Instant Auditing of Cloud Storage Access without Cumulative Evidence

Advicer : Gwan-Hwan Hwang
Student : Wei-Chih Chien

NTNU CSIE CCLAB

2016.03.22

Outline

1. Scenario
2. Real-time Auditing Schemes
   - Instant Auditing of Cloud Storage Access by Cache Partial Merkle tree
   - My Method
3. Protocol Detail
   - Flowchart
   - Download & Upload
   - Audit
4. Experimental Results

Outline

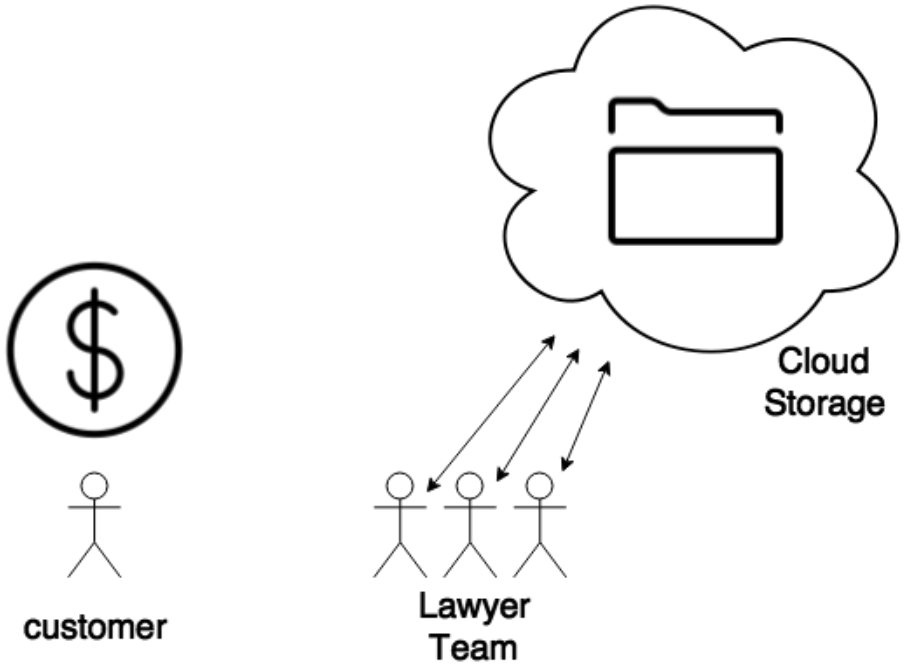
1. Scenario
2. Real-time Auditing Schemes
   - Instant Auditing of Cloud Storage Access by Cache Partial Merkle tree
   - My Method
3. Protocol Detail
   - Flowchart
   - Download & Upload
   - Audit
4. Experimental Results

Scenario

Law Office

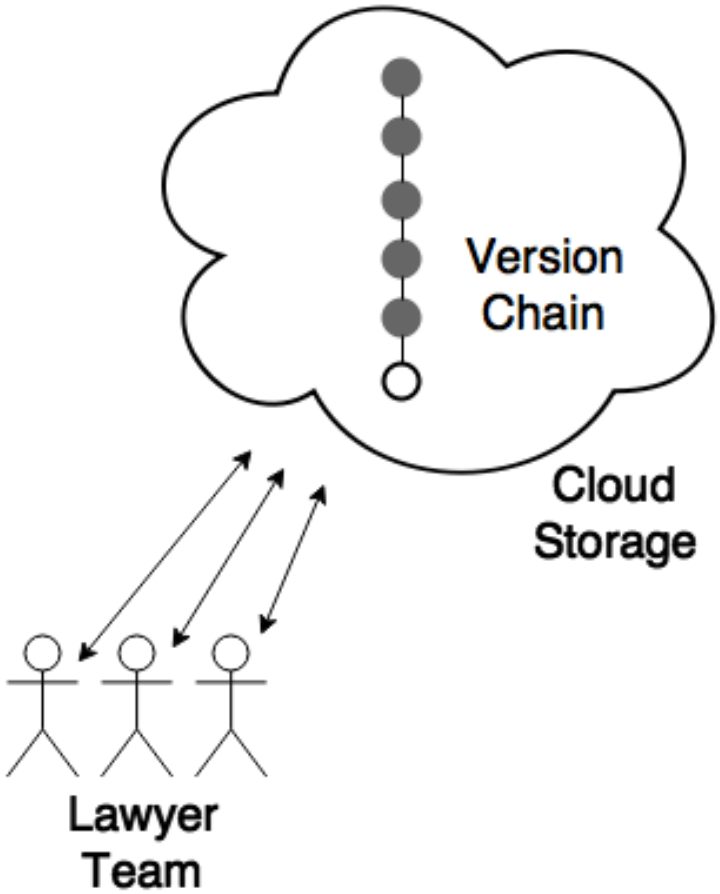
Advantage of Cloud Storage:

- Save money on hardware cost
- Easily access files
- ...



Scenario (CON'T)

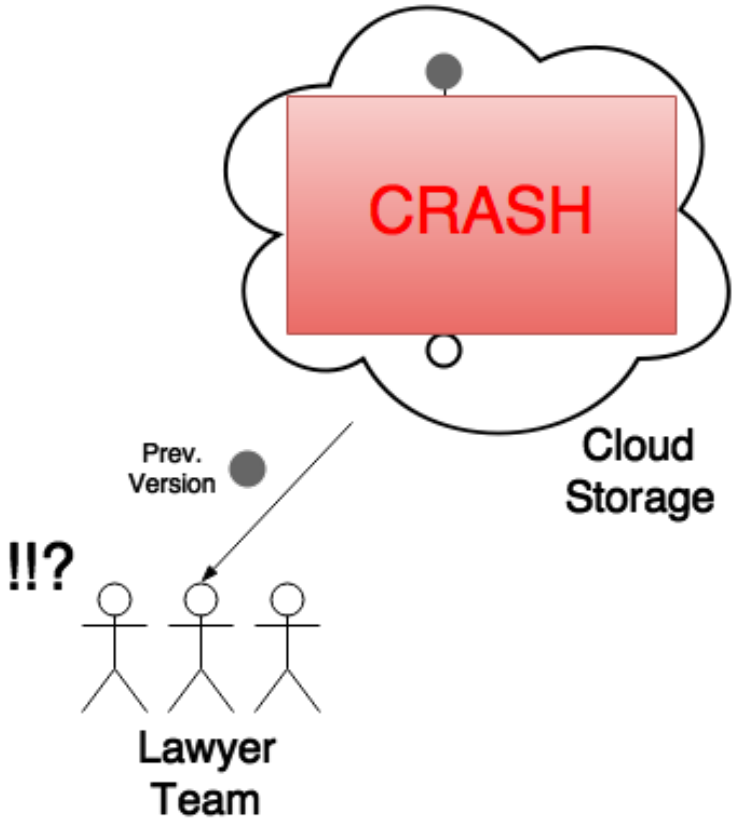
Version Control



Always Get the Latest Version of Files.

Scenario (CON'T)

What if...

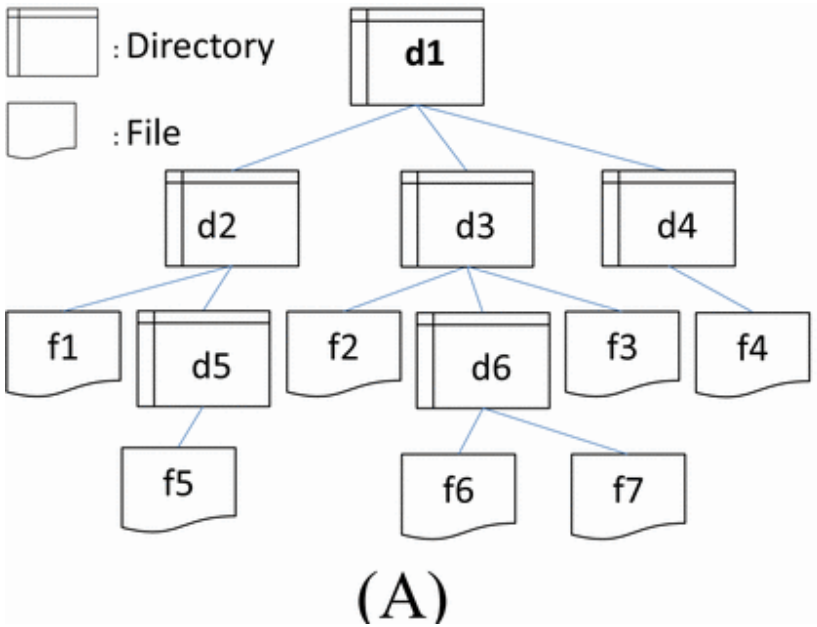


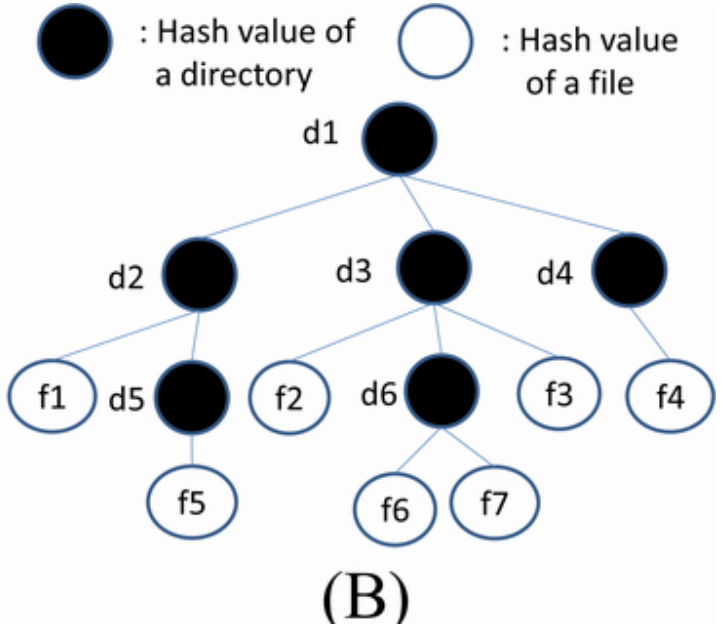
To request compensation, Cryptographic Proof is necessary

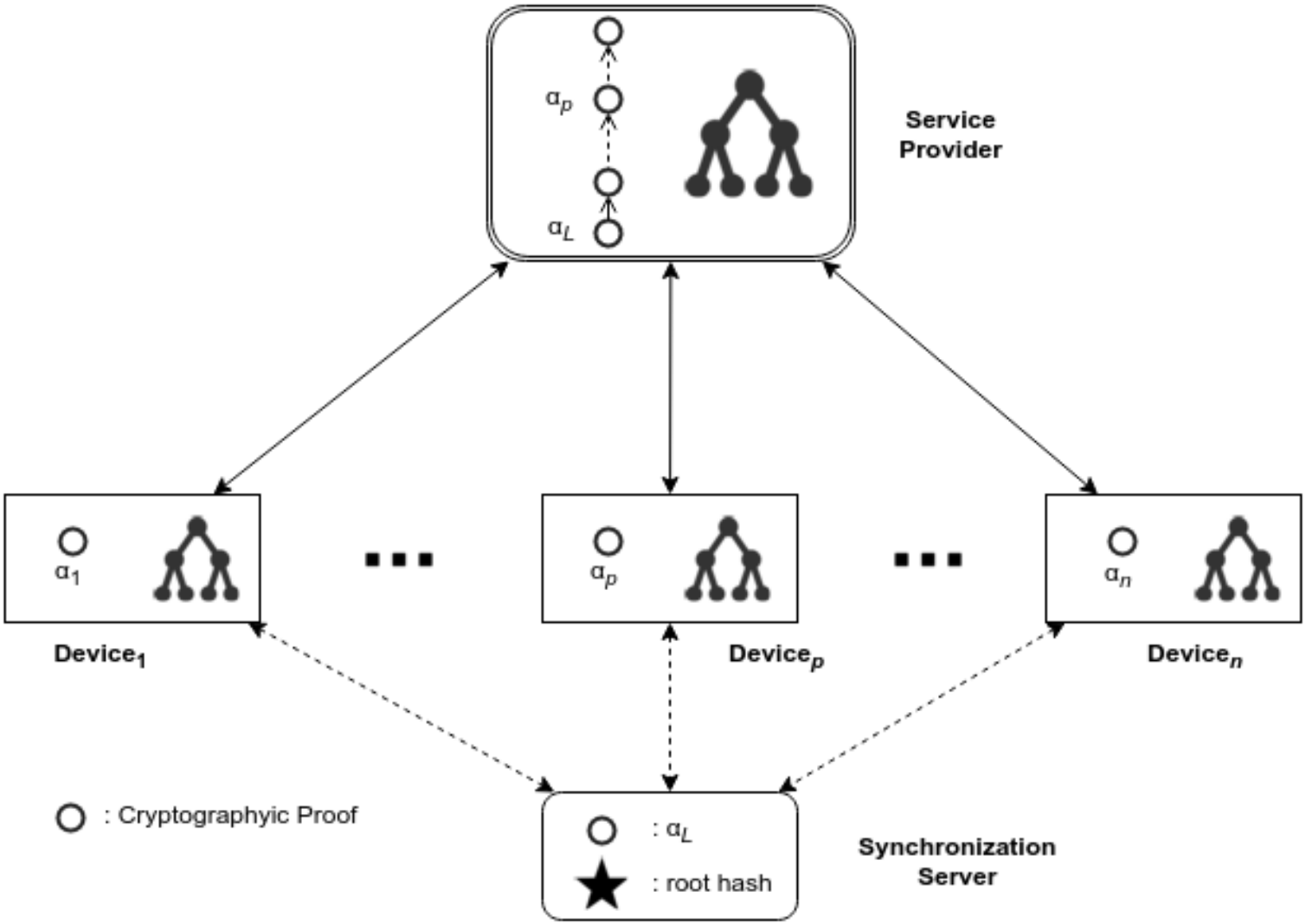
Outline

1. Scenario
2. Real-time Auditing Schemes
   - Instant Auditing of Cloud Storage Access by Cache Partial Merkle tree
   - My Method
3. Protocol Detail
   - Flowchart
   - Download & Upload
   - Audit
4. Experimental Results

Merkle Tree


: Directory
: File
d1
d2
d3
d4
f1
d5
f2
d6
f3
f4
f5
f6
f7
(A)

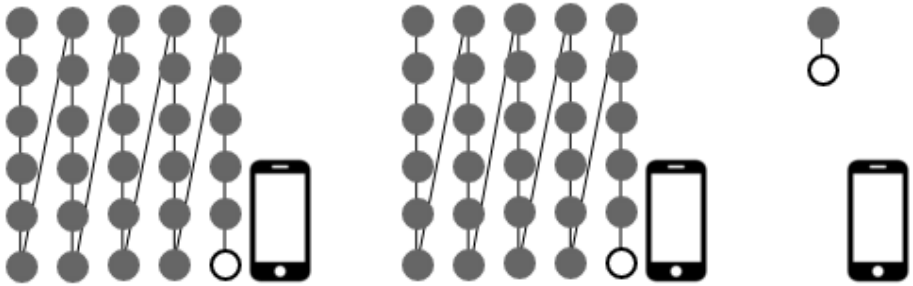

: Hash value of a directory
: Hash value of a file
d1
d2
d3
d4
f1
d5
f2
d6
f3
f4
f5
f6
f7
(B)


α_p
α_L
Service Provider
α_1
α_p
α_n
Device$_1$
Device$_p$
Device$_n$
: Cryptographyic Proof
: α_L
: root hash
Synchronization Server

Worst-case

若有個 device 很久沒有使用，在讀寫檔案前需要更新大量未做的動作

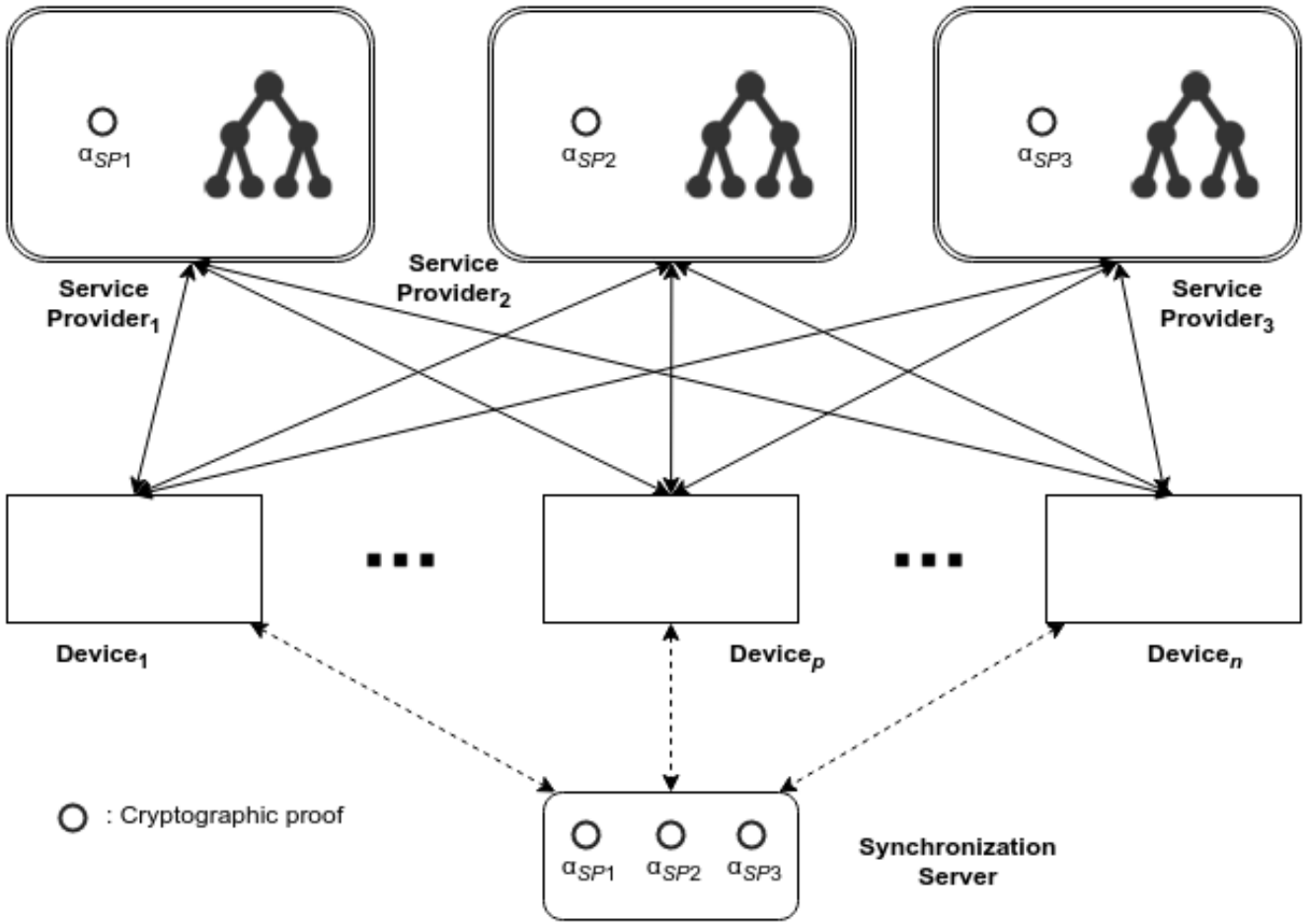
使用者將會明顯感受到漫長的等待時間



My Method

Assumption: 同時有 k 個 server 上同一 file 出問題的機率 ≈ 0

Service Providers are Independent Cloud



Comparison

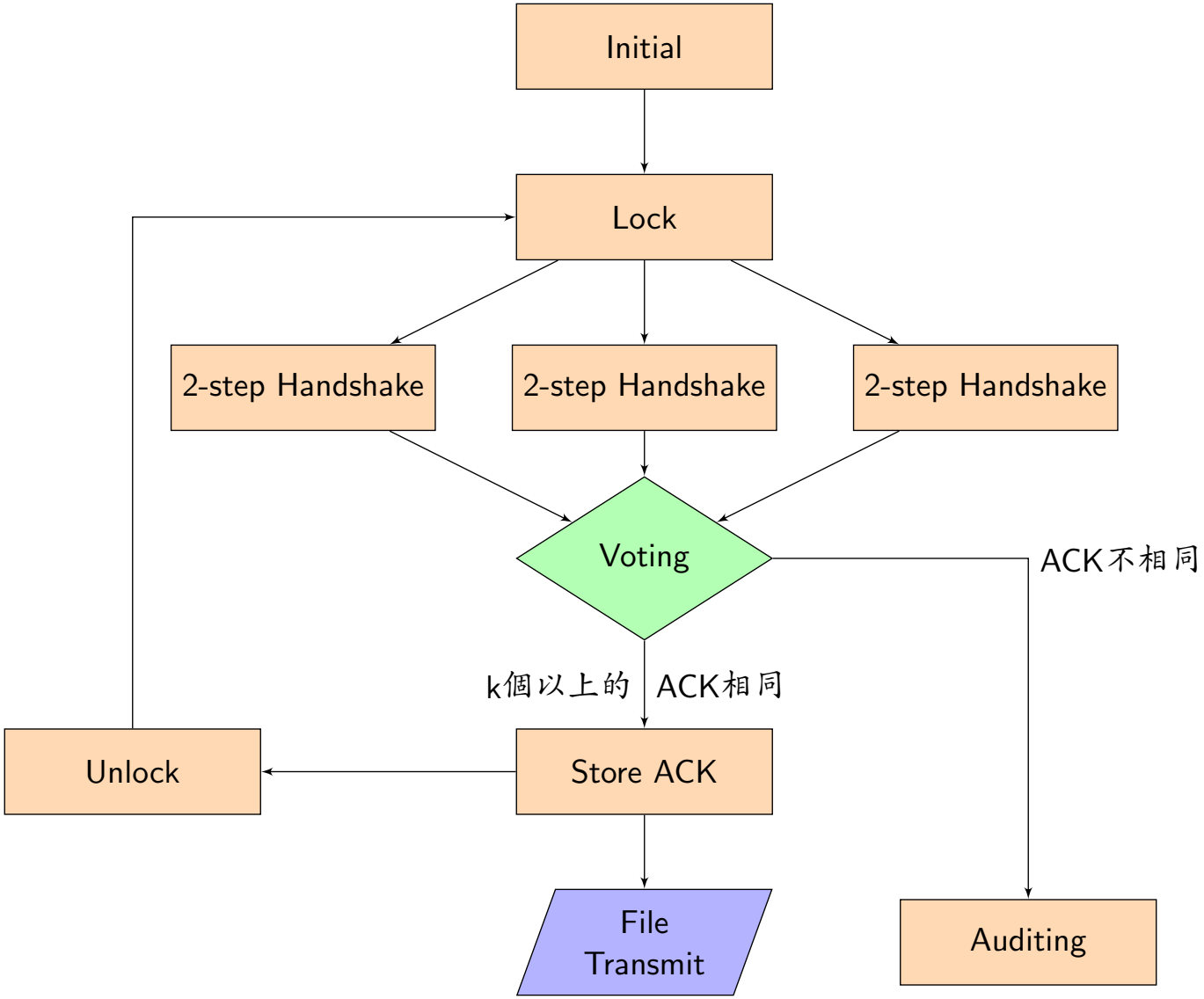
- Pros
  1. Device 不用儲存、也不用修改 Merkle tree, 既省空間又省時間
  2. 資料有多份備份
  3. 解決之前的 Worst-case

- Cons
  1. 需要傳送多份 Request, 處理多份 Response

Outline

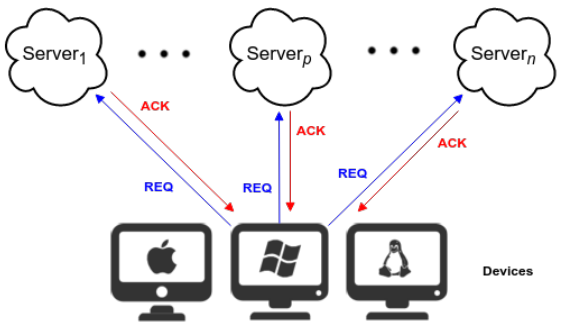
1. Scenario
2. Real-time Auditing Schemes
   - Instant Auditing of Cloud Storage Access by Cache Partial Merkle tree
   - My Method
3. Protocol Detail
   - Flowchart
   - Download & Upload
   - Audit
4. Experimental Results

Flowchart


Initial
Lock
2-step Handshake
2-step Handshake
2-step Handshake
Voting
ACK不相同
k個以上的
ACK相同
Unlock
Store ACK
File
Transmit
Auditing

Download & Upload

2-step Handshake



device $\rightarrow$ servers

$$REQ = (OP, [OP]_{pri(D)})$$
$$OP = (TYPE, PATH, HASH)$$

---

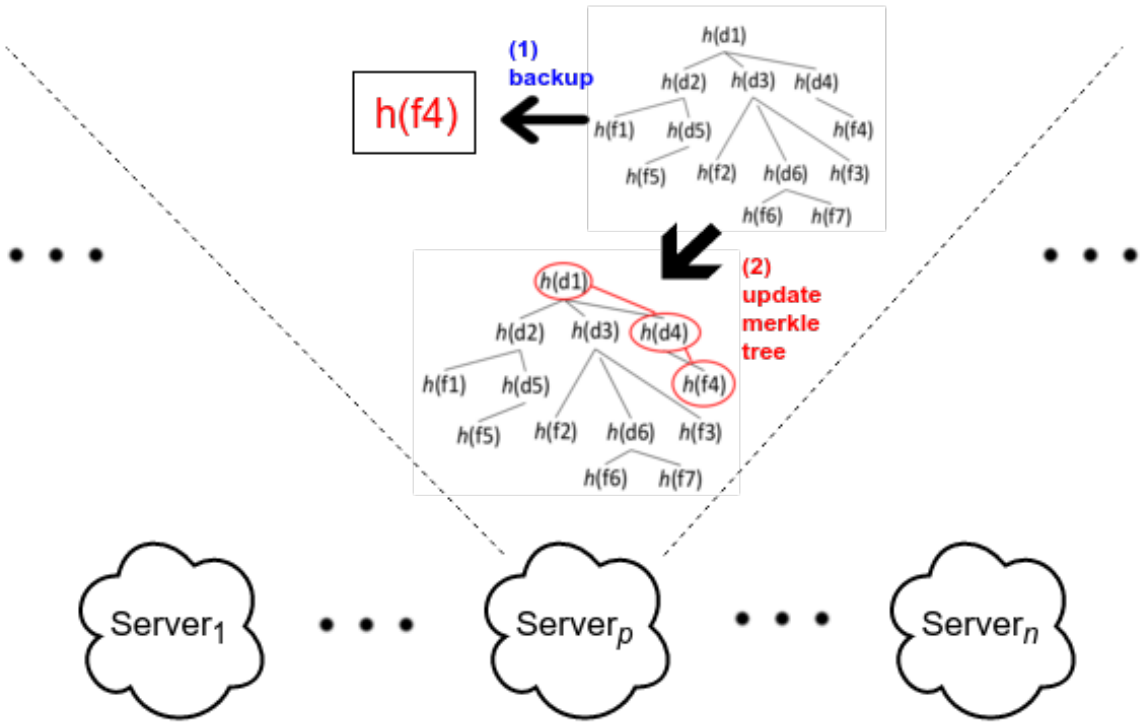
server $\rightarrow$ device

$$ACK = (RESULT, REQ, [RESULT, REQ]_{pri(S)})$$
$$RESULT = (roothash, filehash)$$

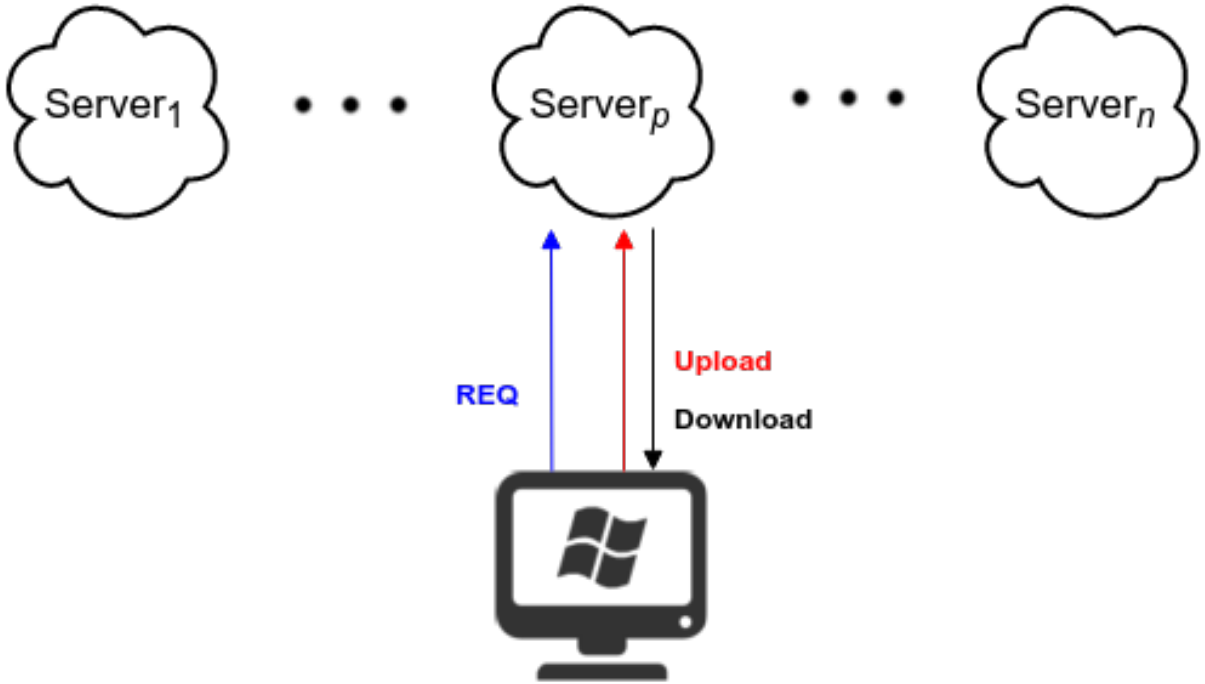
collect ACKs and voting

if Operation is UPLOAD

Server Update Merkle tree



File Transmit



Audit

∵ ACK 中有 roothash
∴ 新的 request 之前，所有的檔案都經過檢查，沒有問題

device request OP_i，收到回傳的 ACK_i
發現 $Server_p$ 的 ACK 有錯誤，因此向 $Server_p$ 稽核

device 向 $Server_p$ 索取 MT_{i-1}
(MT_{i-1} 爲執行 OP_i 之前的 Merkle tree)

★★ 兩點有一個出錯就能確定 $Server_p$ 出錯

★ device 檢查 MT_{i-1} 的 roothash 應和 ACK_{i-1} 中的 roothash 一樣
★ device 以 OP_i 中的 hash value 來更新 MT_{i-1}，
更新後的 roothash 應和 $Server_p$ 現在的 roothash 相同

Outline

1. Scenario
2. Real-time Auditing Schemes
   - Instant Auditing of Cloud Storage Access by Cache Partial Merkle tree
   - My Method
3. Protocol Detail
   - Flowchart
   - Download & Upload
   - Audit
4. Experimental Results

Experimental Results

	Size	File	Directory
A	777 MB	48	6
B	145 MB	54198	188
C	5.95 GB	45089	1459

Table : GENERATE MERKLE TREE'S TIME (IN SEC.)

		A	B	C
	Merkle tree Size	5.4KB	5.08MB	4.37MB
PC	Generate	0.234	1.079	0.952
	Serialize	0.015	0.343	0.265
	Deserialize	0	2.324	1.669
VM	Generate	0.093	0.653	0.246
	Serialize	0.011	0.298	0.238
	Deserialize	0.008	0.269	0.251

Experimental Results

Table : THE EXECUTION TIME OF UPLOAD OPERATIONS (IN SEC.) (Account C)

	NonPOV	WeiChih	WeiShian
Average	1.09991	2.66026	1.05123
	0.203	0.015	0.202
	0.203	0.016	0.203
	0.218	0.031	0.203
	0.218	0.031	0.203
	0.218	0.031	0.218
	.	.	.
	.	.	.
	.	.	.
	4.838	11.536	3.802
	7.289	22.035	7.349
	11.32	36.262	11.947
	15.295	47.495	15.235
	32.208	104.097	34.439

WeiShian's Worst-case:

Synchronize (HashChain length = 100) need 0.595 sec.

Synchronize (HashChain length = 1000) need 5.54 sec.

Experimental Results

Table : THE EXECUTION TIME OF DOWNLOAD OPERATIONS (IN SEC.) (Account C)

	NonPOV	WeiChih	WeiShian
Average	0.82753	0.93367	1.05288
	0	0.015	0.218
	0	0.016	0.218
	0	0.031	0.218
	0	0.031	0.218
	0	0.031	0.218
	.	.	.
	.	.	.
	.	.	.
	3.791	4.628	3.989
	7.499	8.363	7.739
	11.81	12.72	12.031
	14.865	15.778	15.418
	33.513	35.775	33.713

WeiShian's Worst-case:

Synchronize (HashChain length = 100) need 0.827 sec.

Synchronize (HashChain length = 1000) need 7.871 sec.

Thank You!