# The Magic World of Searchable Encryption

Tobias Mueller

HUAWEI

Christian Forler

HOCHSCHULE SCHMALKALDEN
UNIVERSITY OF APPLIED SCIENCES

1. State of the World^W Cloud
2. Simple Encryption
3. Keyword-based Encryption (Song, Wagner, Perrig)
4. Index-based Searchable Encryption
5. Outlook & Conclusions
6. Thanks!

# State of the World^W Cloud

# Upload your data to Stan's Cloud storage. First 2 GBs are for free.

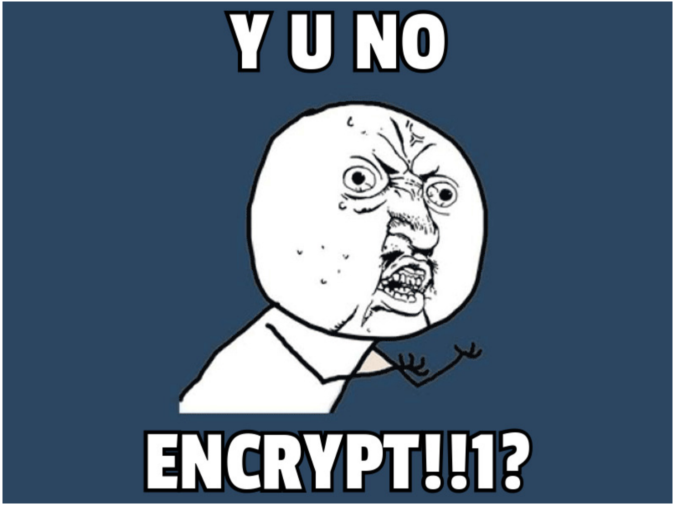
# You even pay them to take your data.

# Newsflash: The dark side performs data mining

# We are on a wrong track

Something has gone completely wrong. . . We pay companies to mine our data for the dark side. . . And sell user profiles. . .

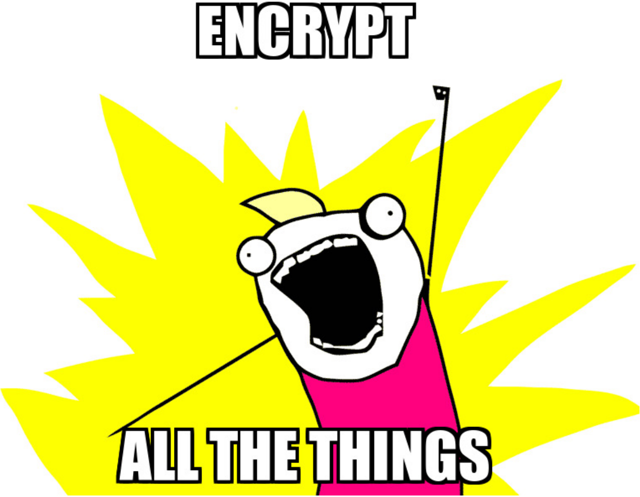
# Now, it is time to get back on track

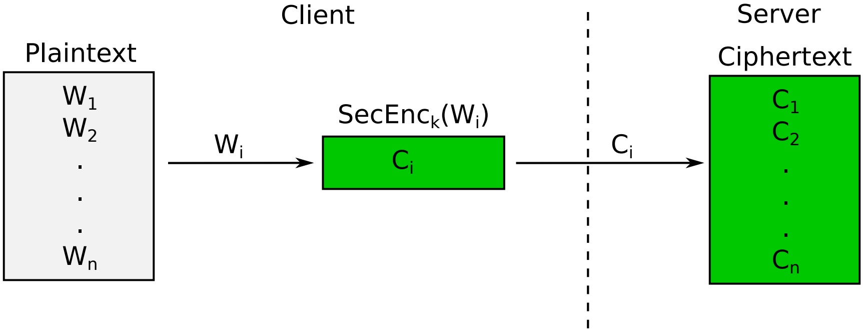


- We do not present a fully-fleged solution.
- We are just pointing into the right direction to get back on track.

# Simple Encryption

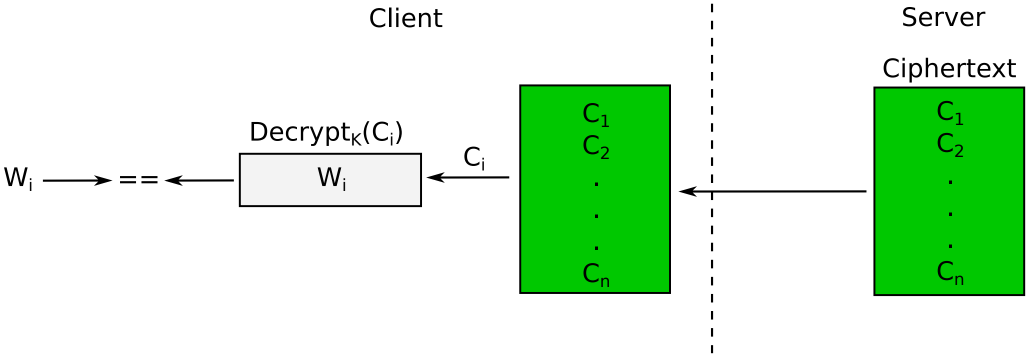
# Encrypt all the things!




Client
Server
Plaintext
$W_1$
$W_2$
.
.
.
$W_n$
$W_i$
$SecEnc_k(W_i)$
$C_i$
$C_i$
Ciphertext
$C_1$
$C_2$
.
.
.
$C_n$

REALLY DOWNLOAD
ALL THE THINGS?

# Simple Crypto - Search



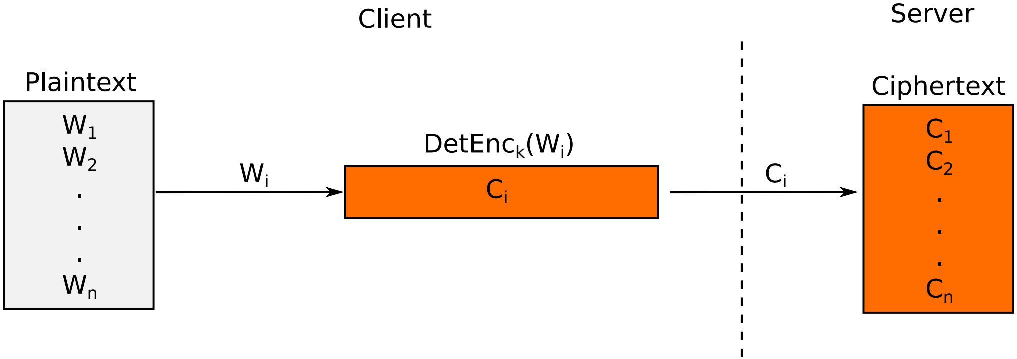
# Can we do better?

- Can we do better?

# Can we do better?

- Can we do better?
- Yes! We can perform deterministic encryption on keywords.

# Setup



Client

Server

Plaintext

$W_1$
$W_2$
.
.
.
$W_n$

$W_i$

$\mathrm{DetEnc}_k(W_i)$

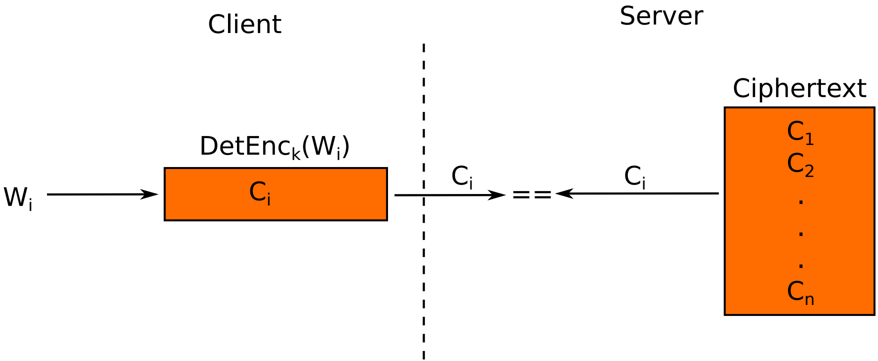
$C_i$

$C_i$

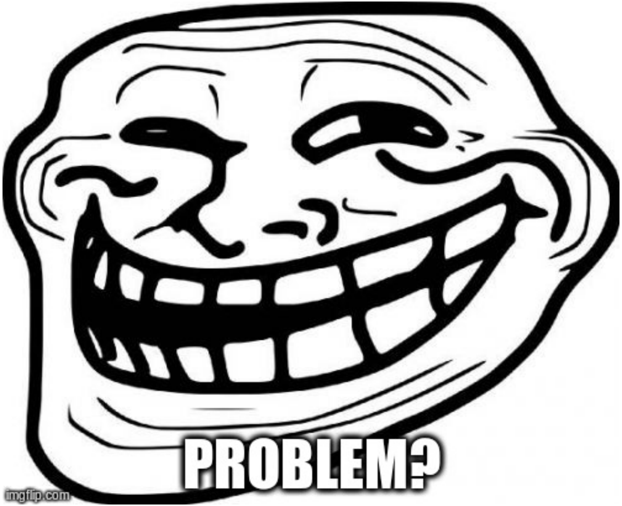
Ciphertext

$C_1$
$C_2$
.
.
.
$C_n$

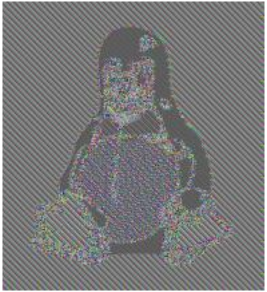
# Search



# Problem

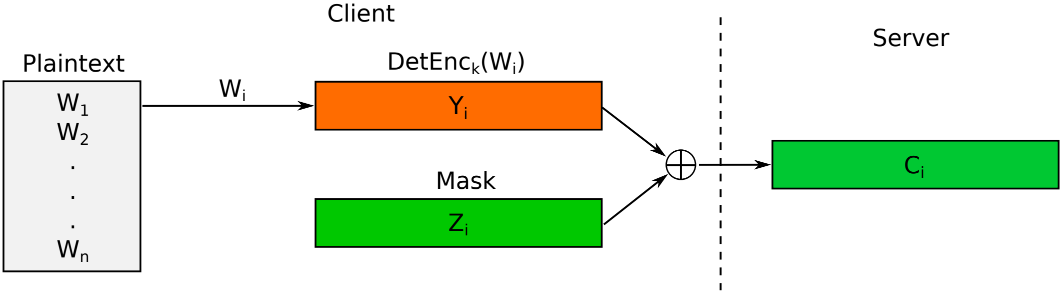


## Deterministic encryption sucks!



# Keyword-based Encryption (Song, Wagner, Perrig)

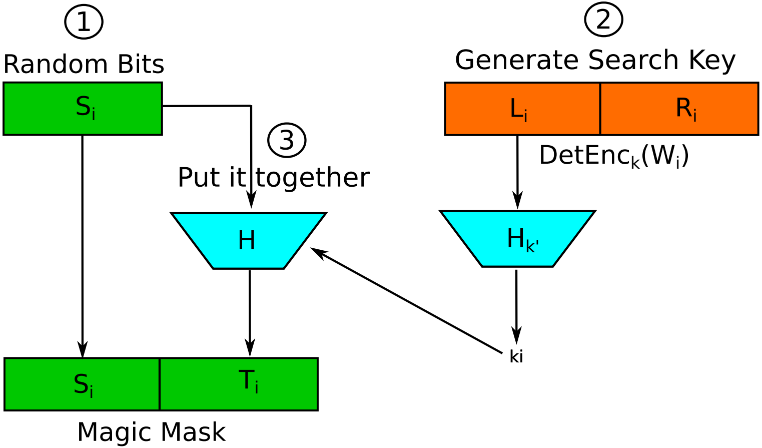
# Encrypt-then-Mask Approach



# Magic Mask

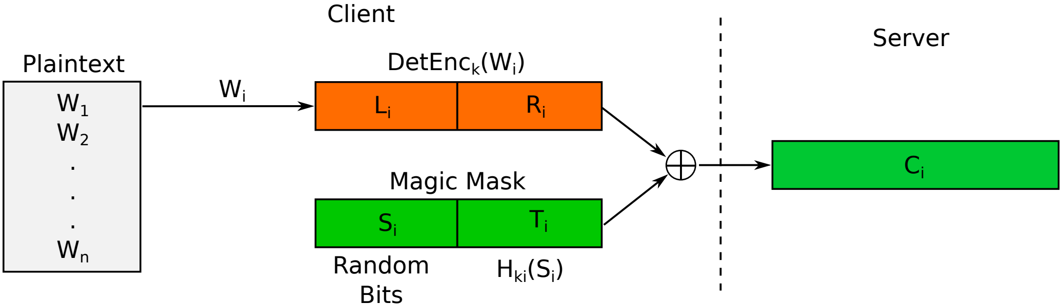
Symmetric Searchable Encryption (SSE) requires a magic mask.

# Let’s Craft a Magic Mask



# Let's do it

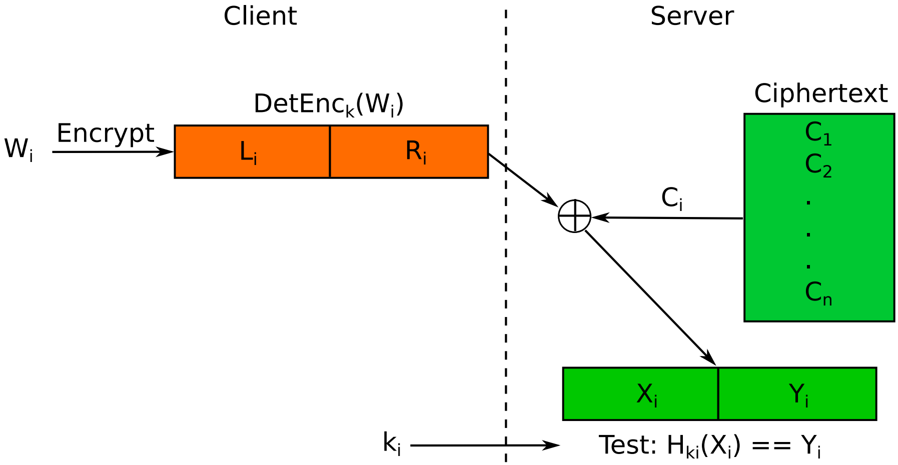
# Song Wagner Perrig (SWP) - Scheme



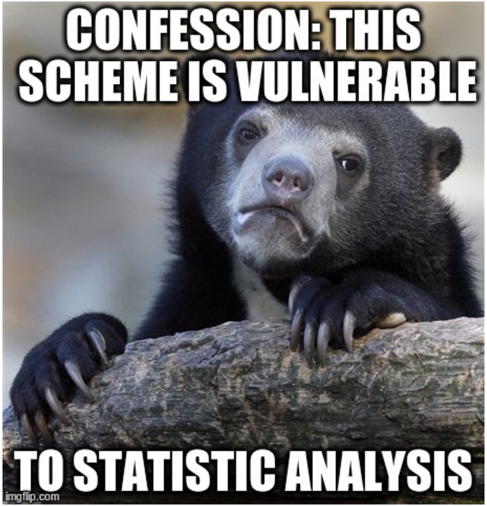
Search key $k_i = H_{k'}(L_i)$

Magic Mask: $T_i$ can be derived from $S_i$, i.e. $T_i = H_{ki}(S_i)$

# Search

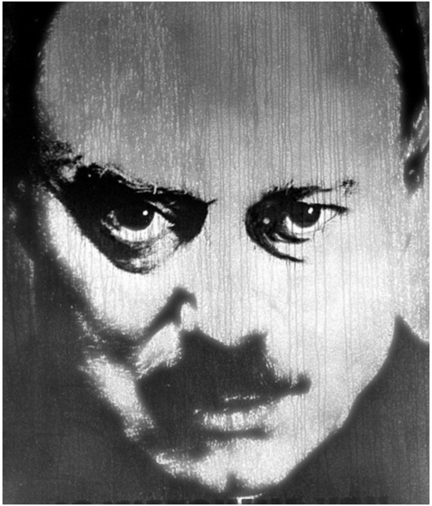

Client
Server
$W_i$
Encrypt
$\mathrm{DetEnc}_k(W_i)$
$L_i$
$R_i$
Ciphertext
$C_1$
$C_2$
.
.
.
$C_n$
$C_i$
$X_i$
$Y_i$
$k_i$
Test: $H_{ki}(X_i) == Y_i$

# Confession


CONFESSION: THIS SCHEME IS VULNERABLE
TO STATISTIC ANALYSIS
imgflip.com

# Statistical Analysis – Estimate Search Pattern

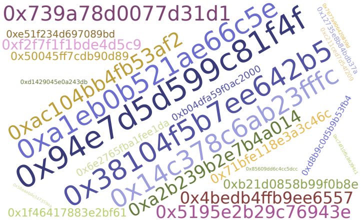
# Statistical Analysis – Monitor User Behaviour

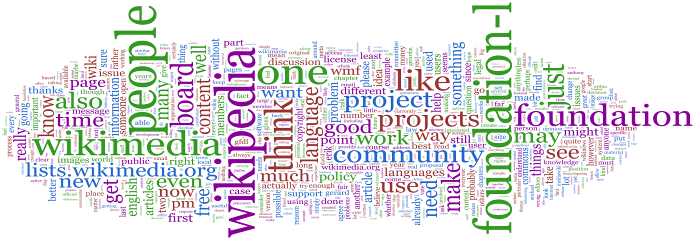


# Statistical Analysis – Monitoring Search Requests

0x739a78d0077d31d1
0xe51f234d697089bd
0xf2f7f1bde4d5c9
0x50045ff7cdb90d89
0xd1429045e0a243db
0xac104bb4fb53af2
0xa1eb0b521ae66c5e
0x94e7d5d599c81f4f
0x7677a906420859d
0x12735e8b84bdb37a
0xc211ced7d54f209
0xb04dfa59f0ac2000
0x38104f5b7ee642b5
0x14c378c6ab23fffc
0x6e2765fba1fee1da
0xa2b239b2e7b4a014
0x71bfe118e3a3c46c
0xd8b9c0d5b9b53fb4
0xc74f10f5c86394c1
0x85609dd6c4cc5dcc
0x58eaabde347359aa
0xb21d0858b99f0b8e
0x4bedb4ffb9ee6557
0x1f46417883e2bf61
0x5195e2b29c76943e

# Statistical Analysis – Compare





# Speed

Plaintext size (King James Bible): 4.3 MB

Ciphertext size: 25 MB

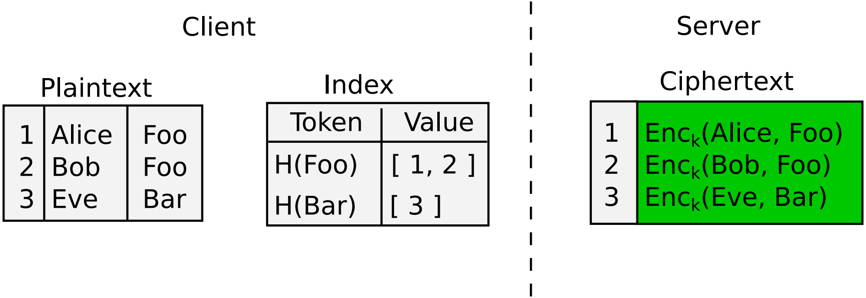
Time to encrypt: 0.211 sec

Search: 0.181 sec

- Foobar 0.181
- God 0.003
- towel 0.155
- Eve 0.005
- wrath 0.014
- dragon 0.094

# Index-based Searchable Encryption

# Plaintext Index - Search



Client

Plaintext

| | | |
|---|---|---|
| 1 | Alice | Foo |
| 2 | Bob | Foo |
| 3 | Eve | Bar |

Index

| Token | Value |
|---|---|
| H(Foo) | [ 1, 2 ] |
| H(Bar) | [ 3 ] |

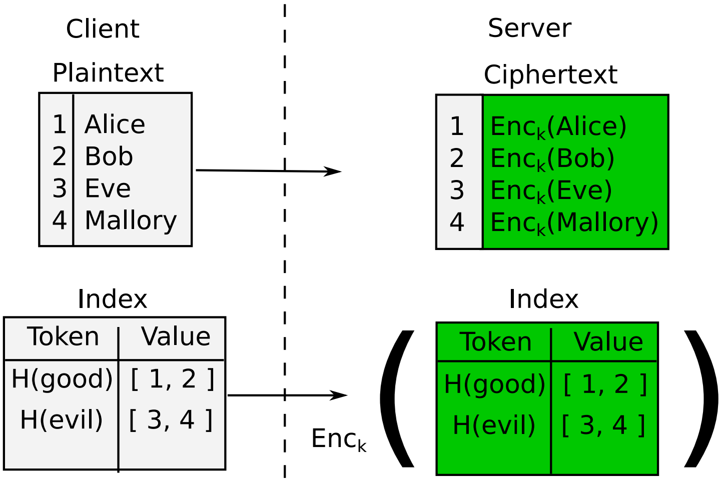
Server

Ciphertext

| | |
|---|---|
| 1 | Enc$_k$(Alice, Foo) |
| 2 | Enc$_k$(Bob, Foo) |
| 3 | Enc$_k$(Eve, Bar) |

# Plaintext Index - Hell of Synchronisation

# Encrypted Index - Setup



Client

Plaintext

| | |
|---|---|
| 1 | Alice |
| 2 | Bob |
| 3 | Eve |
| 4 | Mallory |

Index

| Token | Value |
|---|---|
| H(good) | [ 1, 2 ] |
| H(evil) | [ 3, 4 ] |

Server

Ciphertext

| | |
|---|---|
| 1 | $\text{Enc}_k$(Alice) |
| 2 | $\text{Enc}_k$(Bob) |
| 3 | $\text{Enc}_k$(Eve) |
| 4 | $\text{Enc}_k$(Mallory) |

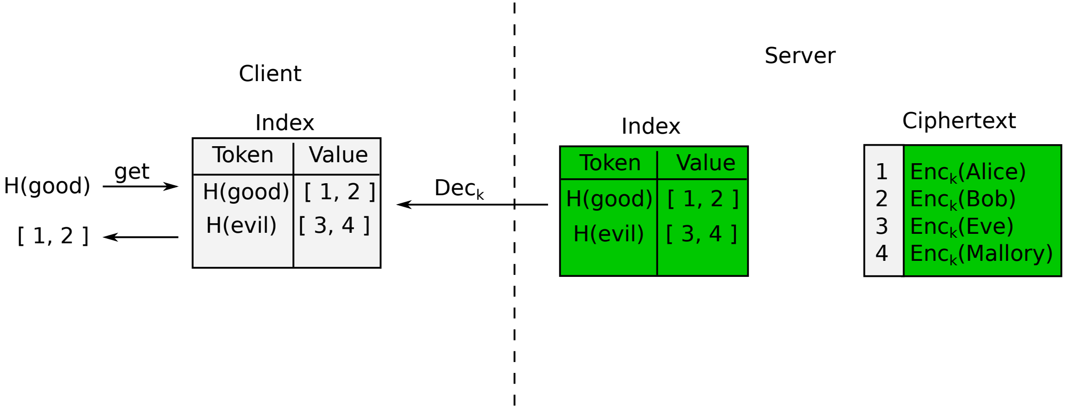
Index

$\text{Enc}_k$ (

| Token | Value |
|---|---|
| H(good) | [ 1, 2 ] |
| H(evil) | [ 3, 4 ] |

)

# Encrypted Index - Search



Client

Index

| Token | Value |
|---|---|
| H(good) | [ 1, 2 ] |
| H(evil) | [ 3, 4 ] |

Server

Index

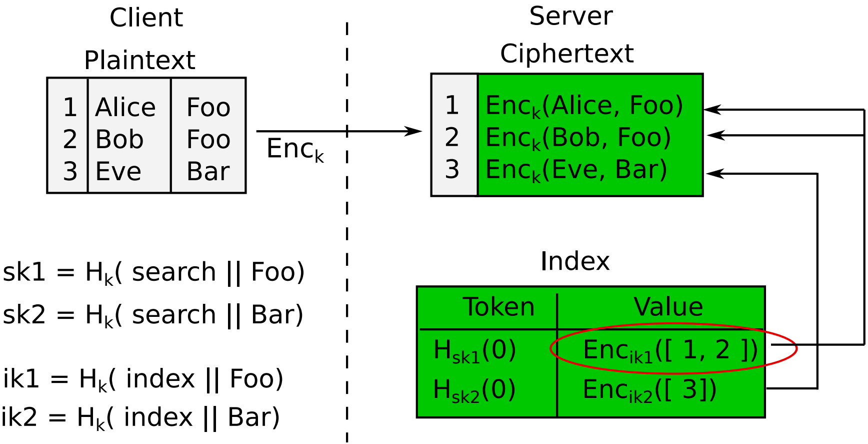
| Token | Value |
|---|---|
| H(good) | [ 1, 2 ] |
| H(evil) | [ 3, 4 ] |

Ciphertext

| | |
|---|---|
| 1 | Enc$_k$(Alice) |
| 2 | Enc$_k$(Bob) |
| 3 | Enc$_k$(Eve) |
| 4 | Enc$_k$(Mallory) |

# Communication Cost

# Encrypted-Index – Setup



Client

Plaintext

| | | |
|---|---|---|
| 1 | Alice | Foo |
| 2 | Bob | Foo |
| 3 | Eve | Bar |

$sk1 = H_k(\text{ search } || \text{ Foo})$

$sk2 = H_k(\text{ search } || \text{ Bar})$

$ik1 = H_k(\text{ index } || \text{ Foo})$

$ik2 = H_k(\text{ index } || \text{ Bar})$

Server

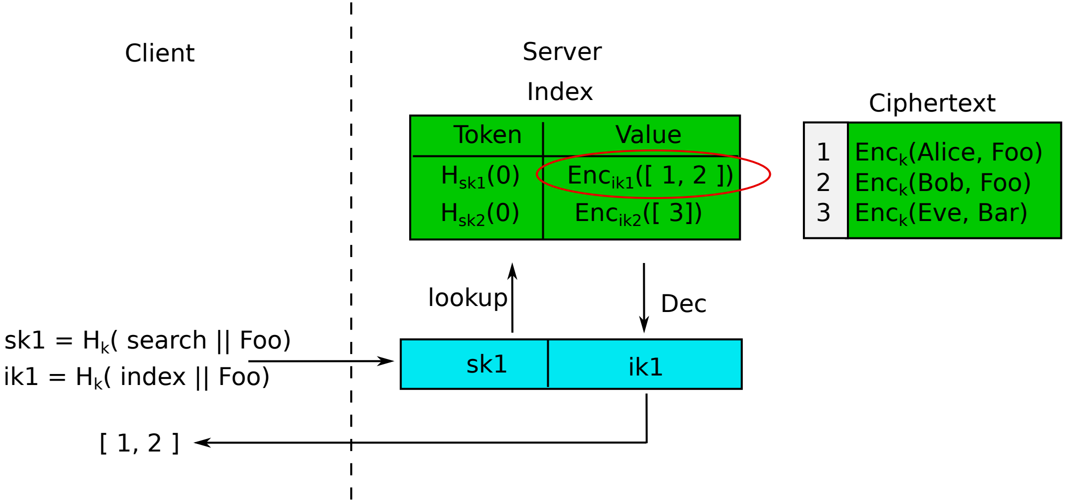
Ciphertext

| | |
|---|---|
| 1 | $\mathrm{Enc}_k(\text{Alice, Foo})$ |
| 2 | $\mathrm{Enc}_k(\text{Bob, Foo})$ |
| 3 | $\mathrm{Enc}_k(\text{Eve, Bar})$ |

Index

| Token | Value |
|---|---|
| $H_{sk1}(0)$ | $\mathrm{Enc}_{ik1}([\ 1, 2\ ])$ |
| $H_{sk2}(0)$ | $\mathrm{Enc}_{ik2}([\ 3])$ |

# Encrypted-Index – Search



Client

Server

Index

| Token | Value |
|---|---|
| $H_{sk1}(0)$ | $Enc_{ik1}([\ 1, 2\ ])$ |
| $H_{sk2}(0)$ | $Enc_{ik2}([\ 3])$ |

Ciphertext

| | |
|---|---|
| 1 | $Enc_k(Alice, Foo)$ |
| 2 | $Enc_k(Bob, Foo)$ |
| 3 | $Enc_k(Eve, Bar)$ |

lookup

Dec

sk1 = $H_k$( search || Foo)

ik1 = $H_k$( index || Foo)

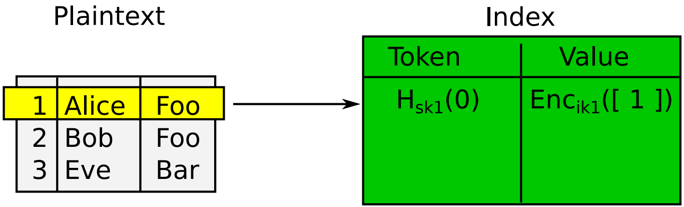
| sk1 | ik1 |
|---|---|

[ 1, 2 ]

# Encrypted-Index – Size matters

IF YOU DON'T HIDE THE SIZE
INSTRUCTO
YOU GONNA HAVE A BAD TIME

# Cash et al. – Setup



Plaintext

| | | |
|---|---|---|
| 1 | Alice | Foo |
| 2 | Bob | Foo |
| 3 | Eve | Bar |

Index

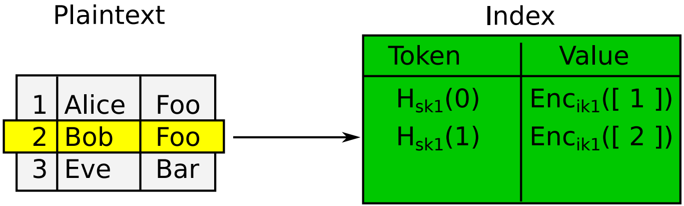
| Token | Value |
|---|---|
| $H_{sk1}(0)$ | $Enc_{ik1}([\ 1\ ])$ |

sk1 = $H_k$( search || Foo)

ik1 = $H_k$( index || Foo)

occurences["Foo"] = 0

# Cash et al. – Setup (contd.)



Plaintext

| | | |
|---|---|---|
| 1 | Alice | Foo |
| 2 | Bob | Foo |
| 3 | Eve | Bar |

Index

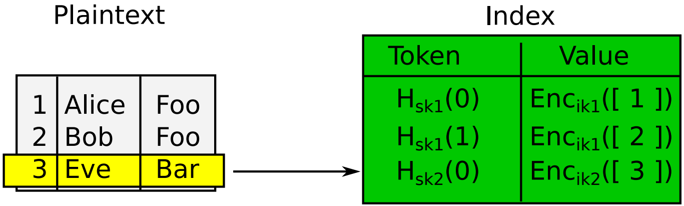
| Token | Value |
|---|---|
| $H_{sk1}(0)$ | $Enc_{ik1}([\ 1\ ])$ |
| $H_{sk1}(1)$ | $Enc_{ik1}([\ 2\ ])$ |

sk1 = $H_k$( search || Foo)

ik1 = $H_k$( index || Foo)

occurences["Foo"] = 1

# Cash et al. – Setup (contd.)



Plaintext

| | | |
|---|---|---|
| 1 | Alice | Foo |
| 2 | Bob | Foo |
| 3 | Eve | Bar |

Index

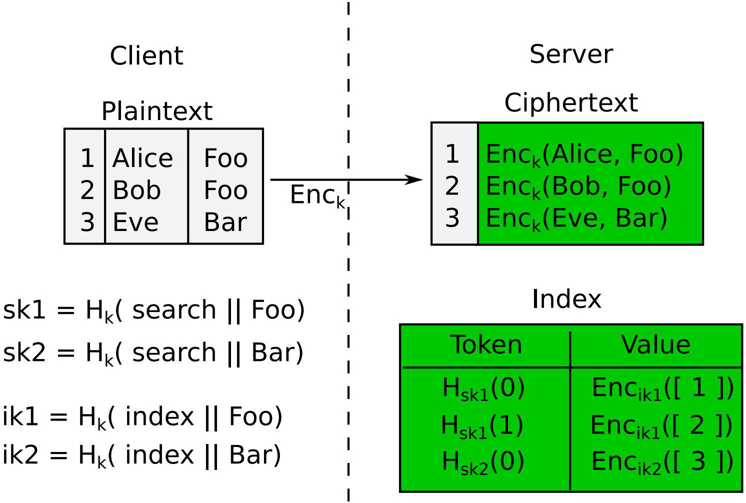
| Token | Value |
|---|---|
| $H_{sk1}(0)$ | $Enc_{ik1}([\ 1\ ])$ |
| $H_{sk1}(1)$ | $Enc_{ik1}([\ 2\ ])$ |
| $H_{sk2}(0)$ | $Enc_{ik2}([\ 3\ ])$ |

$sk2 = H_k(\text{ search } || \text{ Bar})$

$ik2 = H_k(\text{ index } || \text{ Bar})$

occurences["Bar"] = 0

# Cash et al. – Basic Scheme



Client

Plaintext

|   |       |     |
|---|-------|-----|
| 1 | Alice | Foo |
| 2 | Bob   | Foo |
| 3 | Eve   | Bar |

$\xrightarrow{\text{Enc}_k}$

Server

Ciphertext

|   |   |
|---|---|
| 1 | $\text{Enc}_k(\text{Alice, Foo})$ |
| 2 | $\text{Enc}_k(\text{Bob, Foo})$ |
| 3 | $\text{Enc}_k(\text{Eve, Bar})$ |

$sk1 = H_k(\text{ search } \| \text{ Foo})$

$sk2 = H_k(\text{ search } \| \text{ Bar})$

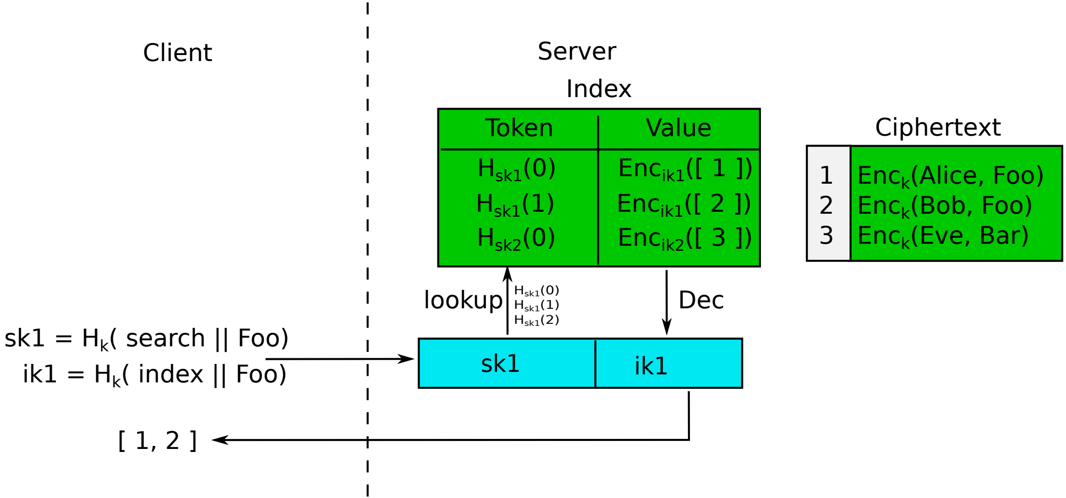
$ik1 = H_k(\text{ index } \| \text{ Foo})$

$ik2 = H_k(\text{ index } \| \text{ Bar})$

Index

| Token | Value |
|-------|-------|
| $H_{sk1}(0)$ | $\text{Enc}_{ik1}([\ 1\ ])$ |
| $H_{sk1}(1)$ | $\text{Enc}_{ik1}([\ 2\ ])$ |
| $H_{sk2}(0)$ | $\text{Enc}_{ik2}([\ 3\ ])$ |

# Cash et al. – Search



| Token | Value |
|---|---|
| $H_{sk1}(0)$ | $Enc_{ik1}([\ 1\ ])$ |
| $H_{sk1}(1)$ | $Enc_{ik1}([\ 2\ ])$ |
| $H_{sk2}(0)$ | $Enc_{ik2}([\ 3\ ])$ |

| | |
|---|---|
| 1 | $Enc_k$(Alice, Foo) |
| 2 | $Enc_k$(Bob, Foo) |
| 3 | $Enc_k$(Eve, Bar) |

sk1 = $H_k$( search || Foo)

ik1 = $H_k$( index || Foo)

[ 1, 2 ]

# Cash et al. – Speed

Plaintext size (King James Bible): 4.3 MB

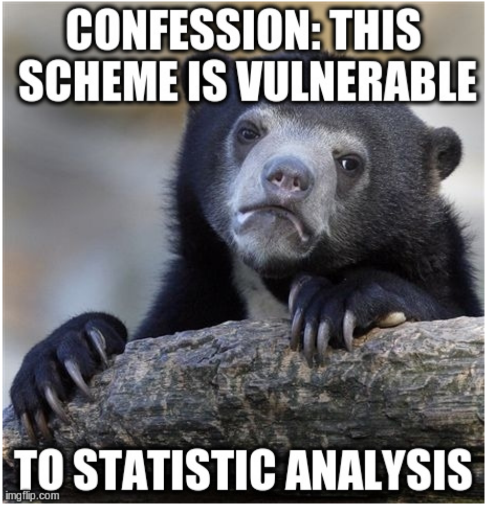
Ciphertext size: 4.3 MB

Index size: 0.125 MB

Time to encrypt: 0.108 sec

Time to search: 0.001 sec

# Cash et al. – Confession



# Outlook & Conclusions

# Outlook

- So far: deterministic search token → statistical analysis

# Outlook

- So far: deterministic search token → statistical analysis
- Making existing approaches practical is a challenge (e.g. FHE)

# Outlook

- So far: deterministic search token → statistical analysis
- Making existing approaches practical is a challenge (e.g. FHE)
- Implement and adapt!!1

# Outlook

- So far: deterministic search token → statistical analysis
- Making existing approaches practical is a challenge (e.g. FHE)
- Implement and adapt!!1
- Let's Encrypt!

# Conclusions

- Presented schemes and their properties

# Conclusions

- Presented schemes and their properties
    - Deterministic Encryption

# Conclusions

- Presented schemes and their properties
  - Deterministic Encryption
    - Fast setup

# Conclusions

- Presented schemes and their properties
    - Deterministic Encryption
        - Fast setup
        - Insecure search

# Conclusions

- Presented schemes and their properties
  - Deterministic Encryption
    - Fast setup
    - Insecure search
  - Keyword (Song, Wagner, Perrig)

# Conclusions

- Presented schemes and their properties
  - Deterministic Encryption
    - Fast setup
    - Insecure search
  - Keyword (Song, Wagner, Perrig)
    - Search is in O(n)

# Conclusions

- Presented schemes and their properties
    - Deterministic Encryption
        - Fast setup
        - Insecure search
    - Keyword (Song, Wagner, Perrig)
        - Search is in O(n)
    - Index (Cash et al.)

# Conclusions

- Presented schemes and their properties
  - Deterministic Encryption
    - Fast setup
    - Insecure search
  - Keyword (Song, Wagner, Perrig)
    - Search is in O(n)
  - Index (Cash et al.)
    - Search is in O(1)

# Conclusions

- Presented schemes and their properties
  - Deterministic Encryption
    - Fast setup
    - Insecure search
  - Keyword (Song, Wagner, Perrig)
    - Search is in O(n)
  - Index (Cash et al.)
    - Search is in O(1)
    - Index maintenance needed (think: Update)

# Conclusions

- Presented schemes and their properties
  - Deterministic Encryption
    - Fast setup
    - Insecure search
  - Keyword (Song, Wagner, Perrig)
    - Search is in O(n)
  - Index (Cash et al.)
    - Search is in O(1)
    - Index maintenance needed (think: Update)
- slightly different features

# Conclusions

- Presented schemes and their properties
  - Deterministic Encryption
    - Fast setup
    - Insecure search
  - Keyword (Song, Wagner, Perrig)
    - Search is in O(n)
  - Index (Cash et al.)
    - Search is in O(1)
    - Index maintenance needed (think: Update)
- slightly different features
- More exist!

# Conclusions

- Presented schemes and their properties
  - Deterministic Encryption
    - Fast setup
    - Insecure search
  - Keyword (Song, Wagner, Perrig)
    - Search is in O(n)
  - Index (Cash et al.)
    - Search is in O(1)
    - Index maintenance needed (think: Update)
- slightly different features
- More exist!
- Searching on encrypted data is practical

# Thanks!

# References

- Dawn Xiaodong Song, David Wagner, Adrian Perrig: Practical Techniques for Searches on Encrypted Data. IEEE Symposium on Security and Privacy 2000: 44-55
- David Cash, Joseph Jaeger, Stanislaw Jarecki, Charanjit S. Jutla, Hugo Krawczyk, Marcel-Catalin Rosu, Michael Steiner: Dynamic Searchable Encryption in Very-Large Databases: Data Structures and Implementation. NDSS 2014