

Harnessing Encrypted Data in Cloud for Secure and Efficient Image Sharing from Mobile Devices

Helei Cui, Xingliang Yuan, and Cong Wang

**Department of Computer and Science
City University of Hong Kong**

April 30th, 2015

Media Data are Ubiquitous

- In 2014, millions of media data are generated in every minute.

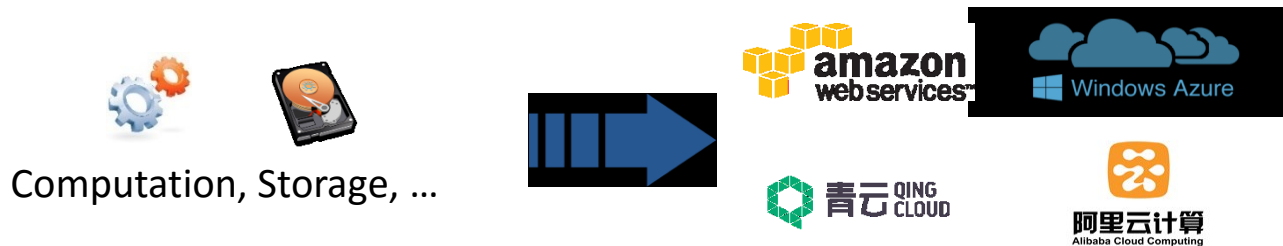


- **INSTAGRAM** users post 216,000 images;
- **WHATSAPP** users share 347,222 images;
- **PINTEREST** users pin 3,472 images;
- **YOUTUBE** users upload 72 hours new videos;
- **VINE** users share 8,333 videos;
- ...

**Data Never Sleeps 2.0, Domo Inc.*

<http://www.domo.com/learn/data-never-sleeps-2>

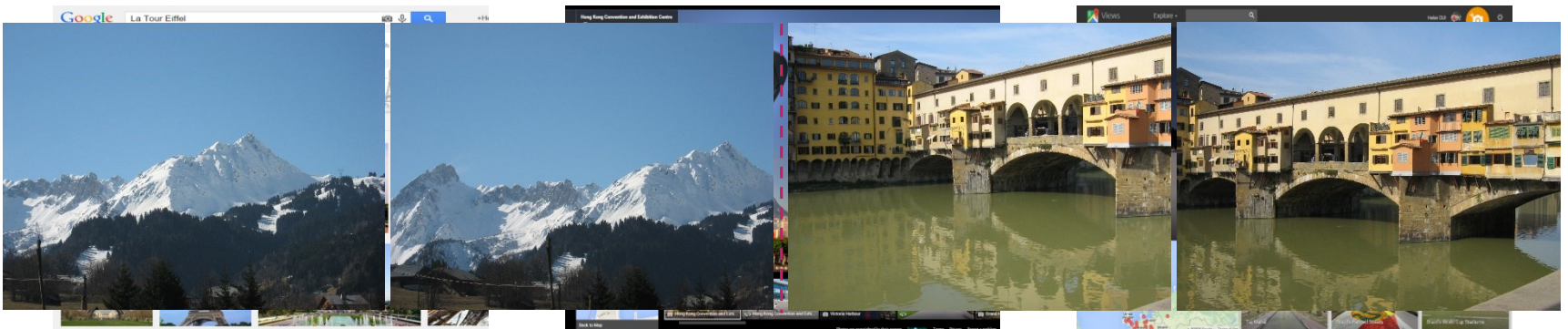
- Many applications utilize public cloud as backend.
 - for storage, processing, delivery, etc.



Exposing content-sensitive data to cloud raises **privacy concerns**.

And Meanwhile ...

- Correlated images occur quite commonly in online image repositories.
 - Images with slightly different viewing angles, resolutions, or qualities.



- Various applications have already leveraged such correlations.
 - E.g., reduce media storage [Zheng et. al AsiaCCS'2015], media transcoding [Fan et. al ICASSP'2009], even image encoding [Yue et. al TMM'2013], etc.

Can we securely leverage the image **correlation** to **save** the cost of original image transmission from mobile devices?

A demo of major steps

1. Secure digest generation;
2. Secure candidate selection;
3. Encrypted image reproduction.



Original image transmission could be saved.



Target scenario: You may want to securely share photos with friends, but the international data roaming can be expensive.

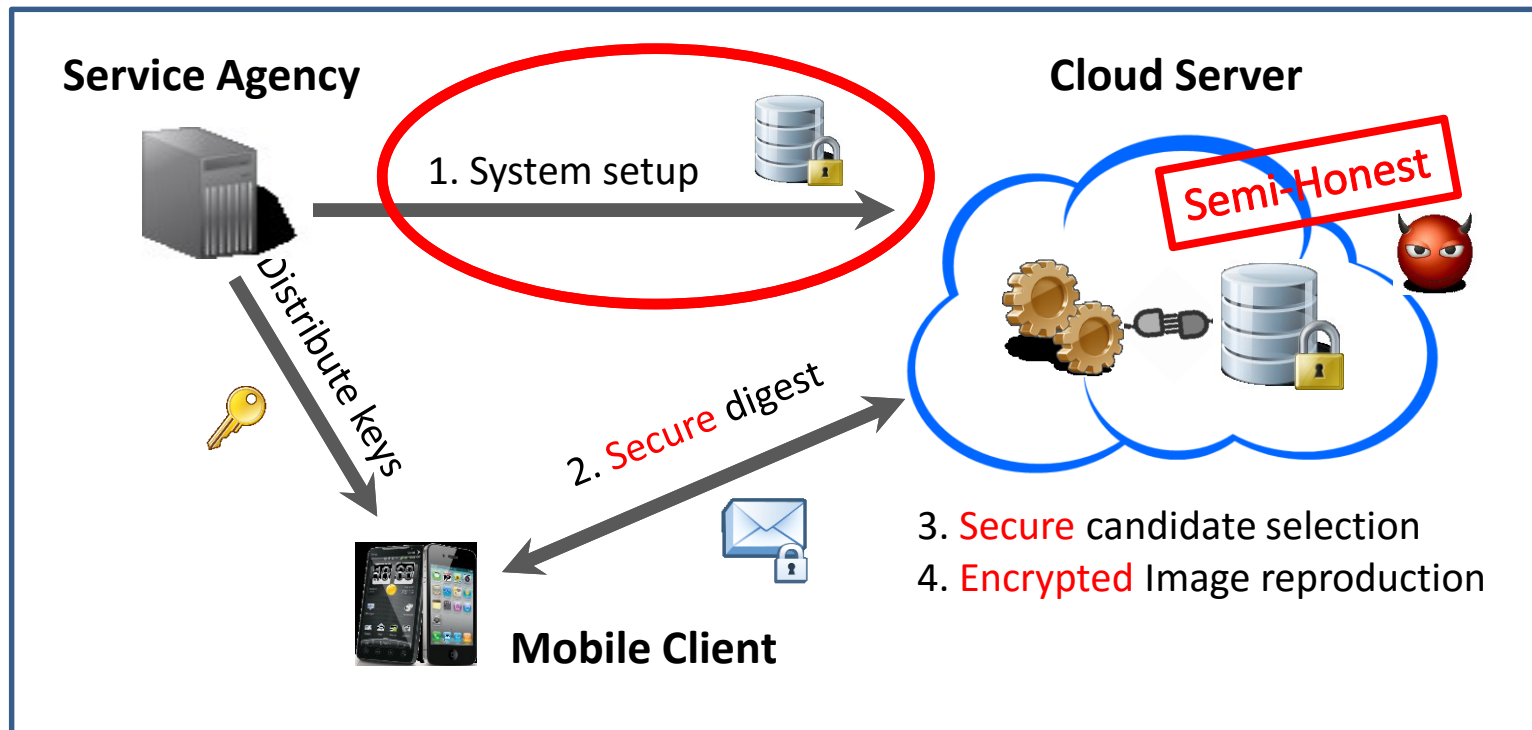
Two challenging subtasks:

- Securely locate encrypted **correlated** image candidates.
- Secure image reproduction at cloud via **encrypted** candidates.

The desirables:

- **Lightweight** computation at client (usually mobile)
- **Compact** data transmission
- Security

System Overview

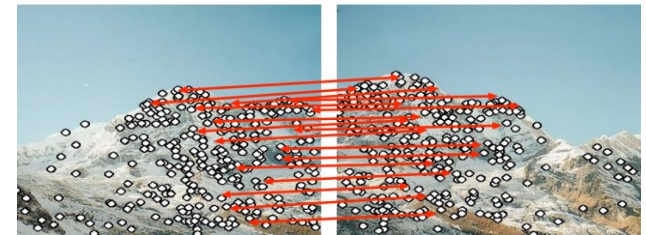


We assume the correlated image datasets are available at cloud.

System Initialization

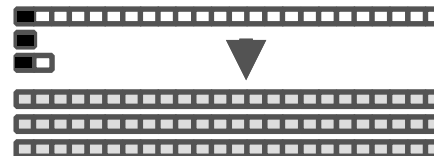
- Need to build an encrypted image database
 - to securely and efficiently locate the candidates.
 - use local feature (e.g., SIFT) and measure the closeness.
 - Adopted in many applications, e.g., object recognition.
 - More feature matches, more similar.

[Brown et al. IJCV'2007]



- Initial attempt
 - Leverage searchable symmetric encryption (SSE)?
 - Use locality-sensitive hash (LSH) to hash the features, treat the hash values as keywords fed into SSE framework. *[Kuzu et al. ICDE'2012]*
 - But direct combination does not necessarily support large datasets.
 - E.g., thousands of features per image, and thousands of images.

One bad exemplary case of
the encrypted inverted index



Secure & Efficient Searching Table

- We explore space efficient SSE [Cash et al. NDSS'2014].
 - Based on generic dictionary \mathcal{D} (vertical design);
 - Treat each LSH keyword as an independent value;
 - Generate multiple key-value pairs, where key is converted from the LSH keyword, and the value contains the image/feature id.
- Padding can be avoided.
- Our construction:
 - For each feature f , compute LSH values:

$$\mathbf{v} = \{g_1(f) || 1, \dots, g_l(f) || l\}, \text{ where } v_i = g_i(f) || i;$$
 - For each v in \mathbf{v} :

$$K_1 \leftarrow P(K_v, 1 || v), K_2 \leftarrow P(K_r, 2 || v);$$

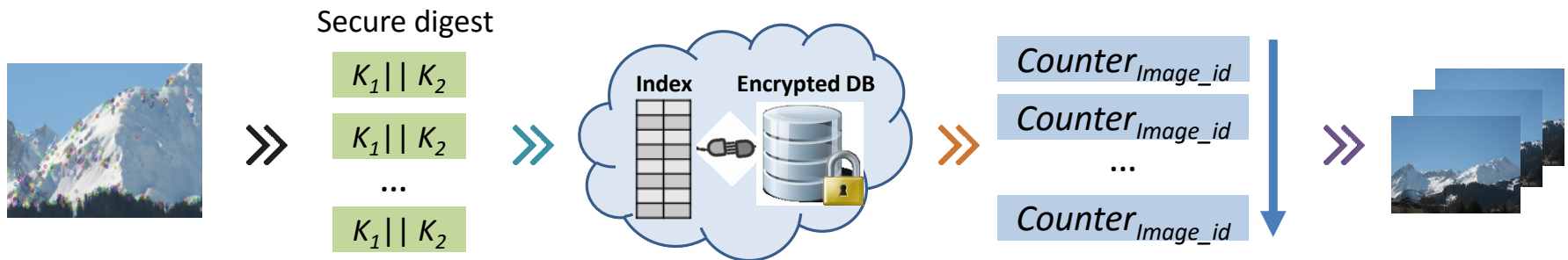
$$a \leftarrow F(K_1, c), b \leftarrow Enc(K_2, f_{id}),$$
 - $f_{id} = Image_{id} || feature_{id}$.

Key	Value
$a \leftarrow F(K_1, c)$	$b \leftarrow Enc(K_2, f_{id})$
...	...
...	...
...	...
...	...
...	...
...	...

Generic Dictionary

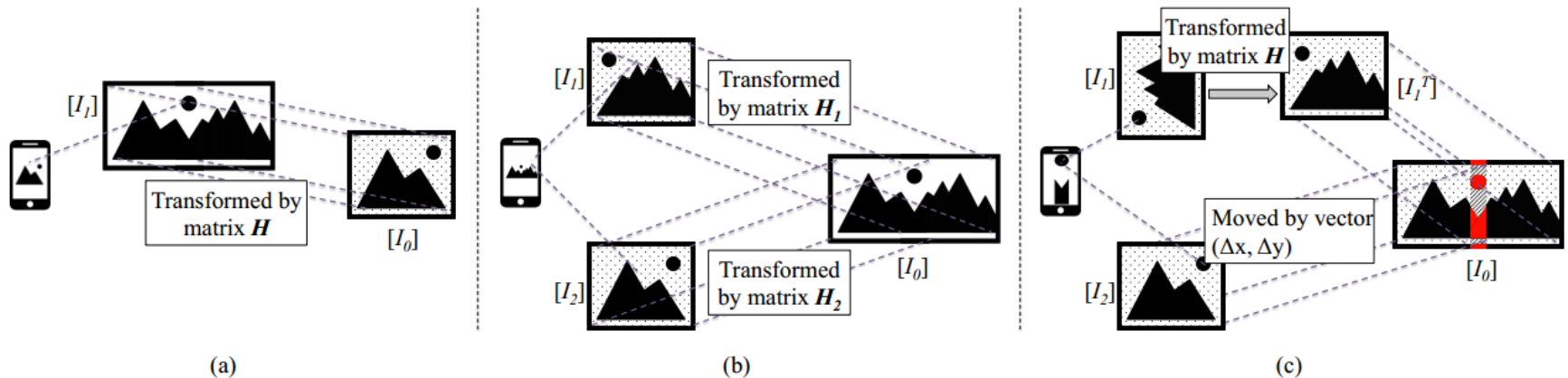
Candidate Selection

- Mobile client sends a compact secure digest ($\{K_1, K_2\}$):
 - Generated from the features of the image of interest;
 - For securely locating matched features at cloud.
- **Voting-based ranking mechanism**: the similarity between two images can be measured by the number of matched features.
 - More feature matches, more similar. [Brown et al. IJCV'2007]
 - Cloud locates the matched encrypted features ($f_{id} = \text{Image}_{id} || \text{feature}_{id}$).
 - Cloud ranks the frequency of Image_{id} to get **top-k** candidates.



Encrypted Image Reproduction

- With the candidates, different possible ways to reconstruct the images can be supported:



- Need to instruct the cloud to reconstruct images from the candidates.
 - Usually it is a regular polygon area;
 - Can be measured by geometric transformation;
 - Denoted as a 3x3 matrix H .

Matrix	Distortion
$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$	
$\begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$	
$\begin{bmatrix} sr_{11} & sr_{12} & t_x \\ sr_{21} & sr_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$	
$\begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$	

Encrypted Image Reproduction (Cont'd)

- Computing H is to eliminate false positive and estimate geometric transformation:

X Directly compute H at cloud in ciphertext domain is not practical.

- E.g., fully homomorphic encryption.















Mobile client can efficiently compute H :

- Candidate size is small (e.g., < 5);
- Feature descriptors are small (76 KB for 500 features);
- The result H is very compact, 36 bytes.

H #1000 features (ms)	H #500 features (ms)
64	24

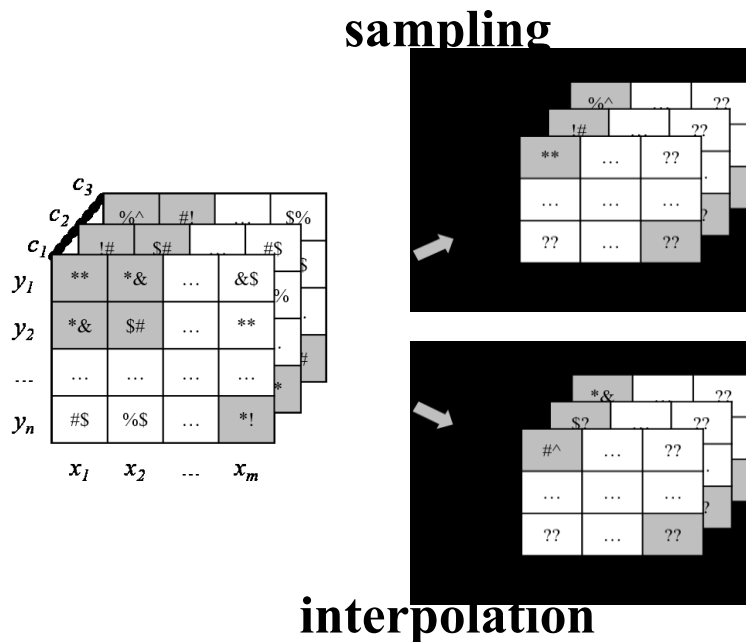
**Test on iPhone 6.*

No.	Highly Correlated Candidates		Decrypted Result
a			
b			
c			
d			

**Examples of newly generated image without cropping.*

Two Approaches

- Knowing H , existing image processing techniques can be adopted in pixel/patch level.



Manipulate the position of each pixel,
e.g., replace, select, removal, etc.

Symmetric Encryption

e.g., AES, Blowfish.

Require computation on pixels.

Semi-homomorphic Encryption

e.g., Paillier cryptosystem.

(we have discussed how to pack multiple values to reduce the storage space.)

Security Analysis

- Image content and features are protected in encrypted forms with semantic security along the service flow.
- **Interaction** in candidate selection, following the security framework of SSE [Curtmola et.al CCS'2006]
 - Simulation based security definition:
 - Real world: conduct real protocol Ω for candidate selection;
 - Ideal world: apply ideal function \mathcal{F} to simulate the service flow.

Adversary *should not be able to differentiate* the real interactions from Ω and the simulated outputs by applying \mathcal{F} .

Real table \mathcal{D}

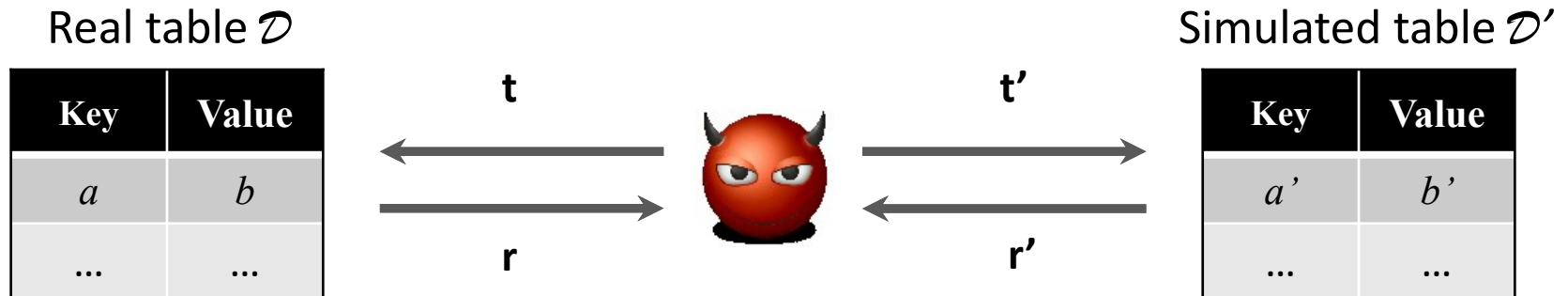
Key	Value
a	b
...	...
...	...



Simulated table \mathcal{D}'

Key	Value
a'	b'
...	...
...	...

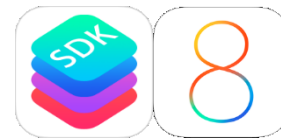
Security Analysis



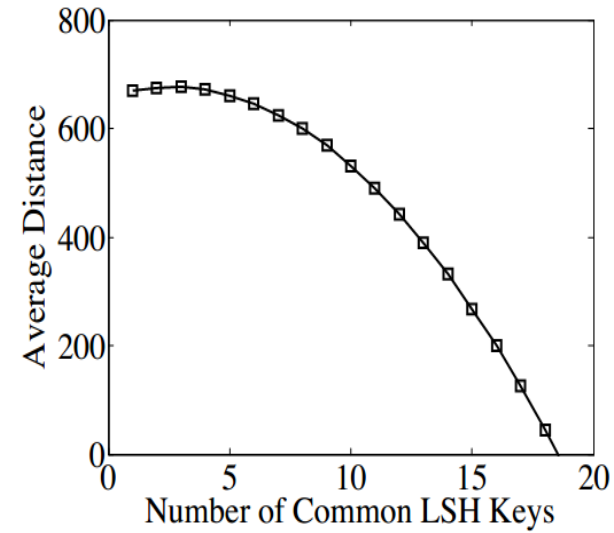
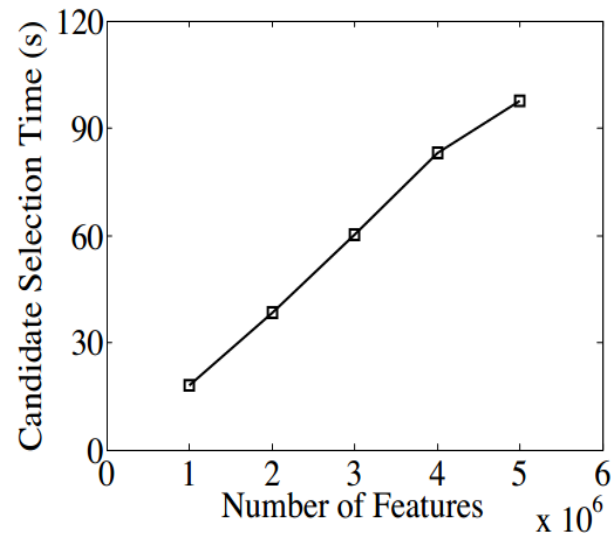
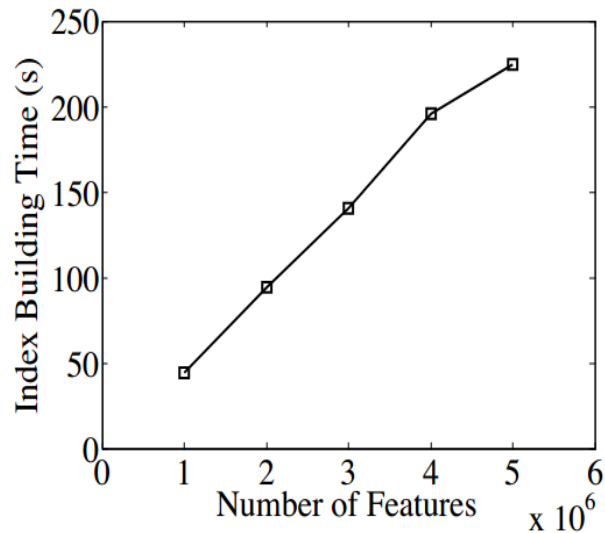
- Quantify the leakage functions (L_1, L_2) in candidate selection:
 - $L_1: (N, [\mathbf{f}], |[\mathbf{f}]|)$, where N is the number of key-value pairs;
 - $L_2: (\{\mathbf{t}\}_q, \{f_{id}\}, \{[\mathbf{f}]\})$, where q is the number of adaptive queries.
- Simulate a query on a simulated searching table:
 - Generates random strings to simulate secure digest \mathbf{t}' ;
 - Returns identical number of feature packages \mathbf{r}' from L_2 ;
 - Achieve (L_1, L_2) -secure against adaptive attacks in random oracle model:
 - Replace the **PRF** with the random oracle $H_1: P(K, v) := H_1(K||v)$;
 - The encryption algorithm **Enc**, on input K, f_{id} , chooses a random $r \in \{0,1\}^\lambda$, and outputs $(r, H_2(K||r) \oplus f_{id})$, where H_2 is another random oracle.

Experiment Evaluation

- AWS server “c3.4xlarge”
- iOS 8.1 SDK
- Java 1.7 SDK
 - Java Cryptography Architecture
- OpenCV 2.4.10
- INRIA Holiday dataset
 - 1491 images, where numbers of images contain with overlapped areas
- MIRFLICKR-25K
 - select 10,000 images

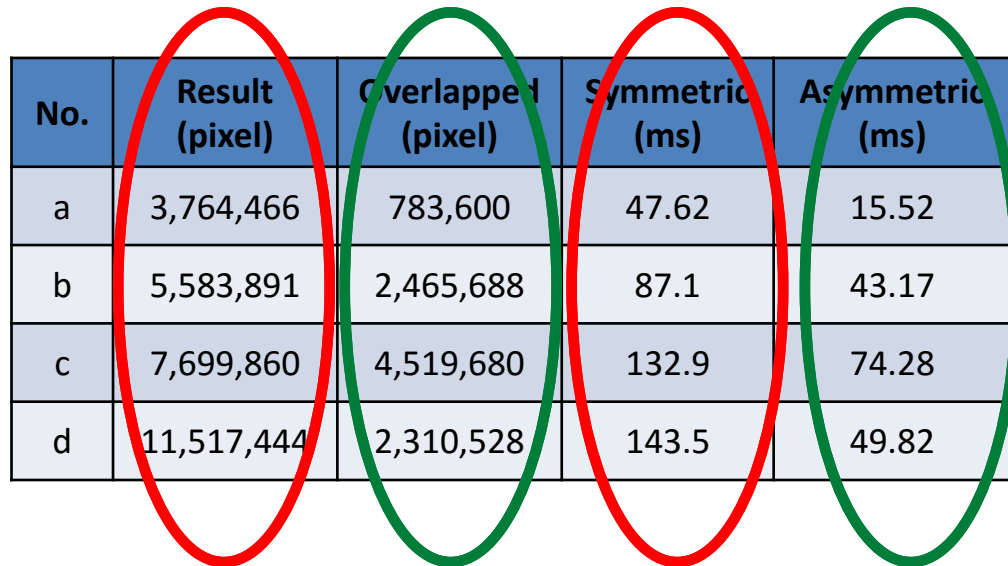


Efficient and Effective Searching Table



- Both the index building time and candidate selection time are **in linear to the size of dataset**.
- The more common LSH keywords the two features share, the more similar they are.
 - Overall accuracy can be guaranteed.

Encrypted Image Reproduction is Fast



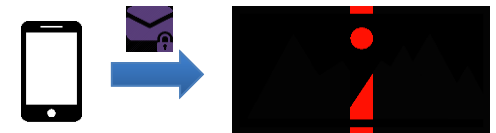
No.	Result (pixel)	Overlapped (pixel)	Symmetric (ms)	Asymmetric (ms)
a	3,764,466	783,600	47.62	15.52
b	5,583,891	2,465,688	87.1	43.17
c	7,699,860	4,519,680	132.9	74.28
d	11,517,444	2,310,528	143.5	49.82

- The time cost of symmetric key based approach is **positively correlated with the result size**;
- But for the other one, it is **positively correlated with the overlapped size**.

Bandwidth Saving

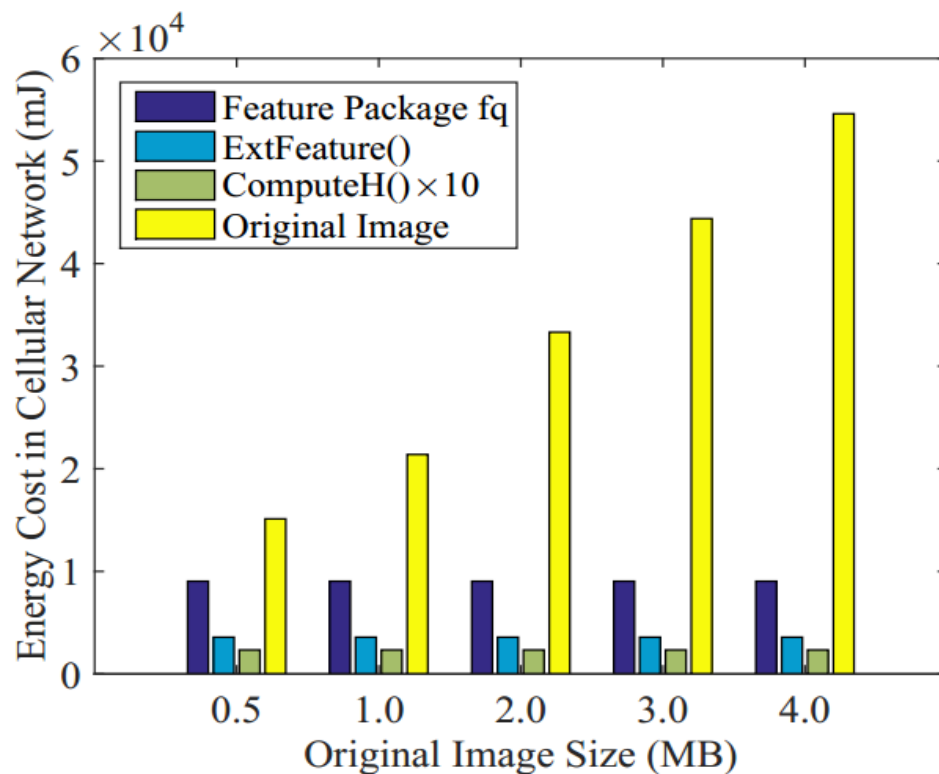
No.	t (KB)	r (KB)	H (Byte)	Result (KB)	Overall Saving
a	39.06	380	36	2309.6	81.9%
b	39.06	380	36	2494.2	83.2%
c	39.06	456	36	2105.6	76.5%
d	39.06	456	36	5548.3	91.1%
*Avg.	39.06	304	36	2764.8	87.6%

- Up to 90% can be saved, compared with the original image size (~2.7MB) in JPEG.
 - Assuming sufficient amount of highly correlated images available at cloud.
 - Mainly depends on the *top-k* candidates.



*Avg. is estimated by the setting ($l=5$, $m=200$, $k=4$), and the result size is 2.7 MB.

Energy Saving (in full version)



- Our design can indeed bring the energy saving, when considering all computations and data transmission.
 - Can be saved from **1.5X** to over **5X**.

**Test on Google Nexus 5 by using the App, Power Tutor 2 Pro.*

Conclusion and Future Works

- Summary:
 - Our system securely leverages the image correlation to save the bandwidth and energy cost of original image transmission from mobile devices.
- Future works:
 - Further reduce the bandwidth, e.g., increase the feature quality to decrease the number of the features.
- Thank you! Questions?

helei.cui@my.cityu.edu.hk