LESS: Big Data Sketching and Encryption on Low Power Platform

Amey Kulkarni[1], Colin Shea[1], Houman Homayoun[2], and Tinoosh Mohsenin[1]

[1]Department of Computer Science & Electrical Engineering, University of Maryland, Baltimore County
[2]Department of Electrical and Computer Engineering, George Mason University

***Abstract*—Ever-growing IoT demands big data processing and cognitive computing on mobile and battery operated devices. However, big data processing on low power embedded cores is challenging due to their limited communication bandwidth and on-chip storage. Additionally, IoT and cloud-based computing demand low overhead security kernel to avoid data breaches. In this paper, we propose a Light-weight Encryption using Scalable Sketching (LESS) framework for big data sketching and encryption using One-Time Random Linear Projections (OTRLP). OTRLP encoded matrix makes the Known Plaintext Attacks (KPA) ineffective, and attackers cannot gain significant information from plaintext-ciphertext pair. LESS framework can reduce data up to 67% with 3.81 dB signal-to-reconstruction error rate (SRER). This framework has two important kernels "sketching" and "sketch-reconstruction", the latter is computationally intensive and costly. We propose to accelerate the sketch reconstruction using Orthogonal Matching Pursuit (OMP) on a domain specific many-core hardware named Power Efficient Nano Cluster (PENC) designed by authors of this paper. To demonstrate efficiency of LESS framework, we integrate it with Hadoop MapReduce platform for objects and scenes identification application. The full hardware integration consists of tiny ARM cores which perform task scheduling and objects identification application, while PENC acts as an accelerator for sketch reconstruction. The full hardware integration results show that the LESS framework achieves 46% reduction in data transfers with very low execution overhead of 0.11% and negligible energy overhead of 0.001% when tested for 2.6 GB streaming input data. The heterogeneous LESS framework requires 2× less transfer time and achieves 2.25× higher throughput per watt compared to MapReduce platform.**

***Keywords*—** Sketching technique, OMP, Domain specific many-core, CPU/GPU, Energy efficient, Real time.

I. Introduction

Continued growth in IoT devices demand big data processing and machine learning applications on mobile and battery operated devices such as health activity trackers and unmanned aerial vehicles (UAVs). These applications need energy efficient processing and storage architecture which can adapt to continuously changing sensor data. In last few years, hardware acceleration for big data processing using FPGAs, tiny cores and domain specific many-cores has become common due to their low energy consumption and fast processing capabilities. However, for such platforms communication bandwidth can be a potential bottleneck for large data transfers [1], [2]. Similarly in real-time applications such as video surveillance, massive amount of data is generated and communicated to the cloud for further processing. In these applications data dimension reduction benefit on-board transmission power. Hence, managing data transfer rates on low power processing nodes is of utmost importance.

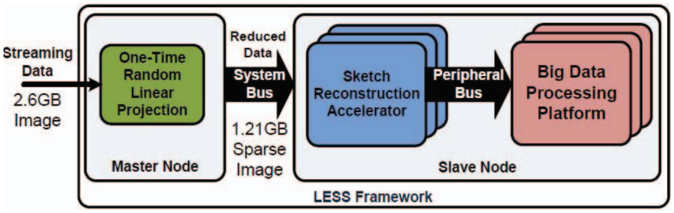

Fig. 1: Proposed LESS Secured Big Data Processing Framework, where the OTRLP kernel acts as an encryption in addition to reducing data, OMP Sketch Reconstruction kernel acts as an decryption and reconstruction of data

On the other hand, IoT and mobile devices gather data from variety of heterogeneous sources and must provide secured path for data collection and processing. Encryption is widely considered to be safe practice to mitigate data exposure risks. However, real-time battery operated devices need light-weight encryption to secure the system. In this paper, we propose a Light-weight Encryption using Scalable Sketching (LESS) framework for big data sketching and encryption using One-Time Random Linear Projections (OTRLP) on embedded cores. To the best of our knowledge, this is the first framework which accelerates and performs secured data transfers for big data applications on embedded cores. The main contribution to the research include:

- Scalable, secure data transfer reduction by adopting LESS framework for big data applications.
- Evaluation and full hardware demonstration of LESS framework on big dataset benchmark for object identification with integration of Hadoop MapReduce platform with respect to hardware overhead in terms of energy and execution time.
- Hardware measurement and evaluations of LESS framework for reduction in data transfer time, throughput per watt, and quality of data reduction in terms of PSNR and Structural Similarity Index (SSIM).

II. Overview of LESS Framework

Figure 1 shows the proposed LESS framework for secured big data processing, in which encryption is achieved by using sketching technique and decryption is performed by OMP sketch-reconstruction algorithm. In big data system scenario,

978-3-9815370-8-6/17/$31.00 ©2017 IEEE

master node receives streaming data $\mathbb{D}_{n \times n}$, where n is size of streaming data matrix[1]. Sketching algorithm performs learned random projections on streaming data to obtain $\mathbb{R}_{m \times n}$. The reduced data $\mathbb{R}_{m \times n}$ and keys $\mathbb{K}$ are transferred over the system bus to the slave node. The slave node consists of OMP sketch-reconstruction kernels to recover sketched signals using keys $\mathbb{K}$ and big data processing platform.

Security Definitions Considering streaming data signal (plaintext) $\mathbb{D}_{n \times n}$ is sparse and seed keys $\mathbb{K}$ which generate one-time measurement matrix $\phi_{m \times n}$ be a measurement matrix such that $m < n$, where m is the number of measurements to be taken and n is length of the original signal. Then sketching problem can be stated as: Reconstruct $\mathbb{D}_{n \times n}$ from the knowledge of

$$\mathbb{R}_{m \times n} = \phi_{m \times n} \mathbb{D}_{n \times n} \tag{1}$$

A private key has two functions $E_k : \mathbb{D}_{n \times n} \rightarrow \mathbb{R}_{m \times n}$ and $D_k : \mathbb{R}_{m \times n} \rightarrow \mathbb{D}_{n \times n}$. Thus, $D_k(E_k(\mathbb{D}_{n \times n})) = \mathbb{D}_{n \times n}$ is unfeasible without knowing key $\mathbb{K}$ to determine $E_k(\mathbb{D}_{n \times n}) = \mathbb{R}_{m \times n}$. In case of known n linearly independent messages an attacker can deduce content of messages, thus data transfers are not secure under KPA if the same matrix is used multiple times. In this paper we use one-time random linear projections (OTRLP) scenario [3] in which each measurement matrix is only used one time and all measurement matrices are statistically independent.

Targeted Big Data Benchmarks We demonstrate efficiency of LESS framework targeting two different big data benchmarks, MIT-CSAIL [4] and CalTech-256 [5] for objects and scenes identification.

III. LESS Framework for Data Reduction and Encryption

Algorithm 1 shows LESS framework for data reduction and encryption. The encryption is performed using secured Gaussian one-time random linear measurement matrix (Step 1-2) at the master node. The reduced and encrypted data obtained from master node is communicated on system bus. At the slave node, signal is reconstructed before big data processing. We implement OMP algorithm to reconstruct sketched signal with knowledge of $\mathbb{K}$ and $\mathbb{R}_{m \times n}$.

OMP algorithm: OMP is a greedy algorithm, it has three different phases, Identification, Augmentation and Residual Update. In Identification phase, index (i) of highest magnitude of $\phi_{m \times n} * R$ is chosen as potential vector to find closest approximation to $\mathbb{D}_{n \times n}$. At each iteration, index (i) is added to the list of estimated support vectors in Augmentation phase. The Residual update phase generates next residual for next iteration. In this phase, formed Q augmented matrix is used in Least Square regression model to find linear relationship between augmented matrix (Q) and measured vector $(\mathbb{R}_{m \times n})$. Finally, the amount of contribution that column $\mathbb{R}_{m \times n}$ provides is subtracted to obtain a residue. The OMP algorithm takes k iterations to determine correct set of columns.

Algorithm 1 LESS Framework for Big Data Reduction and Encryption

```
Input: Streaming Data $\mathbb{D}_{n \times n}$, number of measurements $m$, Sparsity - $k$
Intermediate Stage Output: Reduced and encrypted data $\mathbb{R}_{m \times n}$, Keys to be send over secure channel $\mathbb{K}$
Output: Reconstructed Data

Constructing Encoding Matrix
1: Construct Secured Gaussian-one time measurement matrix $\phi_{m \times n}$ using i.i.d.zero-mean Gaussian variables
2: Perform random projections using Secured Gaussian-OTM on streaming data $\mathbb{D}_{n \times n}$ to obtain $\mathbb{R}_{m \times n}$

Recovery of Signals Using OMP Algorithm
3:Initialization
    • $RU_0 = \mathbb{R}$, $\Lambda_0 = \emptyset$, $Q_0 = \emptyset$ and $t = 0$
4:Identification
    • Find Index $\lambda_t = max_{j=1 \ldots n}$ subject to $| < \phi_j RU_{t-1} > |$
5:Augmentation
    • Update $\Lambda_t = \Lambda_{t-1} \bigcup \lambda_t$
    • Update $Q_t = [Q_{t-1} \ \mathbb{K}_{\Lambda_t}]$
6:Residual Update
    • Solve the Least Squares Problem
      $\mathbb{D}_t = \min_{\mathbb{D}} ||\mathbb{R} - Q_{\mathbb{R}} \mathbb{D}||^2$
    • Calculate new approximation: $\alpha_t = Q_t \ \mathbb{D}_t$
    • Calculate new residual: $RU_t = \mathbb{R} - \alpha_t$
7: Increment t, and repeat from step 2 if $t < k$ After all the iterations, we can find correct sparse signals.
```

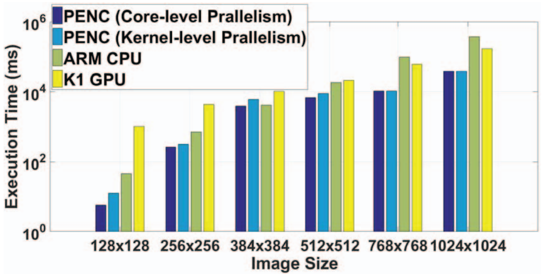

Fig. 2: Comparison of OMP execution time analysis on quad-core ARM CPU, K1 GPU at maximum clock rate of 2320.5MHz and 852MHz respectively, with PENC many-core (both core-level and kernel-level parallelism) implementations at 1GHz

OMP acceleration analysis on various platform: For all platforms, the measurement matrix ϕ is stored on-chip to reduce external memory overhead. Furthermore, Monte-Carlo simulations are performed since measurement matrix and sparse image is based on random variables. Figures 2 and 3 show the comparison of execution time and energy consumption between the ARM CPU, K1 GPU, and PENC many-core architecture [6], [7], [8]. Overall comparison between ARM CPU and K1 GPU shows that, ARM CPU performs best for smaller image sizes and K1 GPU performs better for large image sizes i.e. for higher computational complexity. Compared to ARM CPU and K1 GPU implementation PENC

[1] For convenience to explain overview of the framework, we selected row and column size to be same. In real-time streaming data can be of different column and row sizes

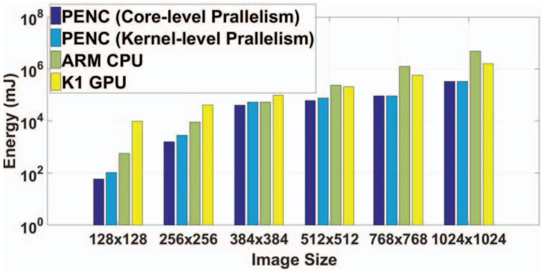

Fig. 3: Comparison of OMP energy consumption analysis on quad-core ARM CPU, K1 GPU at maximum clock rate of 2320.5MHz and 852MHz respectively, with PENC many-core (both core-level and kernel-level parallelism) implementations at 1GHz

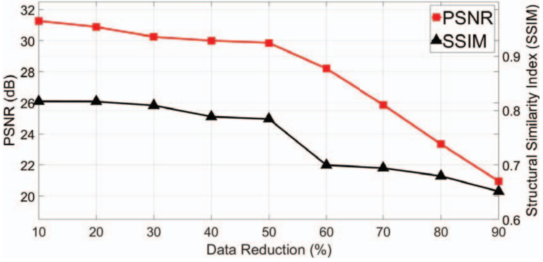

Fig. 4: Analysis of Objects and Scenes Identification on PENC+ARM CPU with respect to PSNR and SSIM for CalTech-256 Dataset

many-core platform performs 8× and 177× faster and saves 15× and 200× energy consumptions, respectively. Additionally considering chip area for TK1 platform in 28nm and PENC platform in 65nm, PENC many-core platform is the most efficient choice for OMP kernels.

Figure 1 shows heterogeneous LESS framework that consists of an accelerator for efficient sketch recovery using a fully flexible and parallel OMP reconstruction architecture and a host processor to perform post processing. An accelerator is a programmable platform adopted to perform compute-intensive sketch-reconstruction while achieving low power and high-speed computations. In this paper based on comparison results, we consider PENC many-core as an accelerator which provides programmability, parallelism and energy efficiency.

IV. Evaluation of MapReduce-LESS Integration

To demonstrate the efficiency of the proposed LESS framework, we integrate it with Hadoop MapReduce for objects and scenes identification application using MIT-CSAIL [4] and CalTech-256 [5] as shown in Figure 7A. A MapReduce object identification implementation was created using PENC many-core platform and the low power NVIDIA Jetson TK1 platform, where PENC is used as an accelerator and TK1 platform as the main processing engine as shown in Figure 7B. Figure 4 shows LESS data quality in terms of PSNR and structural similarity index measure (SSIM). In this experiment, we use 20-stage cascaded classifier trained with different numbers of images of size 512×512 consisting of up to 500 positive samples for each object. For detection stage, each mapper ingest its share of up to 30,000 images depending on the dataset used.

We adopt Hadoop 2.6.3 platform, and the native Hadoop libraries were built from source for the platform. The experiment is performed in four different stages: 1. The images to be analyzed are sketched using OTRLP. The resulting transformed image are stored as binary files for distribution. 2. *SequenceFile* is used to create a persistent data structure for binary key-value pairs. The key is generated from the name of the file and a value is the binary data from the compressed file. It ensures that the binary data of each image is not segmented before object identification occurs. 3. At the consumer (mapper) end, reconstruction of the sketched image is performed using PENC many-core platform. While the reconstructed image is placed into a queue for the consumer thread, the producer thread reads the next key-value pair. 4. Finally, the consumer thread passes reconstructed data to cascade classifiers for the object identification application.

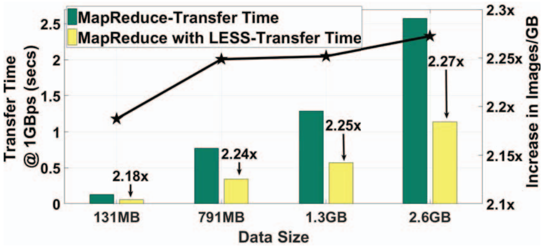

Fig. 5: Comparison of data transfer time of MapReduce and MapReduce with LESS framework at 1GBps for different data sizes with increase in images per GB statistics

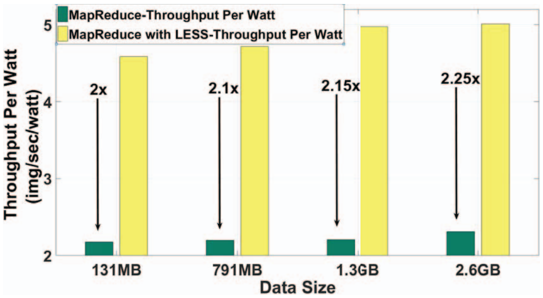

Fig. 6: Comparison between MapReduce and MapReduce with LESS framework for Throughput per Watt analysis

We adopt LESS framework to reduce data storage and transfer requirements, however data reduction brings two important challenges: 1. cost of computation, 2. decompression error rate. We measure computation cost for data reduction in terms of execution time and energy consumption overhead, whereas decompression error rate is measured in terms of PSNR and SSIM. Figure 7 C-E shows the example of original image transition to reconstructed image and objects identification on Hadoop MapReduce platform with LESS framework. Table I shows execution and energy consumption analysis of the LESS framework integrated with Hadoop MapReduce for objects identification application. The proposed LESS framework reduces data transfers by 46% reduction in data. To demonstrate efficiency of the PENC sketch reconstruction acceleration, we implemented MapReduce platform in two different cases, 1. ARM CPU is used for sketch reconstruction and processing i.e for master, sketch reconstruction and mapper, reducer. 2. Combination of PENC and ARM CPU, in which PENC

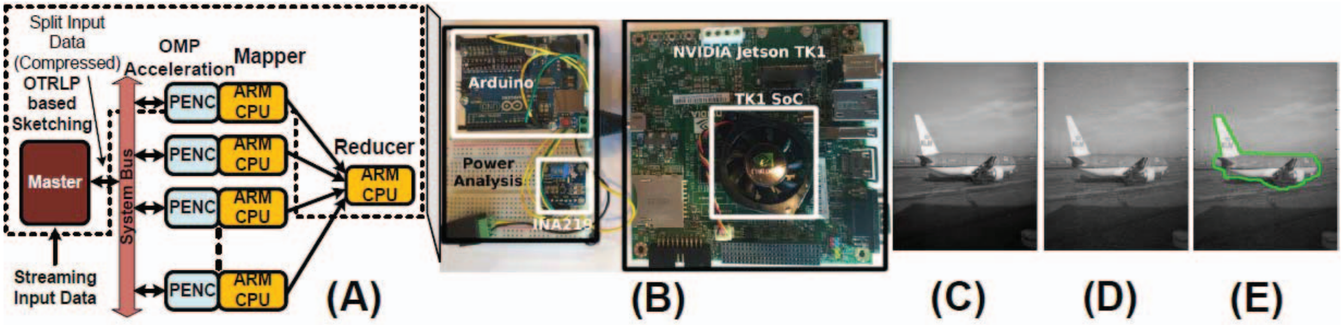

Fig. 7: (A) Integration of LESS Hardware Framework with Hadoop MapReduce, Sketch Reconstruction is achieved using OMP algorithm on PENC many-core platform, Map and Reduce is performed on ARM CPU. (B) MapReduce setup on Nvidia TK1 platform with current measurement setup using a TI INA219 and an Arduino Uno, (C,D,E) Visual representation of image before and after each stage of processing. (C) Original (D) Reconstructed image from the stored compressed image (E) Successful object identification of the reconstructed image

TABLE I: Execution Time and Energy Consumption Analysis of MapReduce integrated with LESS framework for Objects and Scenes Identification Application, with measure at 0.4. In ARM CPU only implementation the Sketch reconstruction and processing performed on ARM CPU whereas in PENC+ARM CPU, Sketch reconstruction is performed on PENC and processing on ARM CPU

	Data Transfer	Application Execution Time						Overhead	
Size of Data	Reduction	ARM Only		PENC + ARM		Improvement		Execution	Energy
	Through	Execution (S)	Energy (J)	Execution (S)	Energy (J)	Execution	Energy	(%)	(%)
	Sketch (%)	(secs)	(Joules)	(secs)	(Joules)	(%)	(%)		
1,000 Images (26MB)	48.62	752	9,487	585.36	7,380	22.2	22.1	2.36	0.003
5,000 Images (131MB)	47.94	3,618	45,613	2,737	34,169	24.3	24.9	1.32	0.003
10,000 Images (263MB)	47.68	7,163	90,315	5,907	74,488	17.5	17.5	0.22	0.002
30,000 Images (791MB)	46.63	23,443	295,578	18,373	231,667	21.6	21.6	0.18	0.001
50,000 Images (1.3GB)	46.56	36,178	456,139	28,455	341,693	21.3	25.1	0.11	0.001

is used for sketch reconstruction and ARM CPU is used for master, mapper and reducer. Compare to ARM CPU implementation, PENC + ARM implementation reduces application processing time by 17-21% and saves 17-25% energy consumption. Additionally we also perform hardware overhead analysis of sketch reconstruction on Hadoop MapReduce platform. LESS framework has very low execution time overhead of 0.11% and negligible energy consumption overhead of 0.001% when tested for 2.6GB data. Figure 5 shows comparison of data transfer time at 1GBps between LESS-MapReduce integration and MapReduce platform, it also shows increased image storage per GB. Compared to MapReduce platform, integration with LESS framework requires 2x less transfer time. LESS framework integration with MapReduce has 2.25x higher throughput per watt compared to MapReduce platform as shown in Figure 6.

V. Conclusion

In this paper we propose a low overhead LESS framework to reduce and encrypt big data processing on low power platforms. LESS framework consists two important kernels, “sketching” and ”sketch-reconstruction”. We implemented OMP algorithm on domain specific PENC many-core platform which acts as an hardware accelerator and ARM CPU platform for Big Data processing. To demonstrate efficiency of the proposed framework, we integrated LESS framework with Hadoop MapReduce platform for objects and scenes identification application. The master that schedules the tasks, and the mapper which executes object identification are implemented on ARM CPU platform whereas, sketch reconstruction is performed on PENC many-core accelerator. The results show that the LESS achieves upto 46% reduction in data transfers with very low execution overhead of 0.11% and negligible energy overhead of 0.001% when tested for 2.6 GB data. The heterogeneous LESS framework requires 2× less transfer time and achieves 2.25× higher throughput per watt compared to MapReduce platform.

VI. Acknowledgment

This research is based upon work partially supported by the National Science Foundation under Grant No.1527151.

References

[1] A. Kulkarni *et al.*, “CS-based secured big data processing on FPGA,” in *Field-Programmable Custom Computing Machines (FCCM), 2016 IEEE 24th Annual International Symposium on*, May 2016, pp. 201–201.

[2] B. Rouhani *et al.*, “Ssketch: An automated framework for streaming sketch-based analysis of big data on fpga,” in *Field-Programmable Custom Computing Machines (FCCM), 2015 IEEE 23rd Annual International Symposium on*, May 2015, pp. 187–194.

[3] T. Bianchi *et al.*, “Analysis of one-time random projections for privacy preserving compressed sensing,” *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 2, pp. 313–327, Feb 2016.

[4] A. Torralba *et al.*, “Context-based vision system for place and object recognition,” in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, Oct 2003, pp. 273–280 vol.1.

[5] G. Griffin, A. Holub, and P. Perona, “Caltech-256 object category dataset,” California Institute of Technology, Tech. Rep. 7694, 2007.

[6] A. Kulkarni *et al.*, “Low energy sketching engines on many-core platform for big data acceleration,” in *Proceedings of the 26th Edition on Great Lakes Symposium on VLSI*, ser. GLSVLSI ’16. ACM, 2016, pp. 57–62.

[7] A. Page *et al.*, “Low-power manycore accelerator for personalized biomedical applications,” in *Proceedings of the 26th Edition on Great Lakes Symposium on VLSI*, ser. GLSVLSI ’16. ACM, 2016, pp. 63–68.

[8] A. Kulkarni *et al.*, “Adaptive real-time trojan detection framework through machine learning,” in *2016 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, May 2016, pp. 120–123.