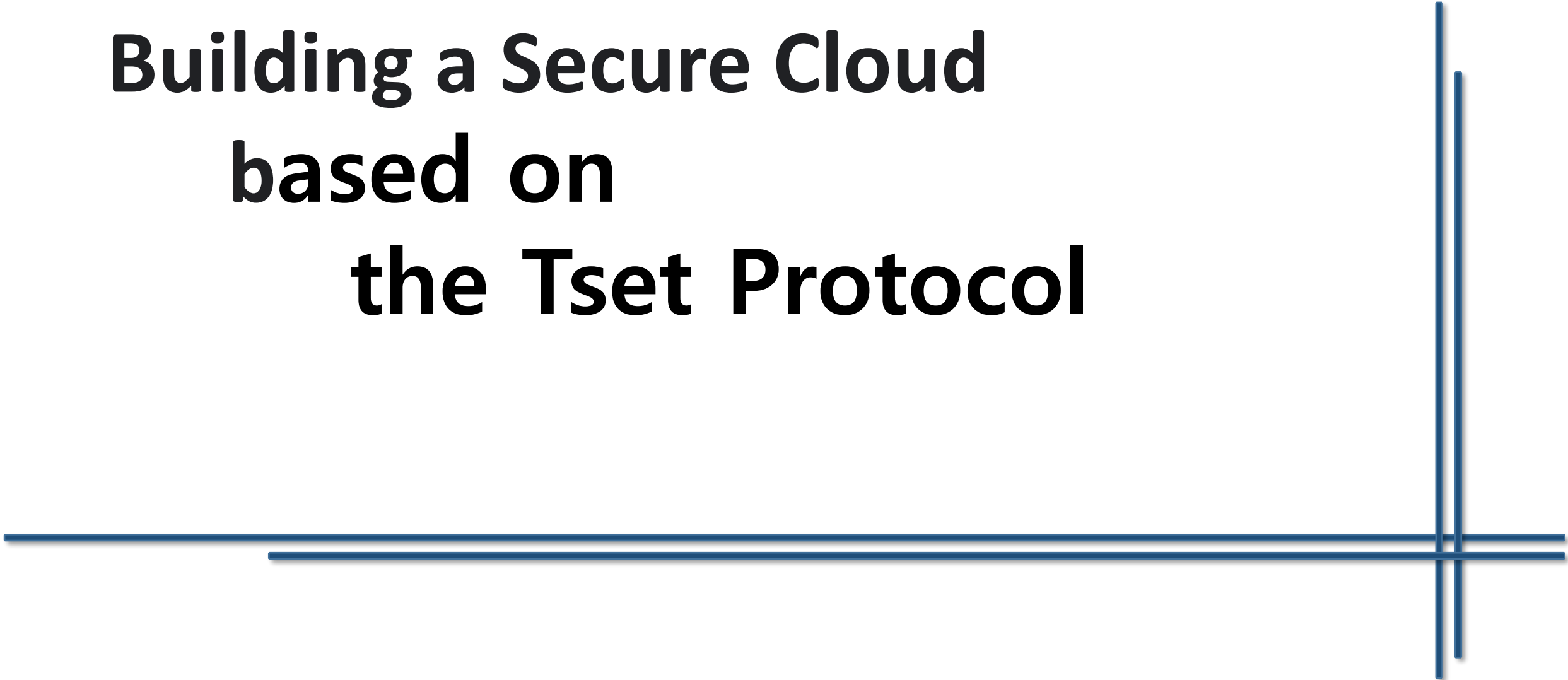


Building a Secure Cloud based on the Tset Protocol



Introduction

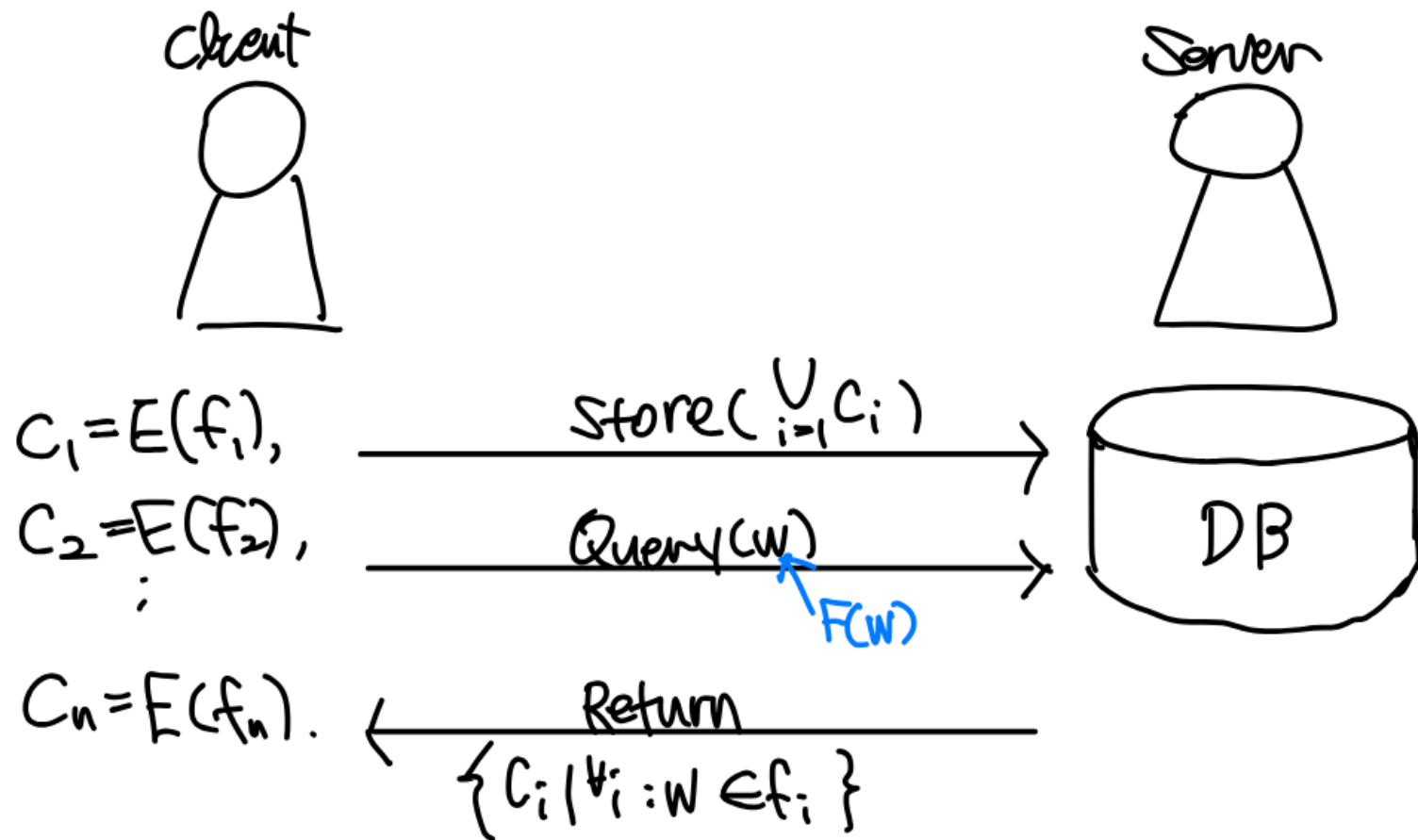
- **Highly-scalable searchable symmetric encryption with support for Boolean queries** 논문의 SSE 시스템 구현 (Tset protocol)
 - **OPENSSL(AES128, SHA256), BOOST/ASIO 라이브러리 사용**
 - **Socket Programming: 클라이언트, 서버-데이터베이스 구현**
 - **MariaDB, 패킷 설계, Serialize**

Why Tset ?

- 사람은 원하는 정보 검색할 때 키워드 위주 서칭을 한다 .
- 클라우드에 암호화 된 파일을 검색 시, 안전한 검색을 지원하는 방법
- Keyword w 제공->w 포함하는 모든 파일의 id찾음
- Ex) 경찰청 클라우드에서 사건 문서를 검색 - 익사, 노란색코트, 인상착의 등의 키워드로 검색 가능
- 이러한 키워드들은 파일 내에 직접적으로 존재할 수도 있지만 파일 내의 단어가 아닌, 주요 키워드 추출 알고리즘을 통해 얻은 독립적인 또 다른 데이터들이다.
- 중복되어 나타나는 키워드의 존재와 분포는 기밀하게 지켜야 할 대상이 된다.

파일 암호화

- Tset 구축 전, 파일 자체의 암호화 시행 후 전송



<Store>

$$X_i \leftarrow \text{AES.E}(K_0, w_i) \text{ and } X_i = \overset{16}{L_i} \parallel \overset{8}{R_i} \text{ (8 bytes)}$$

$$S_i \leftarrow G(K_1, i) \text{ and } \overset{16}{\kappa_{p,i}} \leftarrow G(K_2, L_i) \text{ (12 bytes)}$$

$$U_i \leftarrow \text{F}(\overset{8}{\kappa_{p,i}}, \overset{40 \text{ bytes}}{S_i}) \text{ and set } T_i = \overset{16}{S_i} \parallel \overset{8}{U_i} \text{ (8)}$$

$$c_i \leftarrow X_i \oplus T_i$$

of words n words, $C = (c_1 \parallel c_2 \parallel \dots \parallel c_n)$

Send C to Server

<Restore>

$$c_i^* = A_i \parallel B_i$$

$$S_i \leftarrow G(K_1, i) \text{ and } L_i \leftarrow S_i \oplus A_i$$

$$\kappa_{p,i} \leftarrow G(K_2, L_i) \text{ and } U_i \leftarrow \text{F}(\kappa_{p,i}, S_i) \text{ (8)}$$

$$T_i \leftarrow S_i \parallel U_i$$

$$X_i \leftarrow c_i^* \oplus T_i \text{ and } w_i \leftarrow \text{AES.D}(K_0, X_i)$$

of words n words, $D = (w_1 \parallel w_2 \parallel \dots \parallel w_n)$

Store D as a file.

Why Tset ?

- 클라우드는 외부 서버이기 때문에, 모든 파일이 암호화 되어있는 것은 당연하고
- 1) 어떤 키를 받기 전까지는 Search 자체가 불가
- 2) Client가 키워드를 전송했을 때 Server는 아무것도 모르는 상태에서 키 만으로 검색을 수행해야 함

dict 1

file id : keyword

1 : "crypto", "AES"

2 : "value", "crypto", "action"

3 : "AES", "state", "time"

4 : "OS", "DPR", "Tag"

⋮

dict 2 (IID)

keyword : file id

"Crypto" : 1, 2

"AES" : 1, 3

"value" : 2

"action" : 2

"state" : 3

"time" : 3

⋮

Dict 3 (T분포)

keyword : Encrypted file id

"Crypto" : 1, 2

"AES" : 1, 3

"value" : 2

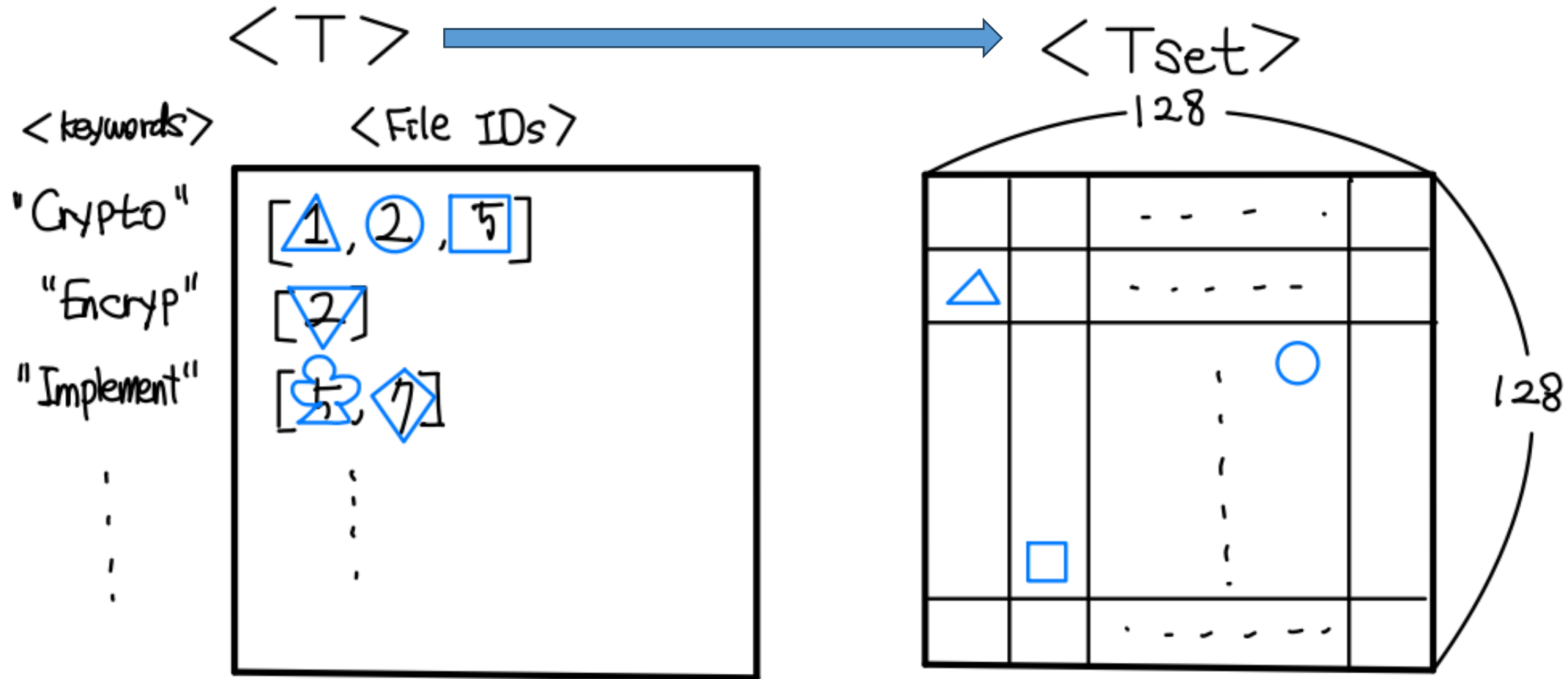
K_e (keyword & K_s)

← 사용해서 각 행 암호화

T 만으로는 키워드 & 파일 모두 암호화되어 있더라도 어떤 키워드가 갖는 파일 개수와 분포를 공격자가 쉽게 알아챌 수 있다.

⇒ TSet protocol 제안

A short explanation of a Tset



• Client 안에서 할 일: 파일 당 키워드 추출하고, T분포 만들기 → 이를 Tset으로 만들어서 binary 형태로 서버로 전송.

(Documents는 미리 Kd로 암호화하여 전송)

필요한 Key: $K_d, K_s, K_T \leftarrow \text{rand}(), \{0,1\}^{\leftarrow 32}$

Document ↘ keyword ↘ stag

TSetSetup(T, W) → (Tset) Type: pair(key, value) = α
Size: $S \times S$.

$W = \bigcup_{i=1}^n w_i$

How to figure out S? 정수이고, 2^n

$\leftarrow \sqrt{\text{max}(\# \text{ of keywords}) \times \text{max}(\# \text{ of ids})}$

for each $w_i \in W$:

$\text{stag} \leftarrow \text{AES.E}(K_T, w_i)$

for $i=1$ to t_w : // t_w : # of ids which contain w

어떤 키워드가 가질 수 있는 파일 id들의 최대값

for each $w_i \in W$:

$stag \leftarrow AES.E(K_T, w_i)$

for $i=1$ to t_w : // t_w : # of ids which contain w

$(b \parallel L \parallel k) \xleftarrow{24b} SHA(AES.E(stag, i))$

if $Free[b] = \emptyset$, then abort

$j \leftarrow Free[b] = \{1, 2, \dots, S\}$ // pick a random column

if $i < t_w$:

$\beta \leftarrow 1$ // 해당하는 파일 id가 더 있다는 indicator

else

$\beta \leftarrow 0$

$\alpha.key \leftarrow L$

$\alpha.value \leftarrow (\beta \parallel e_i) \oplus k$

$Tset[b, j] \leftarrow \alpha$

어떤 키가 가질 수
있는 파일 id들의 화제값

7, 80, 129

$\uparrow \quad \uparrow \quad \uparrow$
 $S=128 \quad p=\frac{1}{2^{80}} \quad 128+1$

$\Rightarrow Tset[b, j]$

b: row, L: Stag 단계에서 retrieve 시 $L == Tset(i,j).first$

↑ ↑
시행에서 AES → parse 실행 Tset

β : 1 ← 더 찾아볼 신호, 0 ← 그만 찾아볼 권하는 신호

e_i : 암호화된 파일 ID들

• Server 에서 할 일 : 클라이언트로부터 Stag(비밀키) 받음

i) DB에 전송된 해당 클라이언트의 Tset 내에서 $\{e_1, e_2, \dots, e_{tw}\}$
찾아서 전송

(client: $\{e_1, e_2, \dots, e_{tw}\}$ 복호화 후 $\{id_1, id_2, \dots, id_{tw}\}$ 전송)

ii) $\{id_1, id_2, \dots, id_{tw}\}$ 를 받은 서버는 DB에서 해당
파일 id에 해당하는 Documents 모두 전송

(client: Document 받아서 복호화 후, 파일로 저장)

TSet Retrieve(TSet, stag) $\rightarrow (V = (\bar{id}_1, \bar{id}_2, \dots, \bar{id}_t))$

$V \leftarrow \emptyset, \beta \leftarrow 1, i \leftarrow 1$

while $\beta = 1$ do

$(b \| L \| K) \leftarrow \text{SHA}(\text{AES.E}(\text{stag}, i))$

$T \leftarrow \text{TSet}[b]$

 for $1 \leq j \leq S$ do

 if $T[b, j].\text{label} = L$ then // 해당하는 파일 아이디 찾기

$V \leftarrow T[b, j].\text{value} \oplus K$ b행에 1개 이상 있을 수 있음.

$\beta \leftarrow V[0]$

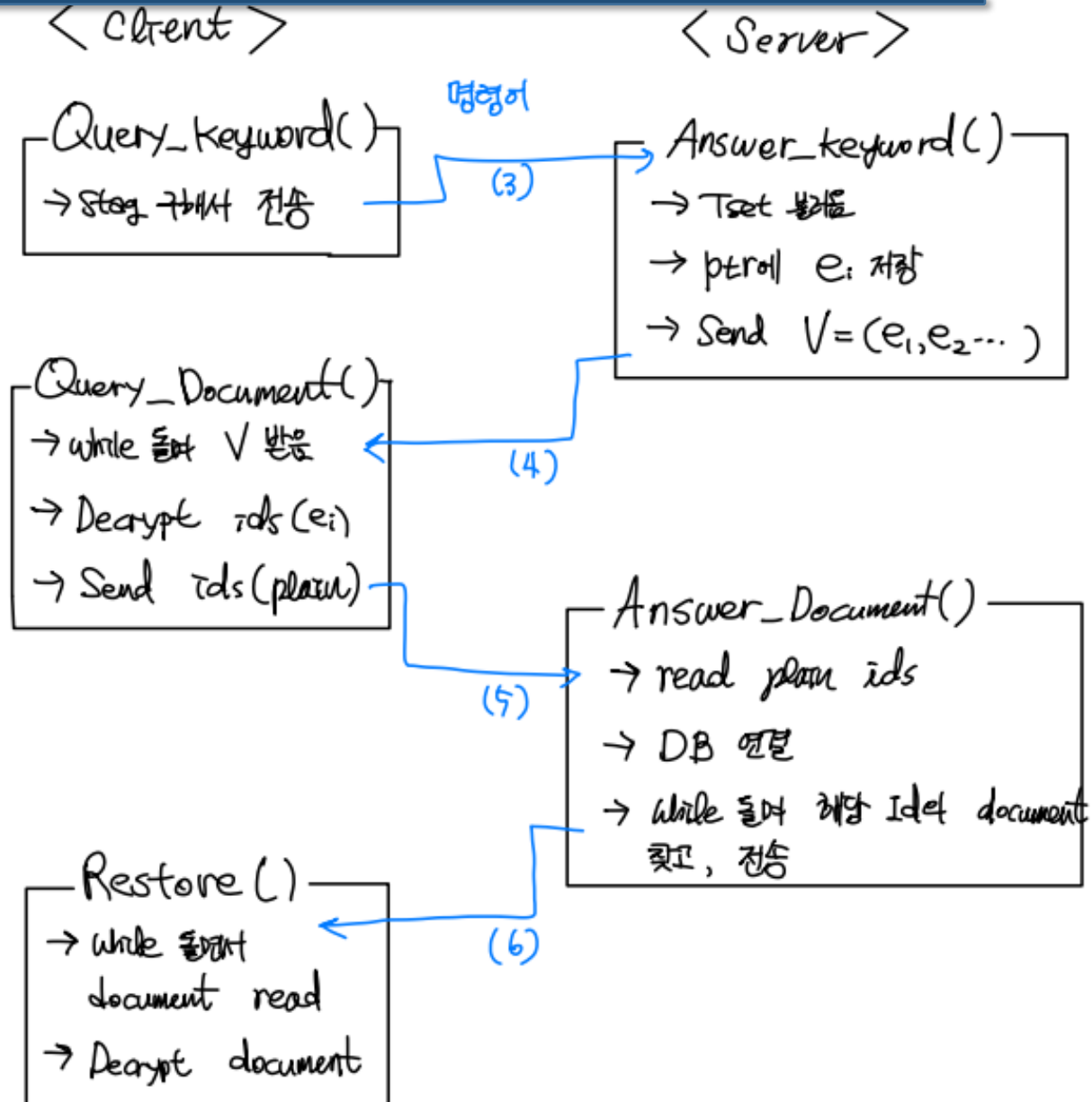
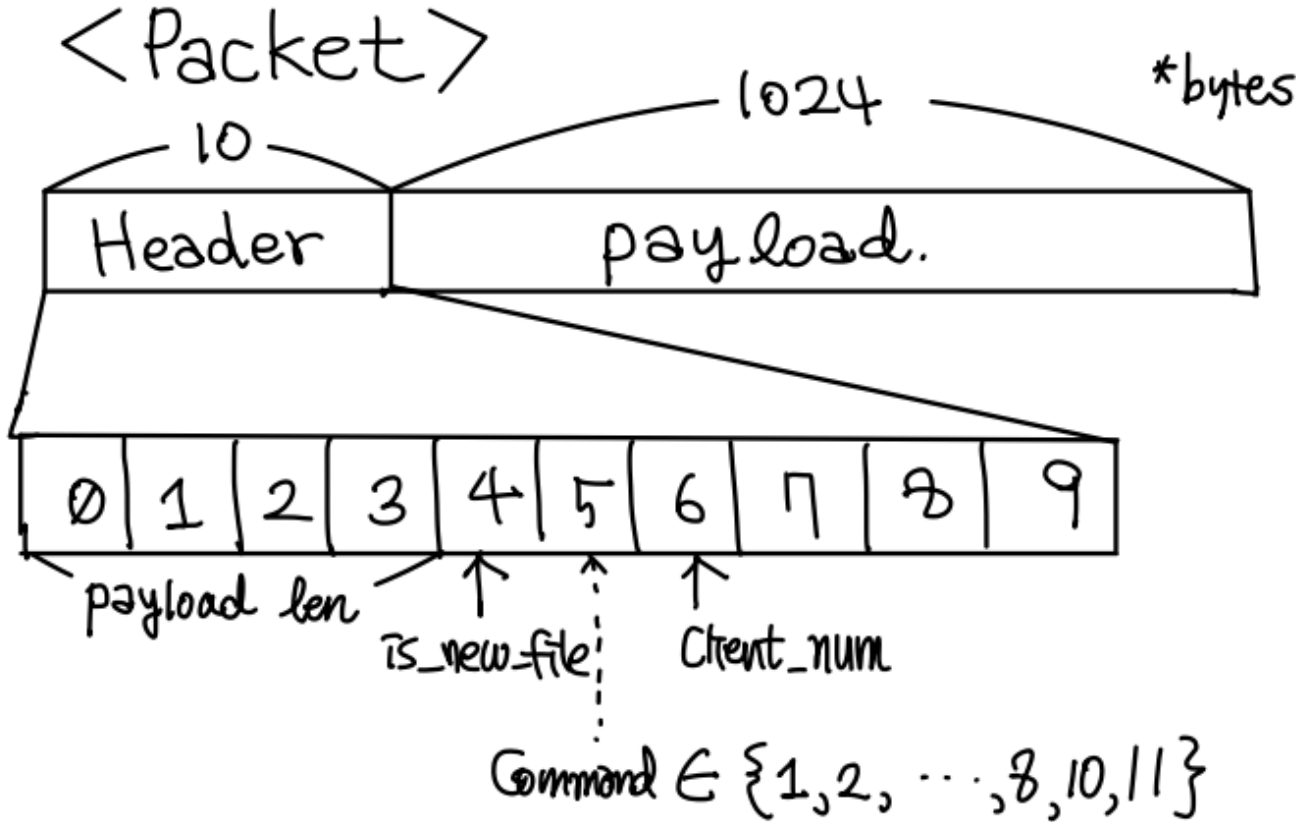
$id \leftarrow V[1:129]$ // V의 바이트 128바이트

$V \leftarrow V \cup \{id\}$

$i \leftarrow i + 1$

return V

Network Communication



Unnamed-1\tsettable\file_ - HeidiSQL 12.3.0.6589

파일 편집 검색 쿼리 도구 이동 도움말

데이터베이스 테이블 필터

호스트: 127.0.0.1 데이터베이스: tsettable 테이블: file_ 데이터 쿼리*

Unamed-1

- information_s...
- tables
- tsettable 1.6 MiB
 - client_ 1.5 MiB
 - file_ 32.0 KiB

tsettable.file_: 4 행 (중) (대략적)

id	contents	date_	client_id
1	원C(語學<韓文???)男??n??#??基t세??F...	2023-11-28	1
2	bp?4n?가??nn...출발(?) z_s라하유도Rn물장...	2023-11-28	1
3	L7j737ak?l(?)을?R??을?n??>??ud(?)L??S...	2023-11-28	1: Alice
4	!*를?n??를??을??을??을??을??을??을??을??...	2023-11-28	1: Alice 2: Taylor

참조 < 파일 뷰어 링크를 클릭하시면
해당 유저 파일만 접근해서 라운가능.

Document table

→ client_id: foreign key

Unnamed-1\tsettable\client_ - HeidiSQL 12.3.0.6589

파일 편집 검색 쿼리 도구 이동 도움말

데이터베이스 테이블 필터

호스트: 127.0.0.1 데이터베이스: tsettable 테이블: client_ 데이터 쿼리*

Unamed-1

- information_s...
- tables
- tsettable 1.6 MiB
 - client_ 1.5 MiB
 - file_ 32.0 KiB

tsettable.client_: 2 행 (중) (대략적)

id	tset	name
1	...	Alice
2	...	Taylor

Client info table

→ tset: blob data

DataBase

The screenshot shows the HeidiSQL application window. At the top, it displays the file path 'Unnamed-1\unsettable\file_\ - HeidiSQL 12.3.0.6589'. Below the title bar is a toolbar with various icons for file operations, navigation, and editing. The main interface is divided into three panes:

- Left Pane:** Shows the database structure. Under 'Unamed-1', there are folders for 'information_s...' and 'tables'. The 'tsettable' database is expanded, showing two tables: 'client_' (1.5 MiB) and 'file_' (32.0 KiB).
- Middle Pane:** Displays the contents of the 'file_' table. It shows four rows with columns 'id', 'contents', 'date_', and 'client_id'. The 'client_id' column has a dropdown menu open, showing the selected value '1: Alice' and another option '2: Taylor'.
- Right Pane:** This pane is currently empty or displaying a very faint, illegible view.

At the bottom of the window, the status bar provides information about the current view: '2,934 문자 (최대: 65,535), 28 줄, 10:15에 커서, read-only'.

The Limitations..

- **Improvements to the code itself**

- Connecting with many clients
- Packet processing structure



- **The Tset**

- The server eventually knows the number of file IDs when you query.
- Actual IDs exposed when you request documents.
- As the queries are repeated, security continues to decrease.
- Only Single keyword search provided

The ways to improve the system.



- **Improvements to the code itself**
 - Profiling needed
- **The Tset**
 - Search two different keywords at Once
 - Reset the Tset every time you query a keyword
 - Compare the cost of reduced security searching keywords & the cost of resetting the Tset

Thank you