

Instant Auditing of Cloud Storage Access without Accumulating Attestations

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

NTNU CSIE CCLAB

2015.09.02

- 1 Scenario
- 2 POV's Evolution
 - Single Client and Service Provider
 - Multiple Clients
 - Reduce Device's Storage Usage
- 3 Protocol Detail
 - Flowchart
 - Initial
 - Read
 - Write
 - Audit
- 4 Implement Steps
- 5 Experimental Results

Outline

1 Scenario

2 POV's Evolution

- Single Client and Service Provider
- Multiple Clients
- Reduce Device's Storage Usage

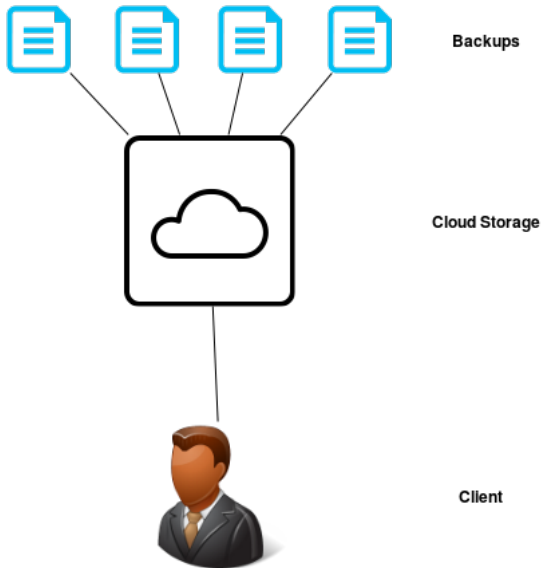
3 Protocol Detail

- Flowchart
- Initial
- Read
- Write
- Audit

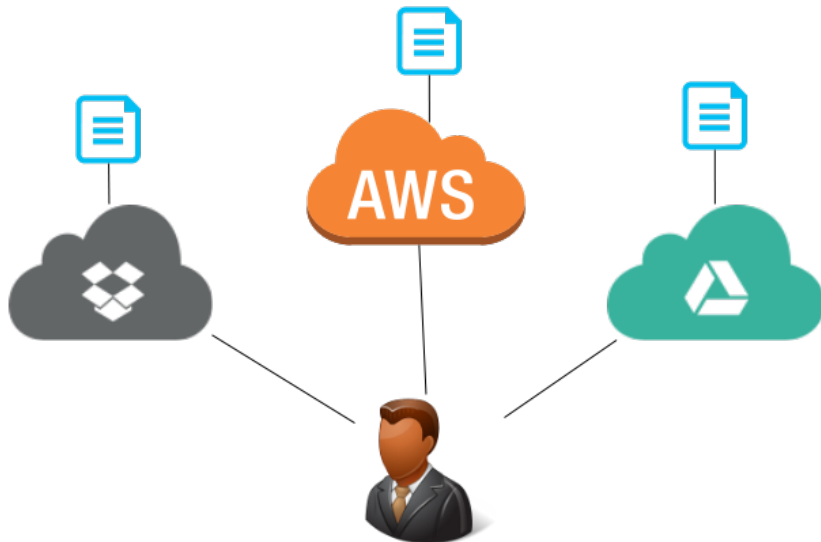
4 Implement Steps

5 Experimental Results

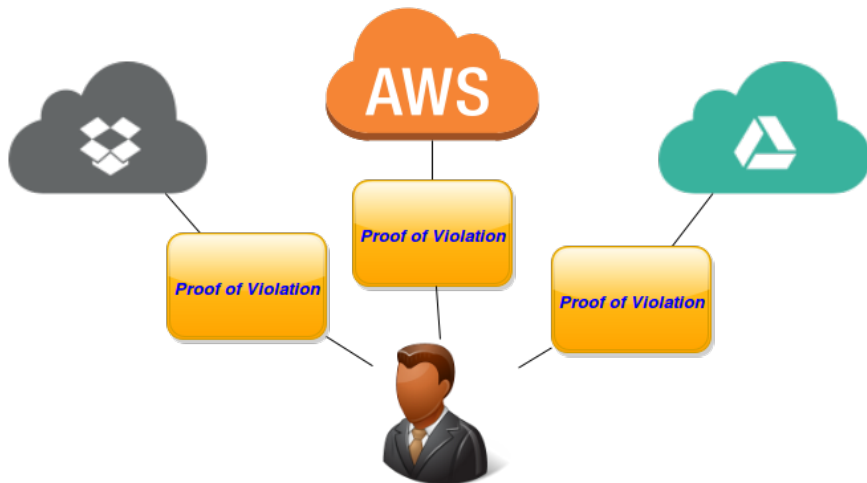
Scenario



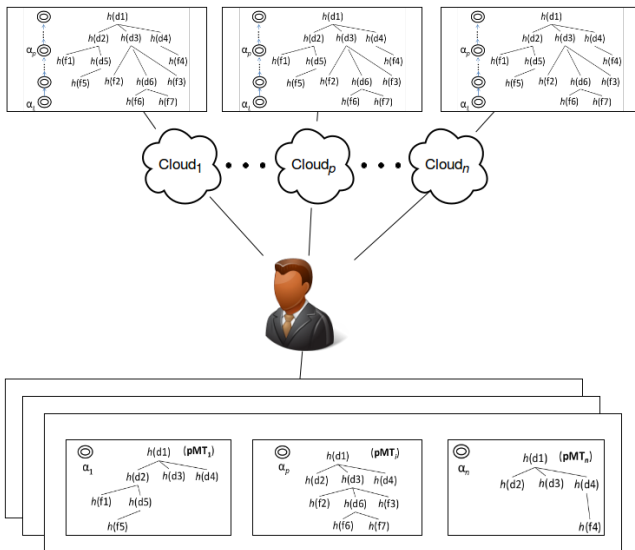
Scenario - Backup



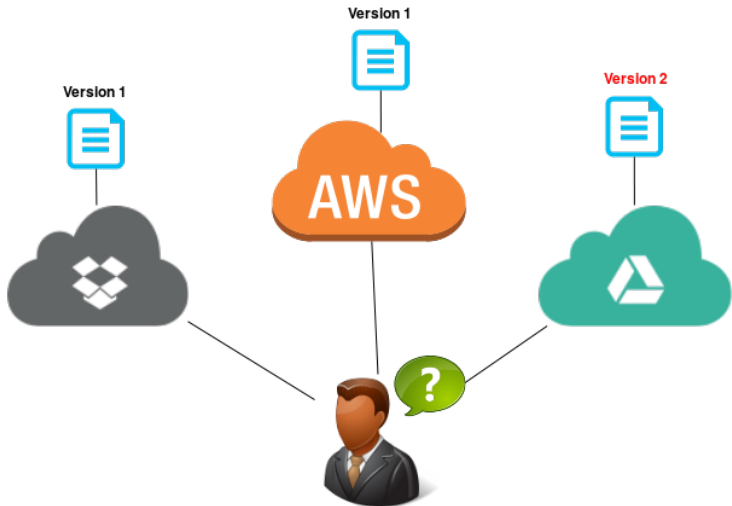
Scenario - POV



Scenario - Problem I : Too Many Attestations



Scenario - Problem II : Version Control is Difficult



Outline

1 Scenario

2 POV's Evolution

- Single Client and Service Provider
- Multiple Clients
- Reduce Device's Storage Usage

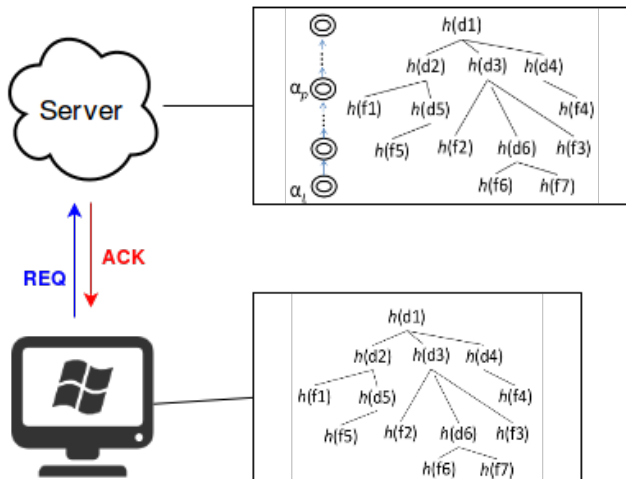
3 Protocol Detail

- Flowchart
- Initial
- Read
- Write
- Audit

4 Implement Steps

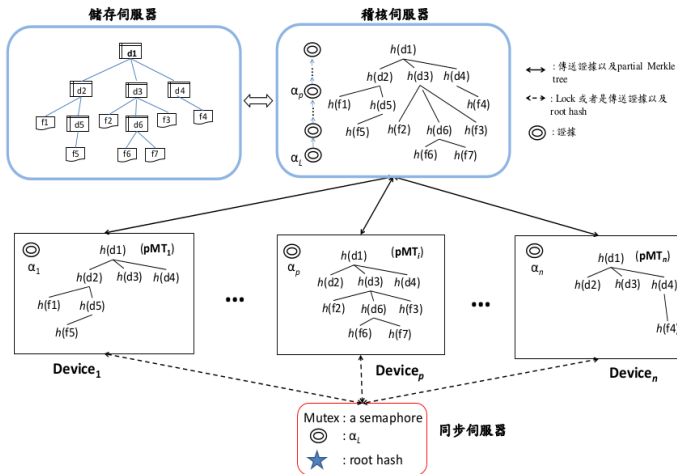
5 Experimental Results

Single Client and Service Provider



Multiple Clients

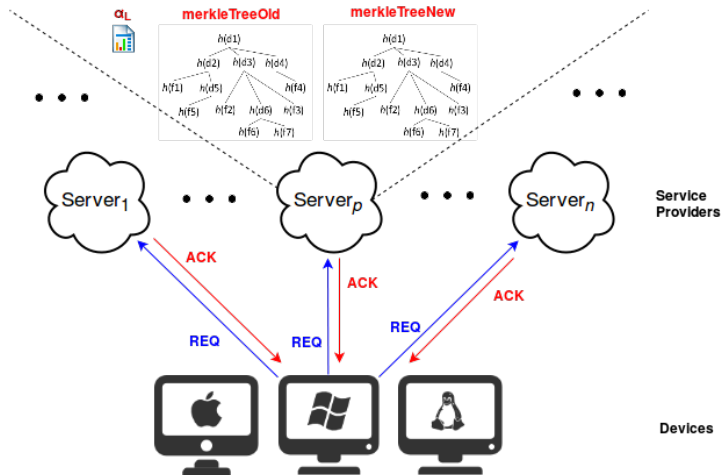
(Instant Auditing of Cloud Storage Access by Cache Partial Merkle tree)



Worst-case: 很久沒更新而累積了大量的files update動作

Reduce Device's Storage Usage

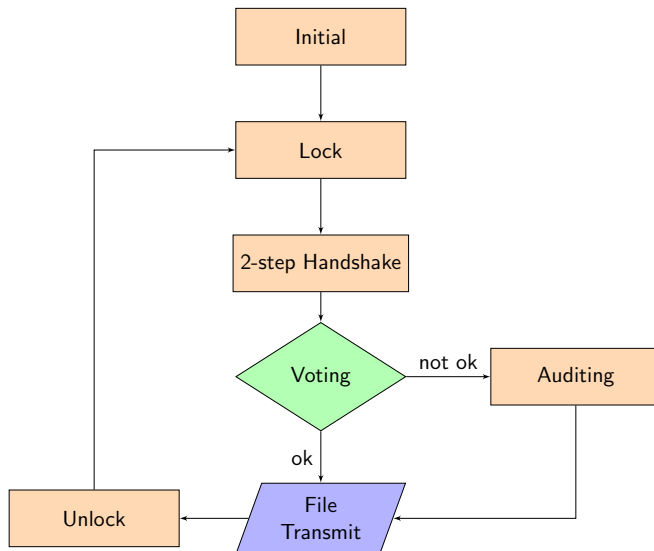
Assumption: 同時有 k 個server上同一file出問題的機率 ≈ 0



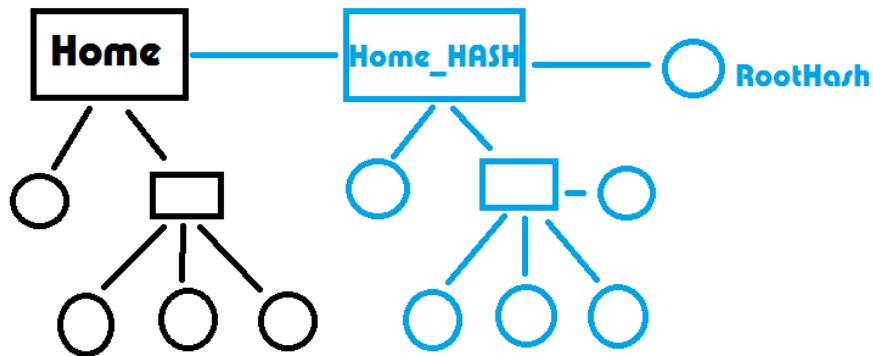
Outline

- 1 Scenario
- 2 POV's Evolution
 - Single Client and Service Provider
 - Multiple Clients
 - Reduce Device's Storage Usage
- 3 Protocol Detail**
 - Flowchart
 - Initial
 - Read
 - Write
 - Audit
- 4 Implement Steps
- 5 Experimental Results

Flowchart

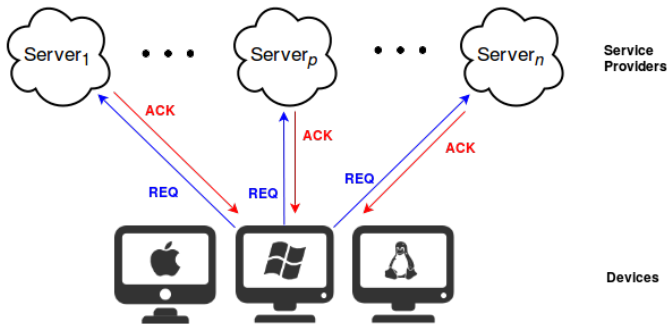


File → Merkle Tree



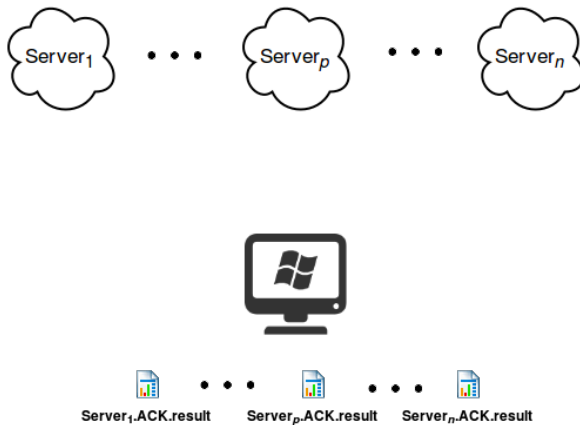
READ

I. 2-step Handshake



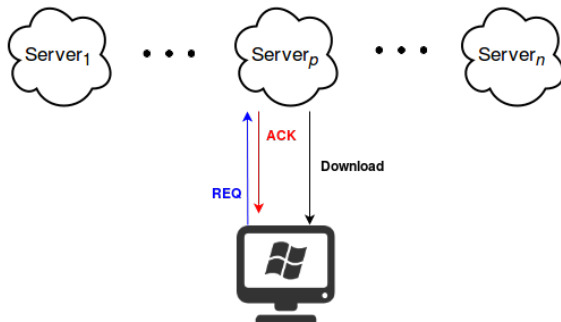
REQ = (op, [op]_{pri(D)})
op.type = DOWNLOAD
op.path = filepath
op.msg = " "

ACK = (result = merkleTreeNew.roothash,
REQ,
[result, REQ]_{pri(s)})



READ

III. Download



$REQ = (op', [op']_{pri(D)})$
 $op'.type = \text{DOWNLOAD}$
 $op'.path = op'.getPath()$
 $op'.msg = ACK.result$

$ACK = (result = merkleTreeNew.roothash,$
 $REQ,$
 $[result, REQ]_{pri(s)})$

WRITE

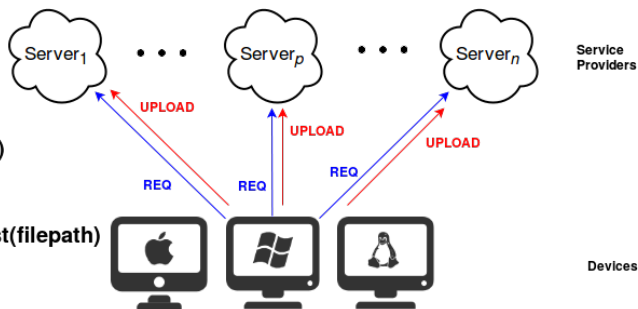
I. Upload

$REQ = (op, [op]_{pri(D)})$

$op.type = \text{UPLOAD}$

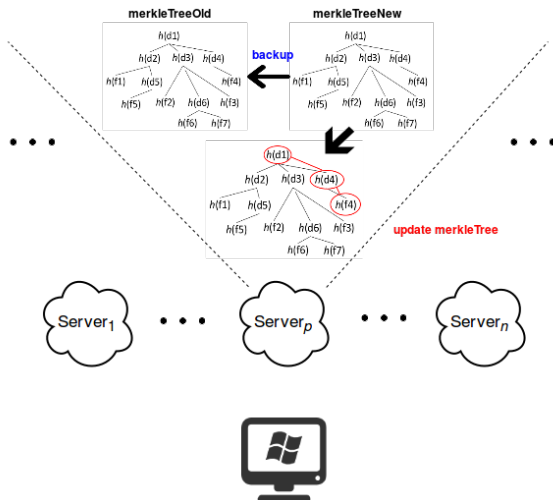
$op.path = \text{filepath}$

$op.msg = \text{Utils.digest(filepath)}$



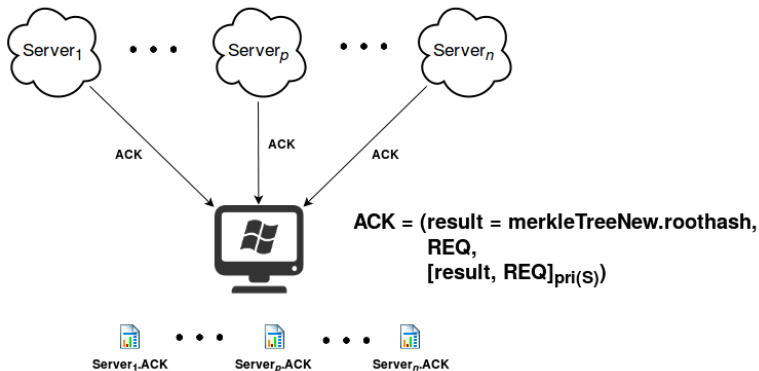
WRITE

II. Update Merkle Tree



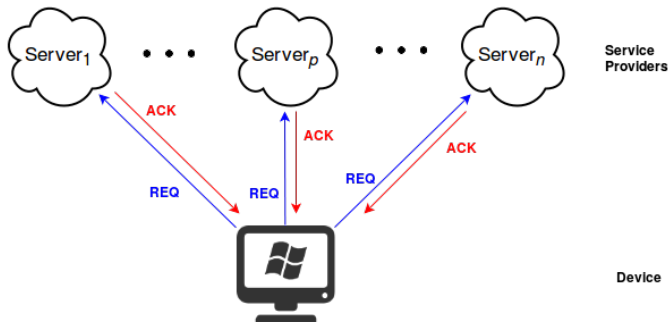
WRITE

III. Voting



AUDIT

I. Download and Check LastReq

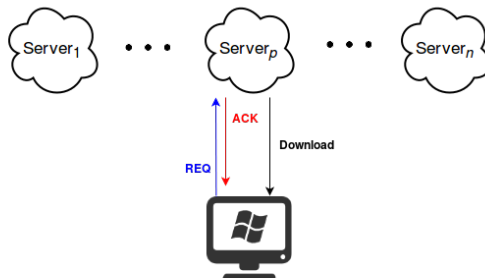


REQ = (op, [op]_{pri(D)})
op.type = AUDIT
op.path = " "
op.msg = " "

ACK = (result = merkleTreeNew.roothash,
REQ = lastReq,
[result, REQ]_{pri(S)})

AUDIT

II. Download Roothash or Merkle Tree



```
REQ = (op, [op]pri(D))  
op.type = AUDIT  
op.path = attestationPath  
op.msg = ""
```

```
ACK = (result = Utils.digest(file),  
      REQ,  
      [result, REQ]pri(S))  
if lastReq.op == DOWNLOAD:  
    file = merkleTreeOld.roothash  
if lastReq.op == UPLOAD:  
    file = serialize(merkleTreeOld)
```

```
1 AUDIT(lastAck)
2   op ← lastAck.req.op
3   if op.type = DOWNLOAD
4       success ← roothash.equals(lastAck.result)
5   if op.type = UPLOAD
6       merkleTreeOld.update(op.msg)
7       success ← roothash.equals(lastAck.result)
8   return success
```

Listing 1 : Audit algorithm

Outline

- 1 Scenario
- 2 POV's Evolution
 - Single Client and Service Provider
 - Multiple Clients
 - Reduce Device's Storage Usage
- 3 Protocol Detail
 - Flowchart
 - Initial
 - Read
 - Write
 - Audit
- 4 Implement Steps
- 5 Experimental Results

Implement Steps

- ① Hash Handle : create, update, delete
- ② Operation Handle : read, write, audit
- ③ File Transmit : send, receive
- ④ Merkle Tree Transmit : serialize
- ⑤ *Instant Auditing of Cloud Storage Access by Cache Partial Merkle tree*
- ⑥ Run on Real Cloud Environment (VM)

Outline

- 1 Scenario
- 2 POV's Evolution
 - Single Client and Service Provider
 - Multiple Clients
 - Reduce Device's Storage Usage
- 3 Protocol Detail
 - Flowchart
 - Initial
 - Read
 - Write
 - Audit
- 4 Implement Steps
- 5 Experimental Results

Create Merkle Tree

Account A	666 MB	42 files	6 directories
Account B	34 MB	54192 files	188 directories
Account C	6.54 GB	58484 files	1718 directories
Account D	20.6 GB	175389 files	5154 directories

Table : TIMES REQUIRED TO GENERATE THE ROOT HASH FROM NOT-HASHED FILES (IN SECONDS)

Account	Senior	My	MerkleTree Size
A	<i>3.404</i>	<i>3.645</i>	3.74 KB
B	<i>16.618</i>	<i>7.669</i>	3.77 MB
C	<i>229.351</i>	<i>242.198</i>	4.30 MB
D		<i>815.408</i>	12.9 MB

Operation Processing Time

Table : DOWNLOAD TIME (ms)

Account	100 times	Audit*
A	4635	34 + 0
B	4660	33 + 0
C	5429	31 + 0
D	5554	31 + 0

Table : UPLOAD TIME (ms)

Account	100 times	Audit*
A	4322	41 + 7
B	5643	421 + 997
C	9236	421 + 2621
D	11466	1263 + 8085

* download attestations time + audit time

Thank
You!

