Energy Savings in Privacy-Preserving Computation Offloading with Protection by Homomorphic Encryption

Jibang Liu and Yung-Hsiang Lu
School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907, USA

ABSTRACT

This paper investigates energy savings on mobile systems in privacy-preserving computation offloading. Offloading computation-intensive programs to servers can save energy but data must be protected for privacy concerns. The protection schemes must guarantee operations performed on the protected data remain meaningful and the results are still acceptable. The protection cannot require excessive amounts of energy overhead. We propose to adopt homomorphic encryption to protect data in image retrieval before sending data to servers. We implement our method on a PDA and evaluates the retrieval performance and energy savings.

1. INTRODUCTION

In recent years, mobile systems become primary platforms for many users, but mobile systems have limited energy. *Offloading* migrates computation from mobile systems to servers and can extend the former's battery life [1, 2, 3, 4]. However, offloading creates challenges about privacy because data are stored in servers and no longer under users' control. Previous studies [1, 2, 3, 4] mostly focus on how to make decisions to offload and save energy. Some studies consider how to protect privacy in data outsourcing [5]. However, none of them considers energy consumption when protecting privacy in computation offloading.

To ensure privacy, data must be properly protected before being sent to a server. Steganography [6] and encryption are two common data-hiding techniques. Steganography hides data so that the server is unaware of the existence of information. Ordinary encryption is commonly used to transmit secret data, but it is inapplicable in offloading because data must be decrypted at the server before performing computation. Researchers propose *homomorphic encryption* [7, 8, 9] that allows computation on encrypted data. These protection methods must pay different amounts of energy overhead to protect privacy.

Image retrieval [10] finds images similar to a query by extracting images' features and comparing the features. It is computation intensive and thus a good candidate for offloading to save energy. ImgSeek [11] and Gabor filtering [12] are both approaches for image retrieval. Offloading image retrieval requires that images be protected before they are sent to servers, and the operations on protected images must be still meaningful. In our previous work [13] we showed that ImgSeek can find similar images that are protected by steganography. However, ImgSeek is sensitive to orientation. In other words, if objects in an image are rotated, ImgSeek often fails to find similar images. In contrast, Gabor filtering can be extended for orientation-invariant retrieval [12]. To protect data in Gabor retrieval, steganography is not suitable any more. We use homomorphic encryption to protect data in Gabor retrieval.

This paper proposes an approach to save energy by offloading image retrieval with privacy protection. We build an energy model for privacy-preserving computation offloading and examine how to protect privacy in image retrieval without sacrificing retrieval accuracy. We demonstrate that homomorphic encryption can be adopted to protect data when offloading the Gabor filtering retrieval algorithm. We evaluate retrieval performance (i.e. accuracy) and energy consumption on an HP PDA. Results show that our method can save different amounts of energy with different degrees of protection.

2. RELATED WORK

Computation offloading has been widely studied. Several studies determine whether to offload and what computation to offload. Rong and Pedram [1] use stochastic models to determine whether to offload. Wolski et al. [2] and Hong et al. [3] decide whether to offload based on network bandwidths. Li et al. [4] use cost graphs from program analysis to decide which parts to offload. None of them considers energy consumption for protecting privacy in computation offloading.

ImgSeek [11] is an image retrieval program that performs two-dimensional Haar wavelet decompositions on the images and then finds 60 coefficients with the largest amplitude as features. Gabor filtering [12] extracts image features at different scales and orientations to compare similarities. The steps includes: ($S1$) applying Gabor filters on an image with different scales and orientations to obtain filtered images and then calculating the array of magnitudes of every filtered image; and ($S2$) generating features from the magnitudes of the filtered image at every scale and orientation and comparing features.

Homomorphic encryption [7, 8, 9] allows computations performed on data without decryption. Gentry [7] has recently proved that it is possible to construct a fully homomorphic encryption scheme, but the efficient homomorphic encryption is still a subject of current research.

Only some limited classes of homomorphisms such as additive, multiplicative, and mixed-multiplicative [8] are found in practice so far. The encryption in [9] supports additive, multiplicative and mixed-multiplicative homomorphisms; similar values in plaintext become very different in ciphertext. It encrypts a number x with keys p, q by picking a random integral r so that the encrypted value $y = Encrypt(x) = (x + rp) \mod N$, here $N = pq$. The decryption uses the key p to recover $x = Decrypt(y) = y \ mod \ p$. This encryption is homomorphic only in the range of integers. Thus, non-integral values must be converted integers to make the homomorphism hold.

This paper investigates energy conservation for protecting privacy in computation offloading. We adopt homomorphic encryption to protect data for offloading a Gabor retrieval program. Images are encrypted before they are sent to a server. On the server, the retrieval program never decrypts the data so privacy is ensured.

3. PRIVACY-PRESERVING COMPUTATION OFFLOADING

Computation offloading may save energy on mobile systems. However, data must be protected before being sent to a server and results in energy overhead. In this section, we first build an energy model for privacy-preserving computation offloading and examine the properties of different image retrieval algorithms and protection methods. Then we propose an approach to save energy by offloading image retrieval with data protected by homomorphic encryption.

3.1 Energy Model

There are four options to run a program C as shown in Figure 1. We build the energy model for each option. Table 1 shows some parameters; $comp(.)$ and $size(.)$ are used to denote the amount of computations and the size of transmitted data.

Figure 1 (a) shows the option of running the program without offloading. The program C is run on a mobile system with input data D and generates a result R. The total amount of energy consumption on the mobile system is:

$$P_c \times \frac{comp(C)}{U} \tag{1}$$

Figure 1 (b) shows the option of offloading without considering privacy. Data D and the program C are offloaded to the server and then the result R is returned to the mobile system. The total energy consumption includes idle energy when the server runs C and the energy for sending D and receiving R through network:

$$P_l \times \frac{comp(C)}{S} + P_n \times \frac{size(D+R)}{B} \tag{2}$$

Figure 1 (c) shows the option of privacy-preserving offloading. Data D are protected as D' by a method P. To process protected data, some modifications to the program may be required. Thus the program C' is possibly different from C. D' and C' are offloaded to the server. The result R' is returned to the mobile system

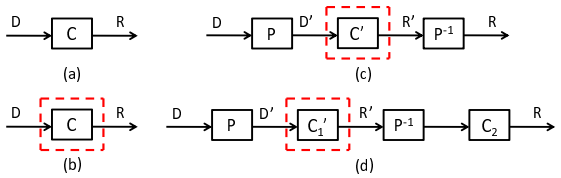

Figure 1: Four options to run a program. The dashed blocks are offloaded.

Table 1: Parameters for offloading

P_c	Computation power	U	Speed of mobile system
P_n	Network power	S	Speed of server
P_l	Idle power	B	Bandwidth of network

and then processed by the inverse protection P^{-1} obtaining the result R. Energy is consumed for performing P and P^{-1}, keeping idle when the server runs C', sending D' and receiving R':

$$P_c \times \frac{comp(P + P^{-1})}{U} + P_l \times \frac{comp(C')}{S} + P_n \times \frac{size(D' + R')}{B} \tag{3}$$

In a program, operations that require additions and multiplications can be efficiently performed on encrypted data and offloaded to a server. Other types of operations may be less efficient on encrypted data and, hence, better performed on unprotected data on the mobile system. Figure 1 (d) shows the option of selective offloading with privacy protection. Data D are protected as D' by P. The program includes two parts C_1 and C_2; only C_1 is offloaded. C_1 is modified as C_1' and offloaded to the server. The result R' is returned to the mobile system and then processed by P^{-1} and C_2 for obtaining the result R. The total energy is consumed for performing P, P^{-1} and C_2, keeping idle when the server runs C_1', sending D' and receiving R':

$$P_c \times \frac{comp(P + P^{-1} + C_2)}{U} + P_l \times \frac{comp(C_1')}{S} + P_n \times \frac{size(D' + R')}{B} \tag{4}$$

3.2 Data Protection in Image Retrieval

Image retrieval is computationally intensive and can benefit from offloading to save energy. To handle the protected data, the image retrieval program may be modified; the modified program must provide acceptable retrieval performance compared with the original program and unprotected data. Different retrieval algorithms may need different protection schemes. We consider two retrieval algorithms: ImgSeek and Gabor filtering, and two protection schemes: steganography and homomorphic encryption. Steganography uses a cover image to disguise the secret image so that it is difficult to detect. A simple and commonly used image steganographic technique is hiding images by replacing bits from the cover image with bits from the secret image. Encryption transforms plain data to make them unreadable. Steganography is different from encryption: the former hides the

existence of data. In contrast, encryption makes the data meaningless without the key. ImgSeek can be performed on steganographic data as shown in [13], based on the linear property of feature extraction. However, ImgSeek assumes the images are with the same scales and orientations. Gabor filtering has better accuracy than ImgSeek for searching similar images with rotated objects. When using Gabor filtering, steganography is not suitable to protect data any more, because the feature extraction in Gabor does not show linear properties. Homomorphic encryption can be used in Gabor retrieval, because Gabor filtering mostly performs additions and multiplications on encrypted data.

3.3 Offloading Gabor Retrieval with Encryption

We adopt homomorphic encryption to protect data when offloading the Gabor filtering retrieval.

3.3.1 *Encryption and Offloading*

Since the encryption and decryption are based on modular operations, we must ensure that both input and output values do not wrap around the modular representation and this can be achieved by a large key N. A small key N may generate the wrong decryption and substantially degrade retrieval performance. Besides, the key size is also important to security of encryption. A large key needs more computation to break; thus it provides better protection. However, a large key also means more energy consumption for transmitting and computing a larger amount of data.

After encryption, some modifications are required on the Gabor filtering retrieval program. The encryption preserves homomorphism only for integers and the coefficients of Gabor filters are not integers; we must represent the coefficients to suitable approximated integers. This is accomplished by quantizing a coefficient s to $\lceil Ks \rfloor$, where $\lceil . \rfloor$ is the rounding function and K is a suitable scaling factor. For example, 0.0127 can be approximated as $\lceil 1000 * 0.0127 \rfloor = 13$, if $K = 1000$. A larger K produces a more accurate approximation. Since all computations are among integers, they can be computed in the encrypted domain. After quantization, the approximated Gabor filters are used to convolve data. Since the encryption expands all pixels into large integers, we must define computations on large integers in the implementation. The modified retrieval program can be performed on both original data and encrypted data, with possibly different degrees of accuracy due to approximation.

The Gabor filtering retrieval program contains two steps $S1$ and $S2$ as described in Section 2. Without encryption, all computations can be offloaded to a server. With encryption, not all of the computations can be offloaded. In the second step $S2$, there are operations such as division and root square which are inefficient to be performed on the encrypted data and thus we decide to keep them on the mobile system running on unprotected data. Our analysis in Section 4.3 shows that $S1$ requires significantly more computation than $S2$ and offloading $S1$ can save substantially amounts of energy.

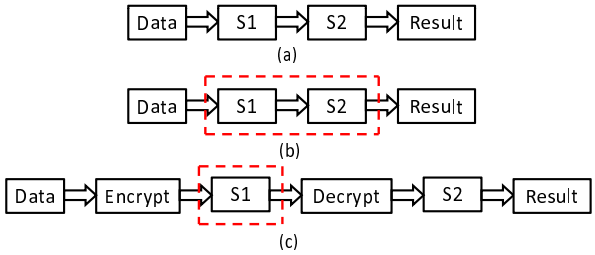

Figure 2: Three options to run Gabor filtering retrieval program. (a) Running on the mobile system without offloading; (b) Offloading without encryption; and (c) Offloading with encryption. Only the dashed blocks are offloaded to a server and the others are run on the mobile system.

3.3.2 *Energy Analysis*

There are three options to run the Gabor retrieval program. The first is directly running the entire program on the mobile system without encryption nor offloading as shown in Figure 2 (a). Its energy model is shown in Equation (1). The mobile system consumes energy for all computations of the retrieval program. The second option shown in Figure 2 (b) offloads the entire program to the server without data protection. Its energy model is shown in Equation (2). Energy for running the retrieval program is greatly reduced because the server is much faster than the mobile system. Meanwhile, offloading also causes extra energy for transmitting data. It may save energy but lose privacy. The third option shown in Figure 2 (c) considers data protection when offloading the program. It first encrypts data and offloads them together with $S1$ to the server, then decrypts the returned results and runs $S2$ to obtain the final retrieval results. Its energy model is shown in Equation (4). In this option, the protection leads to extra energy for encrypting and decrypting data. Also the protection enlarges data size transmitted through network and thus increases transmission energy. This option may protect privacy and save energy if the overhead can be compensated by the energy savings from running the heavy computation on the server

4. EXPERIMENTAL RESULTS

In this section, we evaluate privacy protection, retrieval accuracy, and energy consumption. The programs are implemented in $C\#$ and ported on an HP iPAQ PDA as the mobile system; a computer with 2GHz CPU and 3GB memory works as the server. We connect a 0.25Ω resistor to the PDA's battery and use National Instruments data acquisition card to read the voltages for measuring the energy consumption. In our experiments, every query image is transformed into 9 variations including: (1) blocks of 4 pixels average, (2) graylevels, (3) blocks of 4 pixels shuffled, (4) 45-degree clockwise rotation, (5) 90-degree clockwise rotation, (6) 180-degree rotation, (7) motion blur, (8) zoom blur, and (9) adding noise. For a query image, its 9 transformed images plus itself are regarded as its matched images and the retrieval programs will search for these matched images in the collection.

4.1 Privacy Protection

There are generally three kinds of privacy attacks [8]: (1)*ciphertext-only attacks*: the attacker has access only to some ciphertexts; (2)*known-plaintext attacks*: the attacker has access to some plaintext-ciphertext pairs; and (3)*chosen-plaintext or ciphertext attacks*: the attacker can choose the plaintext-ciphertext pairs. The difficulty of the attacks decreases in the above sequence. In our offloading solution, both encryption and decryption are performed on mobile systems; the server can only access the encrypted data. Thus the attacker can only launch ciphertext-only attacks and they are the weakest form of attacks. We may further weaken the attacks by changing the keys in the encryption of different pixels. The key size is also an important parameter for computational security in cryptography. In our approach, finding the secret p from N is as difficult as factoring N. The General Number Field Sieve algorithm (GNFS) [14] is currently the best known method for factoring large numbers. For a b-bit number N, the computational complexity is: $O(exp((\frac{64}{9}b)^{\frac{1}{3}}(logb)^{\frac{2}{3}}))$, which shows a larger key always requires more computational complexity to break and provides better security than a smaller key.

4.2 Retrieval Performance

We consider the following nine cases for comparison: applying (1) ImgSeek, (2) original Gabor filtering, and (3) modified Gabor filtering program on (a) original data, (b) steganographic data, and (c) encrypted data respectively. The retrieval performance is assessed in terms of the commonly used *recall* defined in [10]: $recall(L) = \frac{X}{Y}$, where L is the total number of returned images, X is the number of matched images in the returned L images, and Y is total number of matched images in the entire collection. In our experiments, $Y = 10$. We return 20 images, so $L = 20$. $0 \leq X \leq Y$, thus $0 \leq recall \leq 1$ with $recall = 1$ as the best performance. For every method, we run the program on the server with 5 queries to search in the collection of 10,000 images downloaded from the Internet, and use the average recalls to evaluate the retrieval performance.

The scaling factor K and the size of key N both affect the retrieval performance. The effect of K is shown in Figure 3 (a). When K is 10, the recall is nearly zero. A larger scaling factor K makes the approximation more precise and the retrieval performance improves. After K reaches 1000, the performance has negligible difference. Figure 3 (b) shows the effect of key size. When N is very small, the recall is low because most of decryptions are incorrect due to the errors from modular operations. The performance improves as the key size increases until N reaches 64 bits and then the performance becomes almost constant. We see $K = 1000$ and 64-bit N can provide reasonable retrieval performance so they are used in the following comparison of different methods.

Figure 4 shows the comparison of retrieval performance from three methods on three types of input data. The Gabor method shows better performance than ImgSeek when searching in original image collection, where some images are rotated. When data are

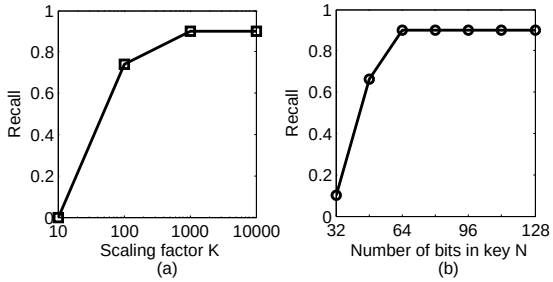

Figure 3: Retrieval performance with different scaling factors (a) and different key sizes (b).

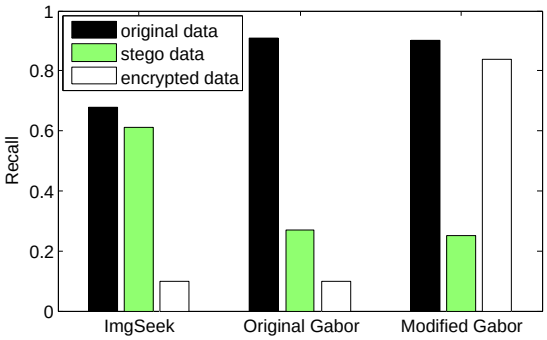

Figure 4: Retrieval performance comparison in different methods. A high bar indicates a better retrieval performance.

protected by steganography, ImgSeek can obtain about 83% performance compared with searching original data, but the recall is very low when data are encrypted. Original Gabor method has better performance on searching original data with recall about 0.91, but the performance on steganographic data and encrypted data are unsatisfactory; thus; modifications are required. Compared with the original one, the modified Gabor program provides performance close to searching unprotected data, and greatly improves the performance on encrypted data from about 0.10 to about 0.84 of the recall. Since the modified Gabor method works well on both original data and encrypted data, *the same program can be offloaded to a server regardless whether a user wants to protect privacy.* The results suggest that different image retrieval algorithms need different protection schemes. Steganography is better for ImgSeek and encryption is better for Gabor filtering.

4.3 Energy Consumption

We consider the following methods to compare energy consumption. (A) The Gabor filtering program is run directly on the PDA. This is the baseline for comparison; (B) The Gabor filtering program is offloaded to the server without encryption; (C)-(G) protect data using homomorphic encryption with different key sizes: (C) 64-bit, (D) 80-bit, (E) 96-bit, (F) 112-bit, and (G) 128-bit. Since the computation is very slow on the PDA, we reduce the image collection to 1000 images.

We first examine the energy consumption of running the two steps $S1$ and $S2$ of Gabor filtering retrieval by the method (A). According to our measurement, running

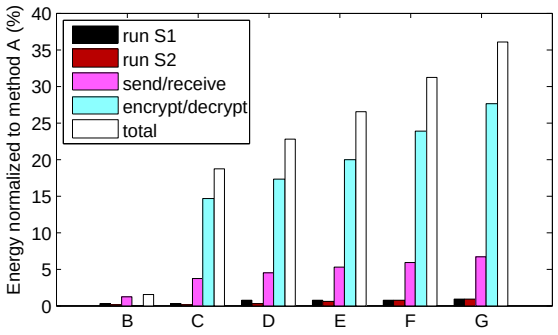

Figure 5: Energy consumption in different offloading methods. For each method, four components and the total energy are measured. All energy is normalized to the method (A). A higher bar indicates more energy consumption.

$S1$ consumes about 99.97% of the total energy consumption; $S2$ consumes negligible amount of energy with only about 0.03%. $S1$ dominates because it includes the convolutions between images and Gabor filters with much more computations than $S2$. Thus, $S1$ is should be offloaded to a server; $S2$ can be run either on the mobile system or on the server.

Next we compare the energy consumption in different offloading methods. Figure 5 shows the total energy and the following four energy components in every offloading method: (1) run $S1$; (2) run $S2$; (3) send/receive data; and (4) encrypt/decrypt data. Although running $S1$ consumes much more energy by the method (A), its energy consumption is greatly reduced when offloaded to the server. Running $S2$ consumes negligible amounts of energy no matter on the mobile system or on the server. Both $S1$ and $S2$ consume less than 1.0% amount of the total energy of method (A). In direct offloading (B), it consumes additional energy for sending and receiving data, only about 1.1% of the energy of (A). In the privacy-preserving offloading methods (C)-(G), data are protected before they are offloaded. The protection causes extra energy for encrypting and decrypting data; they account for the most part of total energy consumption on the mobile system. As the key size increases, more computations are required to encrypt and decrypt data. This part of energy increases from about 14.6% in (C) to 27.6% in (G). The protection also expands input data for transmission. Data size increases as the key size so the amount of energy for transmission through network is raised from 3.7% in (C) to 6.6% in (G).

We compare the total energy consumption in different offloading methods. Energy consumption of all these methods is less than 100% of (A); (B)-(G) all save energy from offloading. (B) consumes the least amount of energy at about 1.42% of (A), but it does not protect privacy. (C)-(G) consume more energy than (B) due to the overhead of data protection. The method based on a larger key size always results in more energy consumption. As the key size increases, the total amount of energy consumption increases from about 18.7% in (C) to about 35.9% in (G).

5. CONCLUSION

We present a method to offload image retrieval to save energy with privacy protection. The data are protected by homomorphic encryption and the computation is offloaded to a server. We implement our method on a PDA and compare retrieval performance and energy consumption. Results show that our method can obtain close performance compared with the direct retrieval without offloading and save energy to different degrees by choosing different key sizes in the encryption.

6. ACKNOWLEDGMENTS

This work is supported in part by NSF grants CNS-0716271. Any opinions, findings, and conclusions or recommendations in the paper are those of the authors and do no necessarily reflect the views of the sponsor.

7. REFERENCES

[1] Rong and Pedram. Extending the Lifetime of a Network of Battery-Powered Mobile Devices by Remote Processing: a Markovian Decision-Based Approach. In *DAC*, pages 906–911, 2003.

[2] Wolski et al. Using Bandwidth Data to Make Computation Offloading Decisions. In *IEEE International Symposium on Parallel and Distributed Processing*, pages 1–8, 2008.

[3] Hong et al. Energy Efficient Content-Based Image Retrieval for Mobile Systems. In *ISCAS*, pages 1673–1676, 2009.

[4] Li et al. Computation Offloading to Save Energy on Handheld Devices: a Partition Scheme. In *CASES*, pages 238–246, 2001.

[5] De Capitani di Vimercati et al. Preserving Confidentiality of Security Policies in Data Outsourcing. In *ACM Workshop on Privacy in the Electronic Society*, pages 75–84, 2008.

[6] Chandramouli et al. Image Steganography and Steganalysis: Concepts and Practice. In *Digital Watermarking*, pages 35–49, 2004.

[7] C. Gentry. Computing Arbitrary Functions of Encrypted Data. In *Communications of the ACM*, pages 97–105, 2010.

[8] Fontaine and Galand. A Survey of Homomorphic Encryption for Nonspecialists. In *EURASIP Journal on Information Security*, volume 2007, pages 1–15, 2007.

[9] Zhu et al. A Method of Homomorphic Encryption. In *International Conference on Wireless Communications, Networking and Mobile Computing*, pages 1–4, 2006.

[10] Datta et al. Image Retrieval: Ideas, Influences, and Trends of the New Age. In *ACM Computing Surveys*, pages 34–94, 2008.

[11] Jacobs et al. Fast Multiresolution Image Querying. In *International Conference on Computer Graphics and Interactive Techniques*, pages 277–286, 1995.

[12] Richard Ng et al. Performance Study of Gabor Filters and Rotation Invariant Gabor Filters. In *International Conference on Multimedia Modeling*, pages 158–162, 2005.

[13] Liu et al. Tradeoff between Energy Savings and Privacy Protection in Computation Offloading. In *ISLPED*, 2010.

[14] C. Monico. General Number Field Sieve Documentation. In *GGNFS Documentation*, 2004.