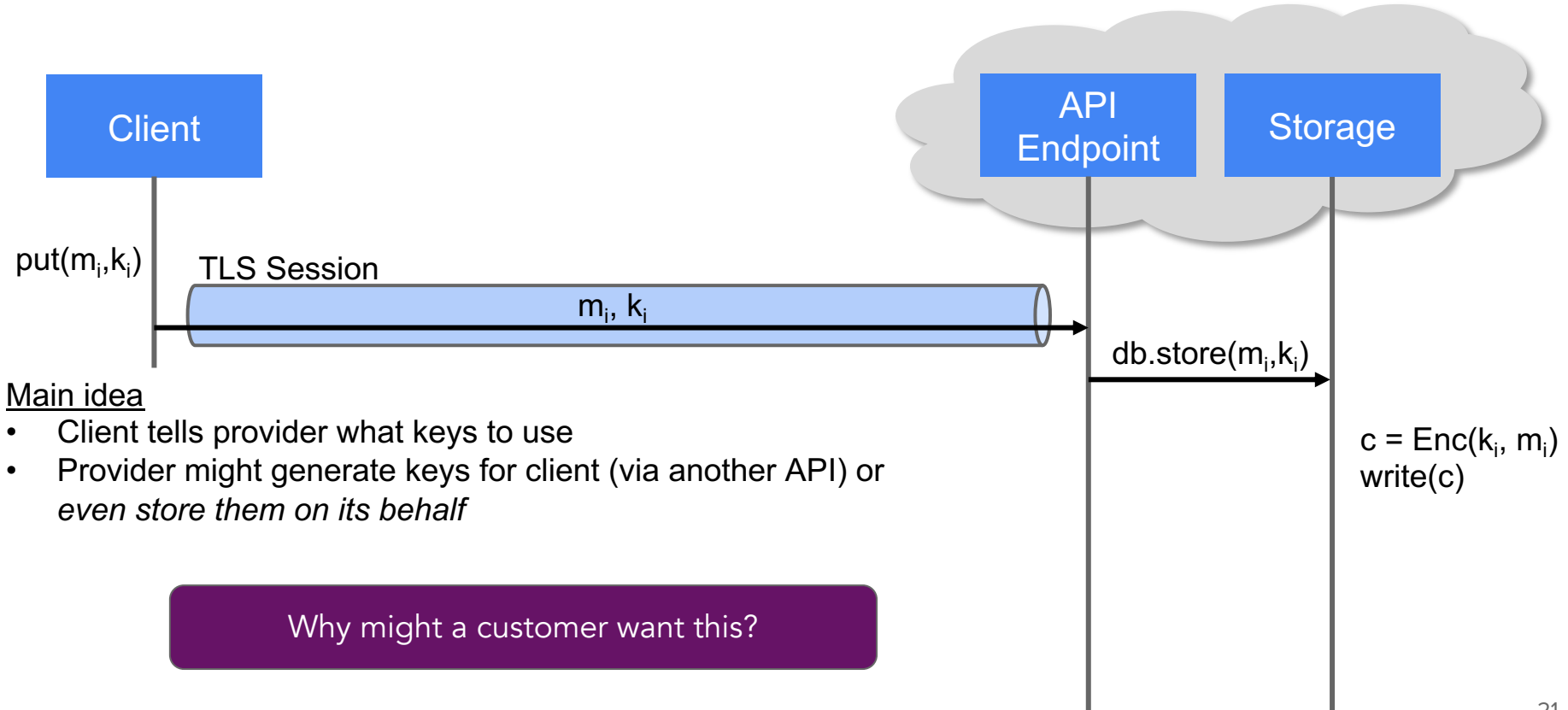


Problems?

=> Message is decrypted at some point while provider is storing it
(may be small, but nonzero)

=> Provider controls what key is used, how many systems/customers it's used on...

Example: Customer-managed keys (one way)

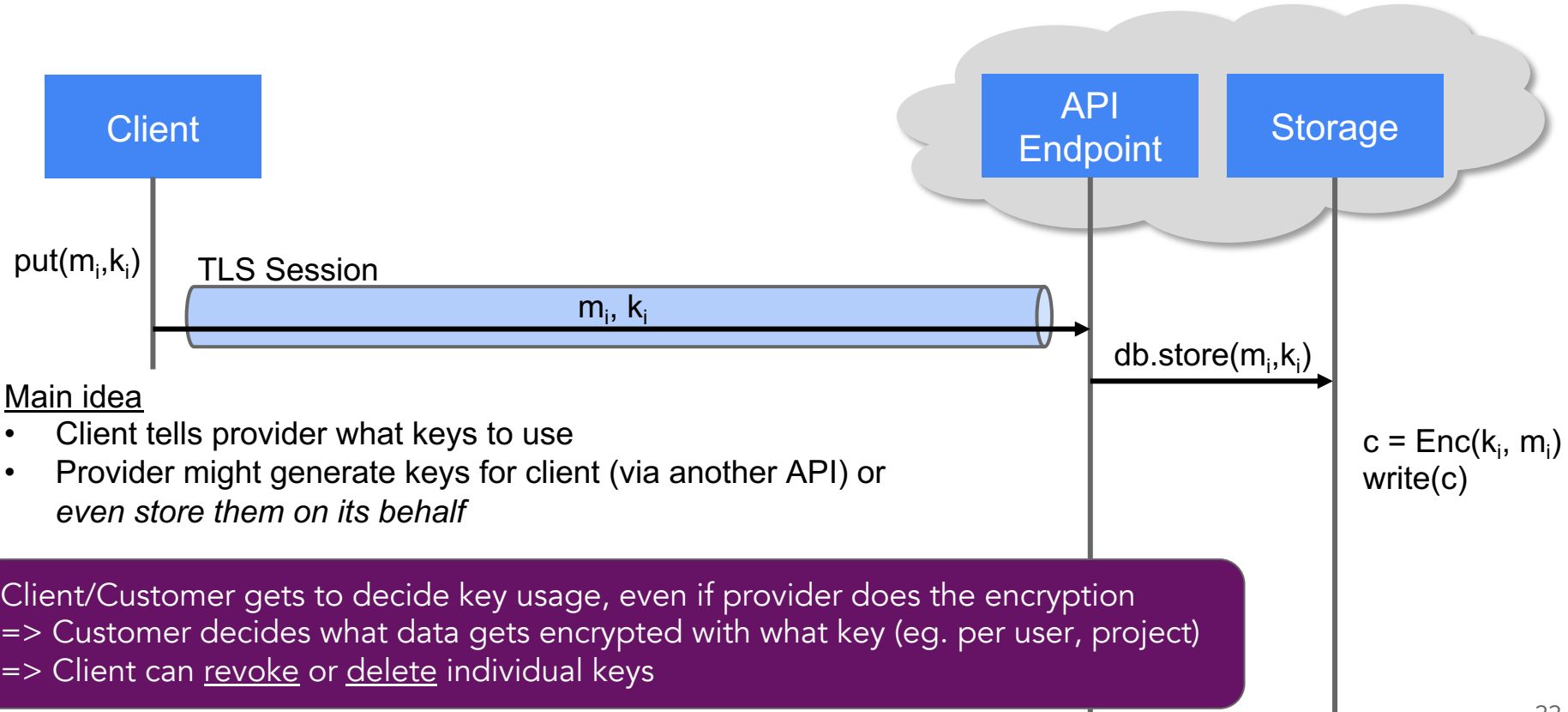


Main idea

- Client tells provider what keys to use
- Provider might generate keys for client (via another API) or *even store them on its behalf*

Why might a customer want this?

Example: Customer-managed keys (one way)

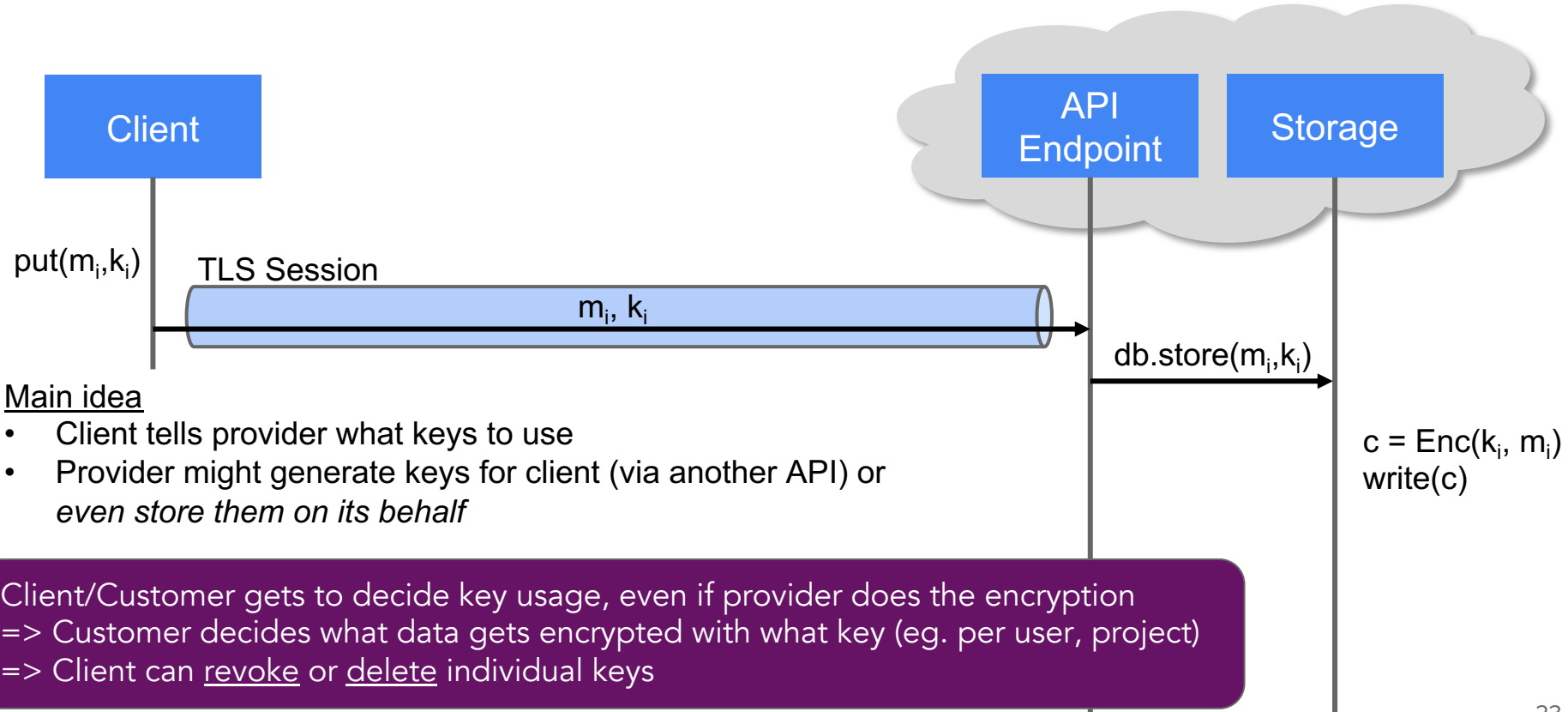


Main idea

- Client tells provider what keys to use
- Provider might generate keys for client (via another API) or *even store them on its behalf*

Client/Customer gets to decide key usage, even if provider does the encryption
=> Customer decides what data gets encrypted with what key (eg. per user, project)
=> Client can revoke or delete individual keys

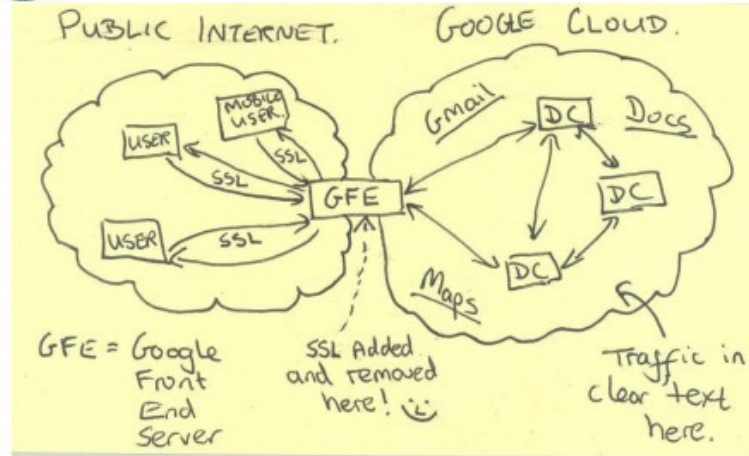
Example: Customer-managed keys (one way)



Problems?

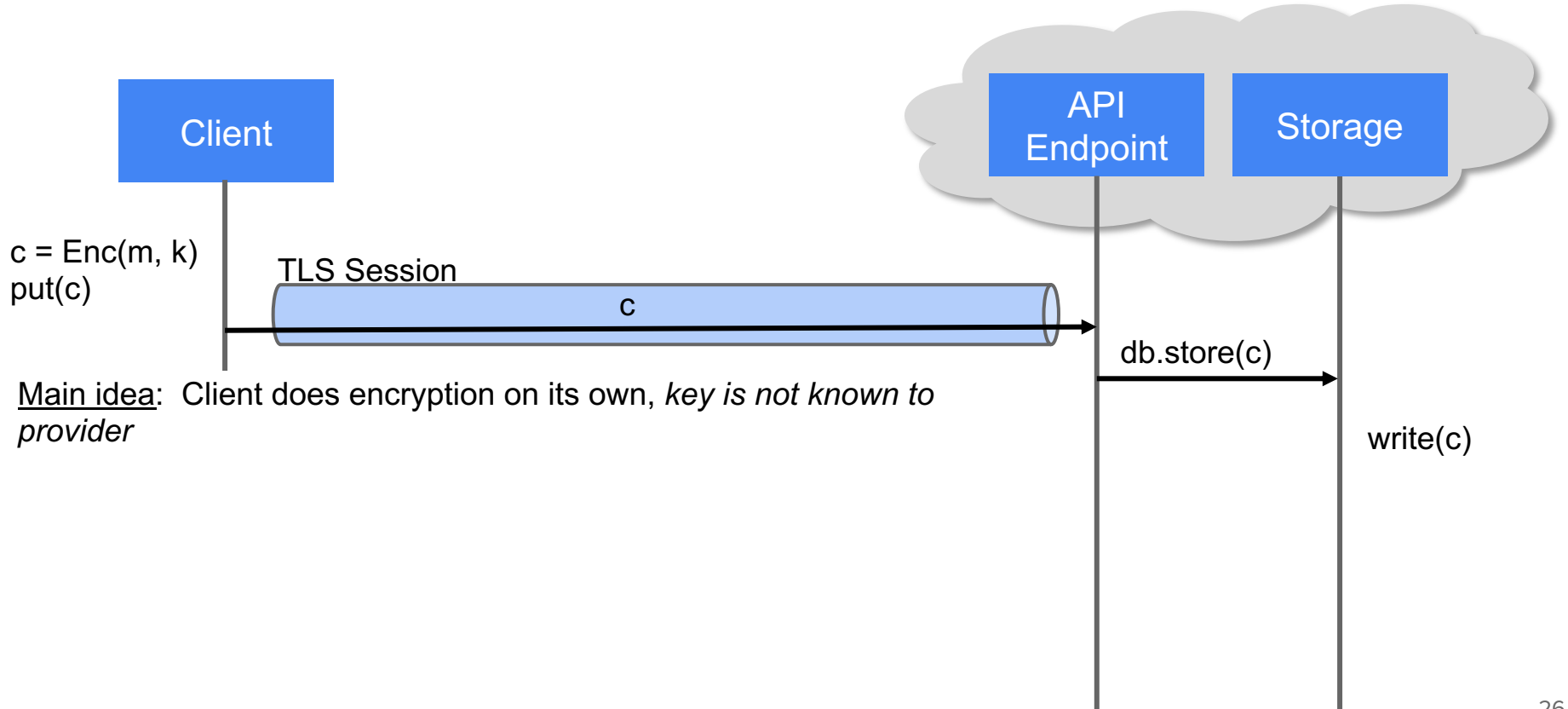


Current Efforts - Google

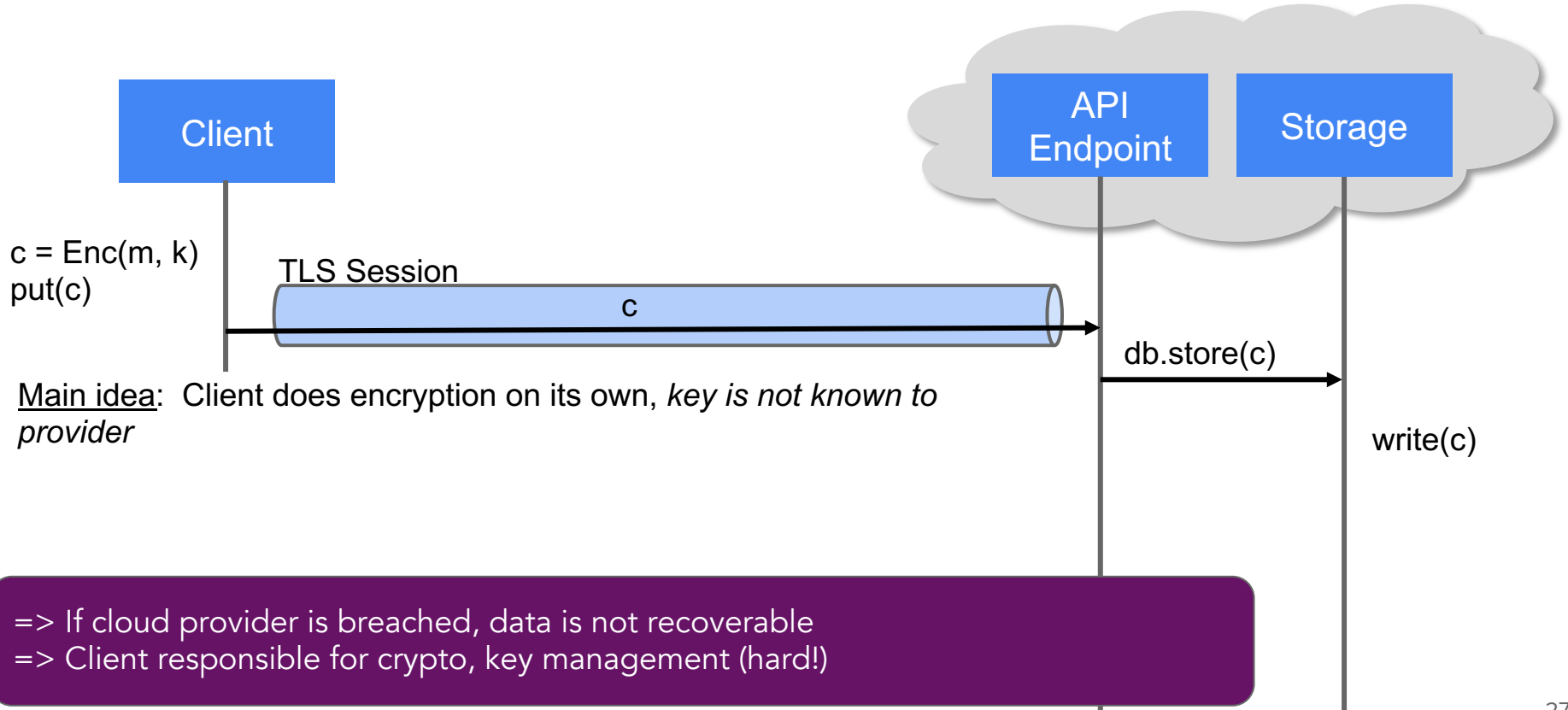


Leaked slide on NSA/GCHQ MUSCULAR program (2013): large-scale interception of (then) unencrypted cloud provider traffic Links: [1](#) [2](#) [3](#)

Another way: End to end encryption (client-side keys)



Another way: End to end encryption (client-side keys)



End-to-end encryption

- Even if cloud provider is breached, data is not recoverable
- Client is more complex
 - Needs to manage keys
 - Cryptographic operations must happen client-side
 - => You'll do this in Dropbox! (we give you the crypto library, though!)

End-to-end encryption

- Even if cloud provider is breached, data is not recoverable
- Client is more complex
 - Needs to manage keys
 - Cryptographic operations must happen client-side
 - => You'll do this in Dropbox! (we give you the crypto library, though!)

End-to-end encryption is starting to become more common in cloud systems, and user-facing applications

=> In general, cloud providers are critical for modern business => incentive to provide security features to meet customer demand (corporate policy, compliance with regulations, etc.)

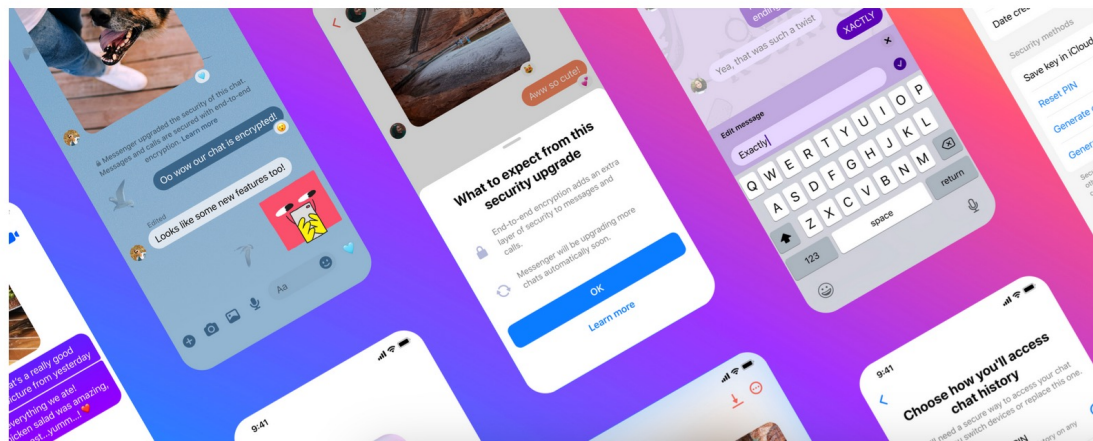
[← Back to Newsroom](#)

Messenger

Launching Default End-to-End Encryption on Messenger

December 6, 2023

By Loredana Crisan, Head of Messenger



Messages & Privacy

You can register your phone number or Apple ID to send iMessages. Apple retains limited information about the use of iMessage, such as whether your device is eligible to use iMessage, for up to 30 days.

- Messages are backed up in iCloud and encrypted if you enable iCloud Backup or Messages in iCloud.
- iMessage is end-to-end encrypted. The phone number or email address you use is shown to the people you contact, and you can choose to share your name and photo.
- Apple retains limited information about the use of iMessage, such as whether your device is eligible to use iMessage, for up to 30 days.

Messages is an app that allows users to communicate via SMS, MMS, iMessage, and Apple Messages for Business. iMessage is an Apple service that sends messages to other iOS devices (with iOS 5 or later), iPadOS devices, visionOS devices, Macs (with OS X 10.8 or later), and Apple Watches. These messages don't count against your messaging plan. Messages sent via iMessage can include photos, videos, and other information.

We designed iMessage to use end-to-end encryption, so there's no way for Apple to decrypt the content of your conversations when they are in transit between devices. Attachments you send over iMessage (such as photos or videos) are encrypted so that no one but the sender and receiver(s) can access them.

[Contact Us](#)[Start free](#)

GO TO PAGE



Compliance offerings

To help you with compliance and reporting, we share information, best practices, and easy access to documentation. Our products regularly undergo independent verification of security, privacy, and compliance controls, achieving certifications against global standards to earn your trust. We're constantly working to expand our coverage.

Filter by:

Countries

Regions

Industries

Focus area

Additional filters

United States X [Clear all](#)

UNITED STATES

Export
Administration



UNITED STATES

FedRAMP



UNITED STATES

FERPA (U.S.)

Advanced Data Protection for iCloud

Starting with iOS 16.2, iPadOS 16.2 and macOS 13.1, you can choose to enable Advanced Data Protection to protect the vast majority of your iCloud data, even in the case of a data breach in the cloud.

With Advanced Data Protection, the number of data categories that use end-to-end encryption rises to 23 and includes your iCloud Backup, Photos, Notes, and more. The table below lists the additional data categories that are protected by end-to-end encryption when you enable Advanced Data Protection.

Data categories and encryption

The table below provides more detail on how iCloud protects your data when using standard data protection or Advanced Data Protection.

Data category	Standard data protection		Advanced Data Protection	
	Encryption	Key storage	Encryption	Key storage
iCloud Mail (1)	In transit & on server	Apple	In transit & on server	Apple
Contacts (2)	In transit & on server	Apple	In transit & on server	Apple

Extra info: Cloud Storage Integrity

May be useful for CS1620/CS2660 component of Dropbox (but not required)

Cloud Storage Integrity

- Alice outsources her files to Bob (cloud storage provider)
- How can Alice check whether a file downloaded from Bob has not been corrupted?

Did the Cloud Corrupt my Files?

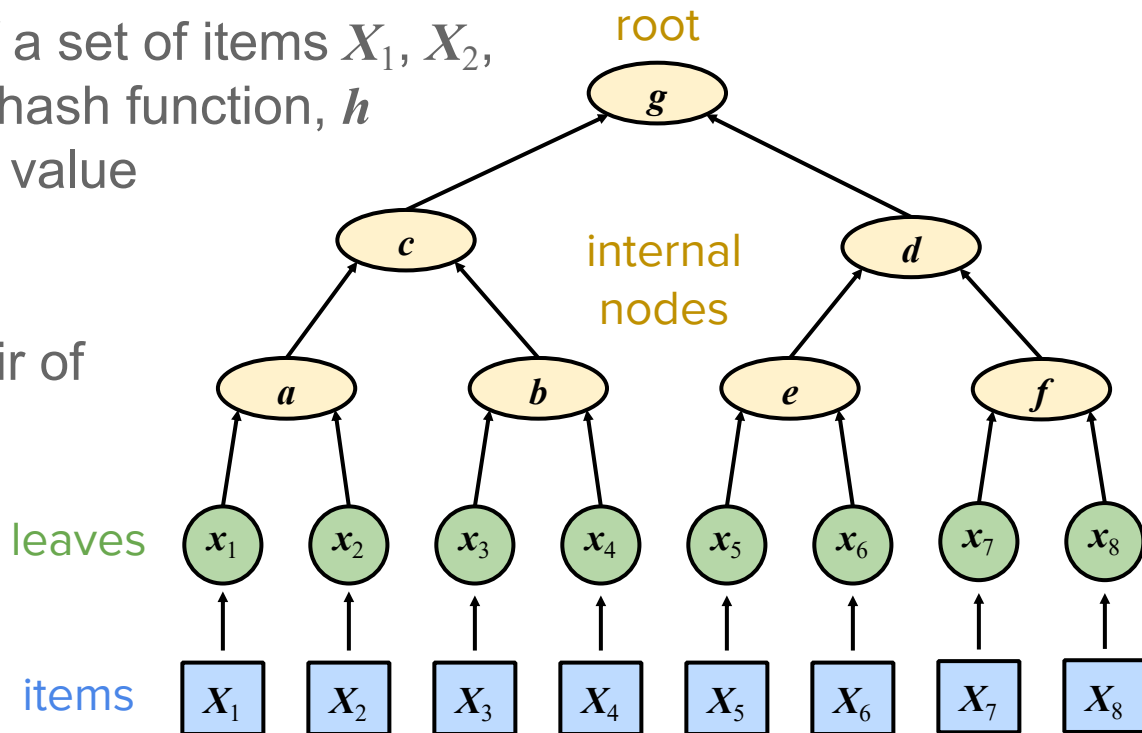
- Alice outsources her files to Bob (cloud storage provider)
- How can Alice check whether a file subsequently downloaded from Bob has not been corrupted?
- Basic solution
 - Alice computes and keeps cryptographic hashes of her files
 - Upon download of a file from Bob, Alice checks the hash of the downloaded file against the stored hash
- Alice detects any change in the file with overwhelming probability

More Efficient Integrity Verification




- Storing n hashes is more efficient than storing n files for Alice
- However, the asymptotic space requirement for Alice is still $O(n)$
- Improved solution
 - Using a cryptographic hash function, Alice builds a **Merkle tree** over her files, where leaves store hashes of files and internal nodes store hierarchically computed hash values
 - Alice keeps the **root hash** of the Merkle tree (and discards the rest of the tree)
 - The asymptotic space requirement for Alice is now only $O(1)$

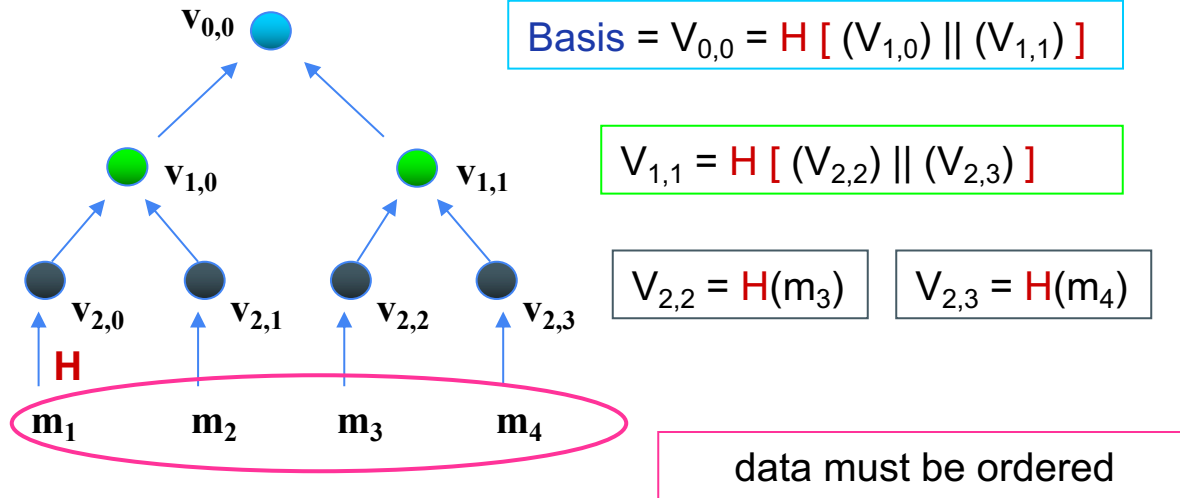
What is a Merkle Tree

- Binary tree built on top of a set of items X_1, X_2, \dots using a cryptographic hash function, h
- Each node stores a hash value
- Leaf: hash of item
 - $x_i = h(X_i)$
- Internal node: hash of pair of values at children
 - $a = h(x_1 x_2)$
 - $b = h(x_3 x_4)$
 - $c = h(a b)$
 - ...



Hash Tree (Merkle): building

-  Basis: authenticated tree root
-  Authentication structure
-  Data hash



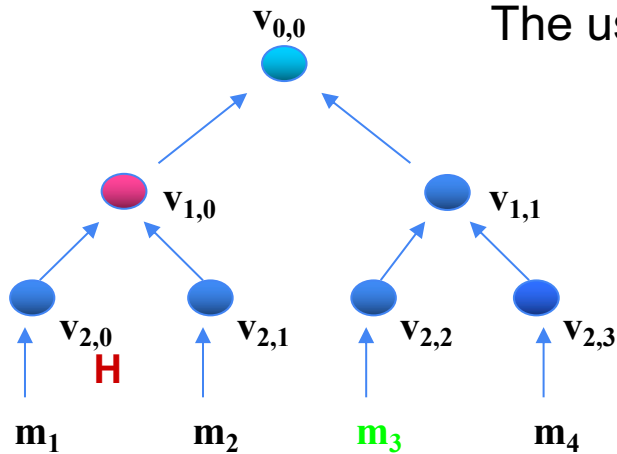
H is a Hash function

Hash Tree (Merkle): test

A user would verify data authenticity of m_3

Authenticated answer is made by: $m_3, V_{2,3}, V_{1,0}$

And from the Basis signed by a CA.



The user can verify if m_3 is authentic:

$$V_{2,2} = H(m_3)$$

$$V_{1,1} = H(V_{2,2} || V_{2,3})$$

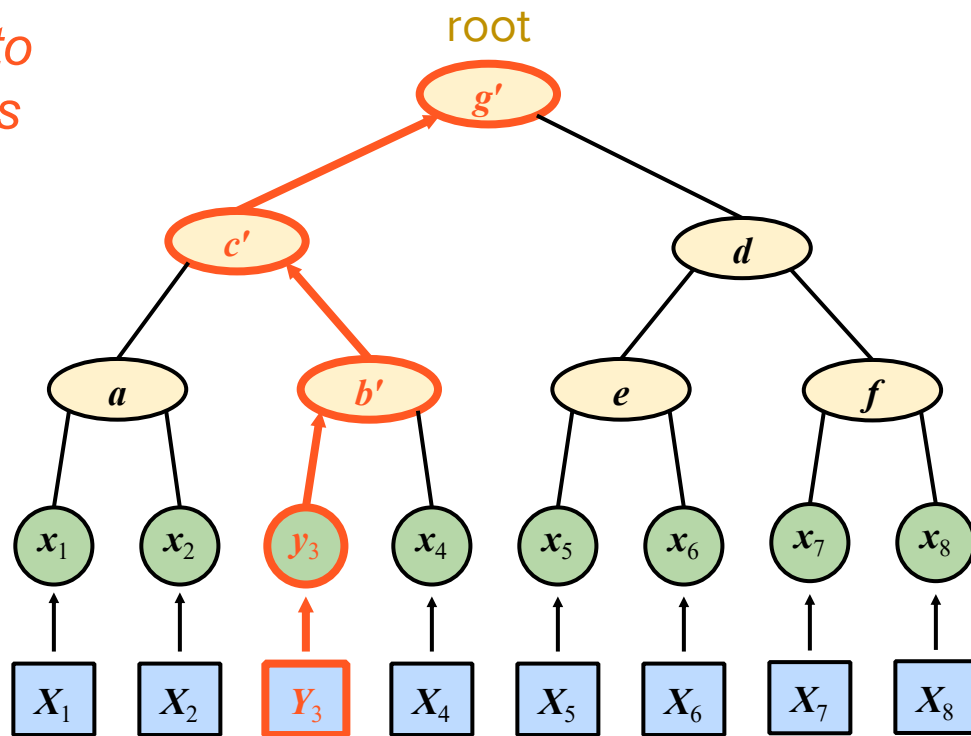
$$V_{0,0} = H(V_{1,0} || V_{1,1})$$

If Basis == $V_{0,0}$ then m_3 is authentic

Integrity Property of a Merkle Tree

Given a Merkle tree, it is unfeasible to modify any nonempty subset of items without modifying also the root hash

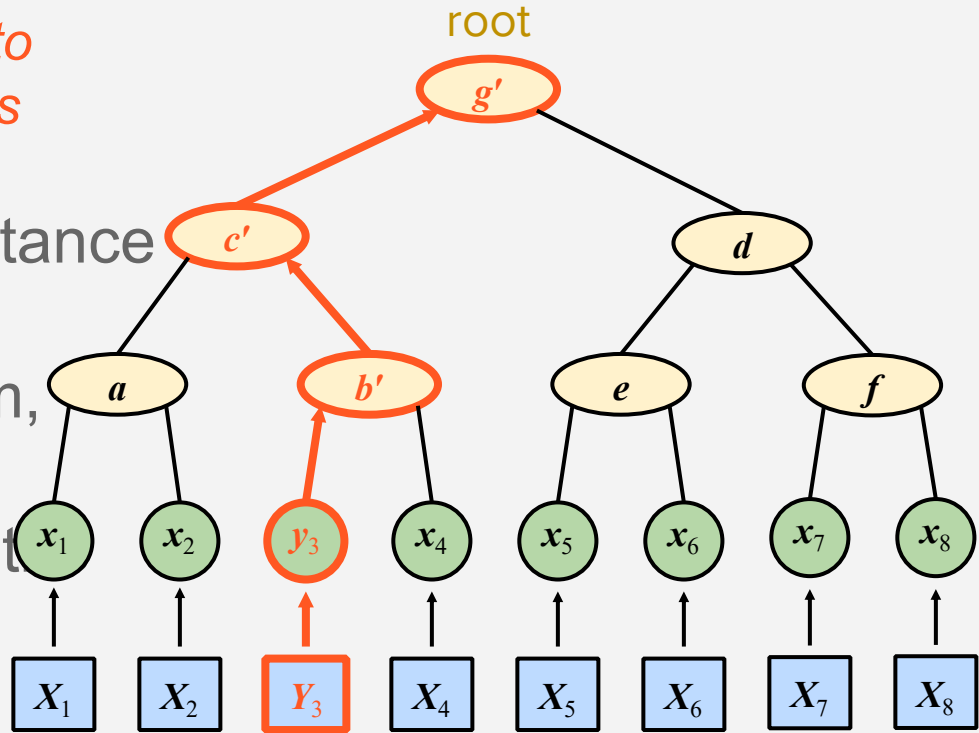
- Why?



Integrity Property of a Merkle Tree

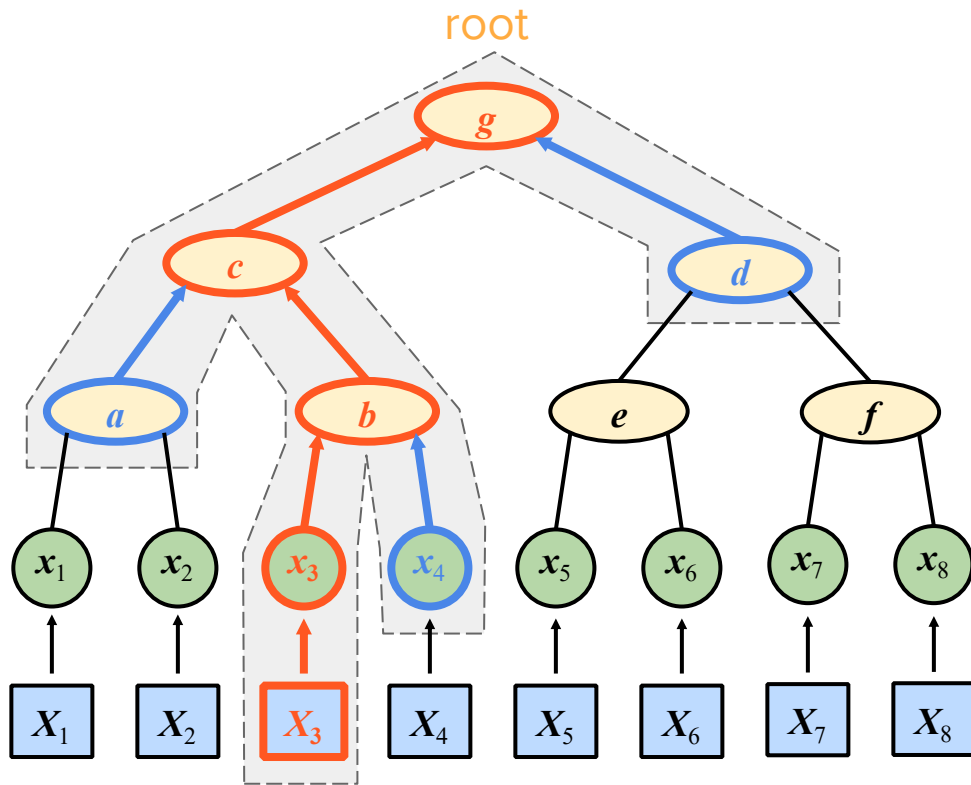
Given a Merkle tree, it is unfeasible to modify any nonempty subset of items without modifying also the root hash

- Follows from collision resistance of the hash function
- Suppose we modify an item, say X_3 , into Y_3
- The nodes from the leaf to the root change value, else we found a collision of the hash function



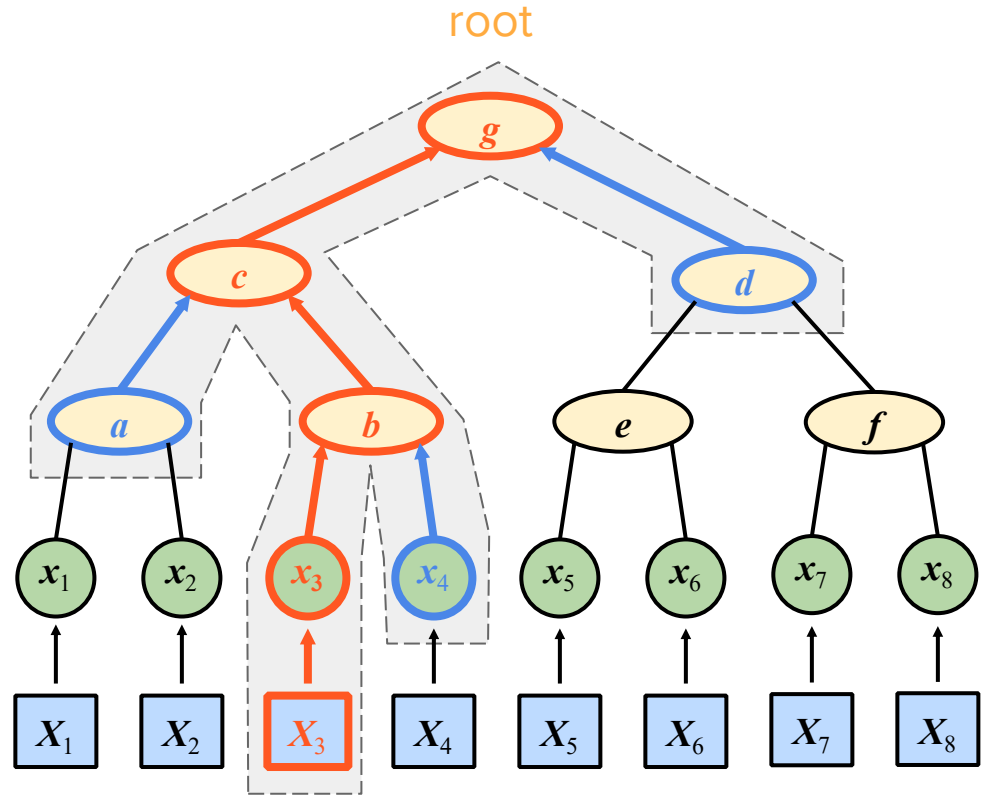
Proof of an Item in a Merkle Tree

- A Merkle tree provides a proof that an item is in the set:
 - sequence of hash values and L/R (left/right) indicators
- To build the proof for an item:
 - Start at the leaf and go up to the root
 - At each node, pick hash value and side of sibling node
- Example: proof for X_3
 - (x_4, R) , (a, L) , (d, R)



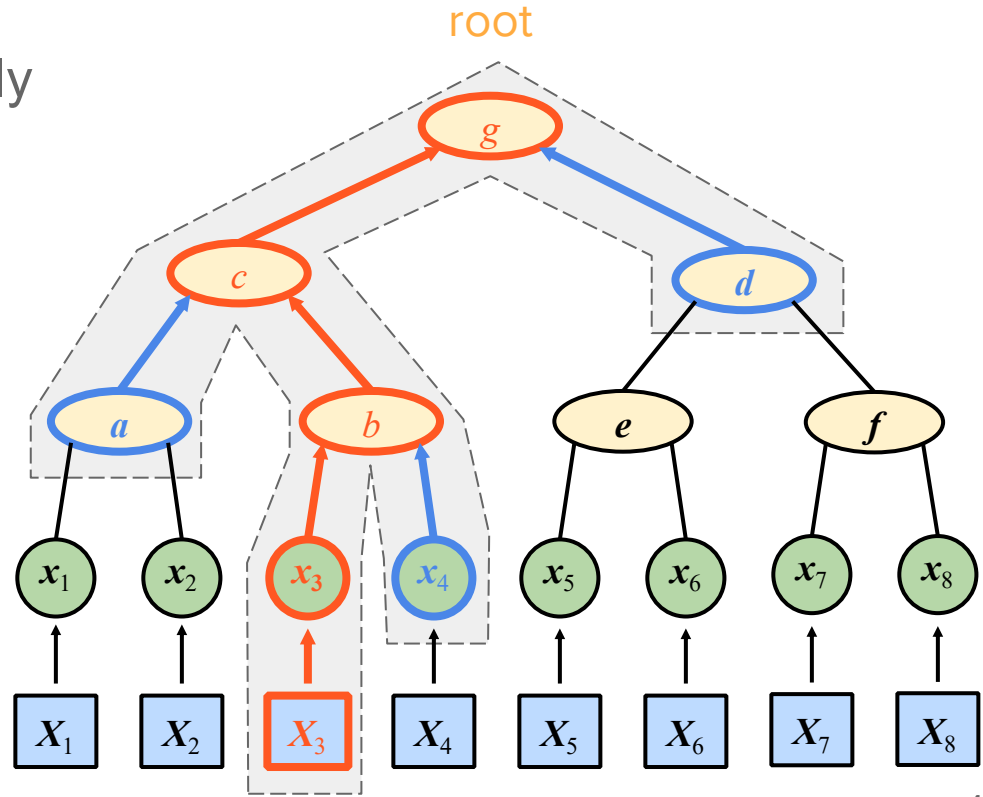
Proof of an Item in a Merkle Tree (cont.)

- Proof verification:
 - Compare root hash with hash derived from item and proof
- Proof for X_3 :
 - $(x_4, R), (a, L), (d, R)$
- Verification:
 - $g = h(h(a, h(x_3, x_4)), d)$
- The integrity property ensures one cannot forge proofs
- Proofs have size proportional to the logarithm of the number of items



Proof of an Item in a Merkle Tree (cont.)

- The proof of an item is essentially a chain of hashes
- L/R indicators denote order of hashing at each node of the chain
- Size of proof (number of values) is height of tree, i.e., logarithm in base two of number of items
- Examples:
 - 8 items, proof size 3
 - 1,024 items, proof size 10
 - 1 M items, proof size 20
 - 1 B items, proof size 30



Who is Merkle?

Ralph C. Merkle

A pioneer of modern
cryptography

<http://www.merkle.com/>



Source: <http://www.merkle.com/>

Selected Related Work

- **Authenticated Data Structures**
 - Two party ADS model where the client maintains a proof of validity for the data [Goodrich Tamassia 03]
 - Authenticated skip list embedded in a relational table [Di Battista Palazzi 07]
- **Storage check**
 - Efficient integrity checking of untrusted network storage [Heitzmann, Palazzi, Papamanthou, Tamassia 08]
- **Set operations**
 - Query Racing: Fast Completeness Certification of Query Results [Palazzi, Pizzonia, Pucacco 10]