RGC Ref No. 17202318

RESEARCH GRANTS COUNCIL

Application for Allocation from
the General Research Fund for 2018/19
Application Form (GRF1)

- **Please read the Explanatory Notes GRF2 (Sep 17) carefully before completing this form.**
- **To safeguard the interests of the researcher and the university, the awardee university bears the primary responsibility for prevention, detection and investigation of research misconduct, including but not limited to misuse of funds, data falsification, plagiarism and double-dipping. The university is strongly encouraged to vet the grant applications using anti-plagiarism software before submitting them to the RGC.**

PART I SUMMARY OF THE RESEARCH PROPOSAL

[To be completed by the applicant(s)]

1. Particulars of the Project

(a) (i) Name and Academic Affiliation of Principal Investigator (PI):

Name	Post	Unit/ Department/ University
Dr Cui, Heming	Assistant Professor	Department of Computer Science/The University of Hong Kong

(ii) Is the PI a new appointee within 2 years of full-time paid appointment to his/her first substantive position as an academic staff in a university at the time of submission of the proposal?

Yes ☐ **No** ☑

(iii) Title of Project: New Systems and Algorithms for Preserving Big-data Privacy in Clouds

(iv) Nature of Application

New ☑ **Re-submission** ☐ **Continuation** ☐

(b) (i) Primary Field: Software **& Code** 2206
Secondary Field: Computing Hardware **& Code** 2203

RGC Ref No. 17202318

(ii) A maximum of five keywords to characterise the work of your proposal (a maximum of 30 characters for each keyword)

1) Big-data Privacy
2) Cloud Computing
3) Data Flow Tracking
4) Differential Privacy
5) Intel SGX

(iii) Project Duration: 36 Months*

*** for longer term projects, please explain in your research plan in Part II 2(b)(i) why the proposed research cannot be completed within the normal span of 36 months.**

(iv) Total Amount Requested: $ 1,313,400

(c) Abstract of Research comprehensible to a non-specialist (either a maximum of 400 words in one A4 page of PDF document in standard RGC format or a maximum of 400 words for direct input in the text box):

In this big-data era, many business vendors (e.g., Uber) store data on clouds. Meanwhile, business vendors and third-parties implement self-defined queries (e.g., MapReduce) to process data, causing two severe privacy problems. First, third-parties can easily leak sensitive fields in data records (e.g., credit cards in Uber orders) through query results. Second, careless or malicious cloud providers (e.g., Amazon) can observe the data being queried.

This proposal aims to preserve big-data privacy with a holistic methodology: people can still run unmodified big-data queries, and this proposal will automatically prevent sensitive data leakage at runtime by accomplishing three objectives.

First, to confine malicious third-parties, we will build Kakute, the first Data Flow Tracking (DFT) system for big-data. Kakute provides easy-to-use APIs for business vendors to tag sensitive data fields, it then automatically tracks unmodified queries and prevents sensitive data flowing to query results. A challenge in existing DFT systems is that propagating tags in data-intensive computations is too slow (e.g., a notable DFT system incurs 128X performance overhead compared to native, insecure queries). By leveraging subtle efficiency features of big-data queries, we will create two fast tag propagation techniques. Our Kakute preliminary prototype presented in [ACSAC '17] incurs merely 32.3% performance overhead.

Second, we will create a Fine-grained Differential Privacy (FDP) technique and its new algorithms. Kakute and differential privacy are synergistic on confining malicious third-parties, because differential privacy allows aggregation computations on sensitive data

by adding noise to hide individual information. Unfortunately, existing differential privacy techniques are coarse-grained (inaccurate): they often conservatively add excessive noise to all query results, because they cannot track which sensitive inputs flowed to which results. By leveraging Kakute, our FDP technique automatically adds noise to only the sensitive (tagged) parts of results, effectively hiding sensitive data and making most results accurate.

Third, to confine malicious cloud providers, we will leverage the Intel SGX (Software Guard Extensions) hardware to build the first privacy-preserving compiler for unmodified big-data queries. Existing SGX-based systems for big-data have two major challenges: they have to rewrite the queries from Java to SGX-compatible C++, or their trusted components running in SGX are too large (e.g., an entire JVM). To tackle these challenges, our compiler runs only the Java big-data queries in SGX using a thin, verified just-in-time translator created by us. The compiler also carries our new efficient SGX-based runtime techniques.

The success of this proposal will effectively preserve big-data privacy in clouds and benefit most people.

RGC Ref No. 17202318

(d) Special funding template (Applicants can select more than one box)

- ☐ **Clinical Research Fellowship Scheme (Please also complete an additional form (Enclosure I) and see Enclosure II) (only available for applications under Biology and Medicine Panel)**
- ☐ **Support for Individual Research (Time-off) (see Enclosure III) (only available for applications under Humanities and Social Sciences Panel and Business Studies Panel)**
- ☐ **Longer-term Research Grant (see Enclosure IV)**
- ☐ **Employment of Relief Teacher under Humanities and Social Sciences Panel (see Enclosure V) (only available for applications under Humanities and Social Sciences Panel)**
- ☐ **Provision of Research Experience for Undergraduate Student (see Enclosure VI)**
- ☐ **Support for Academic Research related to Public Policy Developments (see Enclosure VII)**

RGC Ref No. 17202318

PART II DETAILS OF THE RESEARCH PROPOSAL

[To be completed by the applicant(s)]

RESEARCH DETAILS

1. Impact and objectives

(a maximum of 800 words in total for the long-term impact and project objectives)

(a) Long-term impact:

The big-data and cloud computing trends enable great opportunities to all entities, including data providers (e.g., business vendors and computer users), cloud providers (e.g., Amazon), and computation providers who implement big-data queries (e.g., business vendors and their third-party partners). Unfortunately, despite decades of effort, data leakage remains one of the most severe threats in clouds. From the perspective of data providers (owners), severe data leakage incidents have been triggered by both computation providers (e.g., some iCloud third-parties leaked celebrity accounts on Internet in 2017) and cloud providers (e.g., the 2013 Yahoo Cloud compromise and the 2014 J.P. Morgan account leakage).

From the perspective of big-data queries, data leakage can happen from both within and outside. When a query runs, code running within can often be buggy or malicious. Cloud providers, which run outside the queries, have also incurred numerous data leakage incidents due to compromises on external and insider attacks. Therefore, this proposal takes a holistic methodology: preventing leakage for data providers from both within (Objective 1 and 2) and outside (Objective 3) the queries.

In the short term, to confine malicious third-parties in private clouds (i.e., a data provider is the cloud provider), we plan to accomplish both Objective 1 and Objective 2. Objective 1 proposes Kakute, the first Data Flow Tracking (DFT) system for big data queries. Kakute tackles a notorious performance challenge on porting DFT to data-intensive computations. To achieve a robust DFT architecture for distributed big-data frameworks (e.g., Spark), Kakute completely captures the frameworks' inter-computer data flows. We have implemented a Kakute prototype and integrated it with Spark. Kakute carries built-in checkers for four security and reliability problems: sensitive data leakage, data provenance, programming bugs, and performance bugs. Kakute incurs a moderate performance overhead of 32.3% compared to native, insecure queries. It also effectively detects 13 real-world security and performance bugs. These promising preliminary results have been presented in [ACSAC '17] and [TPDS '17].

Kakute and differential privacy are synergistic on confining malicious third-parties: Kakute enforces mandatory access control on sensitive data, but it may cause some query results containing sensitive data to be missing; differential privacy allows the aggregation computations on sensitive data while hiding individual privacy, but due to the lack of precise data flow tracking, it often suffers from excessive noise and inaccurate results. Therefore, the Objective 2 of this proposal takes the first significant step to integrate DFT and differential privacy, leading to a novel Fine-grained Differential Privacy (FDP)

technique. By leveraging Kakute, FDP automatically adds noise to only the sensitive parts of results, effectively hiding sensitive data and making most results accurate.

In the intermediate term, we plan to confine malicious public cloud providers by accomplishing Objective 3. The Intel SGX hardware enforces strong confidentiality for data and code even if the cloud is malicious. Moreover, SGX is a good fit for big-data queries because these queries do data-intensive computations in user space and rarely invoke system calls.

However, existing SGX-based systems for big-data have two major challenges: they have to manually rewrite the Java big-data queries into SGX-compatible C++, or their trusted computing base (TCB) is too large (e.g., an entire JVM). To tackle these two challenges, Objective 3 will build the first privacy-preserving Java compiler with minimum TCB. Our compiler will run only the Java big-data queries in SGX with a thin, verified just-in-time translator, and the rest of JVM is outside SGX without affecting the privacy of the data being queried.

In the long term, by integrating the outcomes of all the three objectives in this proposal and extensively applying them on real-world software, we can help data providers enforce comprehensive privacy against both computation providers and cloud providers. This can benefit most people. For instance, many Hong Kong finance companies demand strong privacy for their data deployed in clouds.

(b) Objectives

[Please list the objectives in point form]

1. [To build the first Data Flow Tracking (DFT) system for private clouds]
We will create Kakute, a fast DFT system that can track and prevent sensitive data leakage in self-defined big-data queries. We will make Kakute support diverse big-data queries on large, popular datasets, and we will make Kakute incur reasonable performance overhead compared to native, insecure queries.
2. [To create a Fine-grained Differential Privacy (FDP) technique for private clouds]
We will leverage Kakute to develop FDP and its new algorithms, which will only add noise to sensitive data fields or query results, preserving strong differential privacy for sensitive data and good accuracy for most results. We will extensively study FDP's accuracy improvements on both sensitive and insensitive data compared to existing differential privacy techniques.
3. [To construct the first compiler for big-data privacy in public clouds]
Our compiler will support unmodified big-data frameworks by creating a thin, verified translator that automatically translates Java bytecode to SGX-compatible code. We will quantitatively evaluate the performance overhead of our compiler and whether it can protect data privacy against real-world privileged attacks.

RGC Ref No. 17202318

2. Background of research, research plan and methodology:

(a maximum of seven A4 pages in total in standard RGC format for items (a) and (b)(i); a maximum of one A4 page for item (b)(ii))

(a) Background of research

(b) (i) Research plan and methodology

Attached 7 pages(s) as follows

1 Research Background

This proposal has three entities: data providers, cloud providers, and computation providers (who write big-data queries). In private clouds, data providers are cloud providers; in public clouds, they differ. This section shows relevant techniques (§1.1, §1.2, and §1.3), motivation (§1.4), and related work (§1.5 and §1.6).

1.1 Big-data computing frameworks

Big-data frameworks (e.g., Spark [75] and MapReduce [28]) are popular for computations on tremendous amounts of data records. These frameworks provide self-defined Java functions (e.g., `map`/`reduce`) to let computation providers write their algorithms, and the frameworks automatically apply these functions on the data stored across computers in parallel.

To avoid excessive computation, big-data frameworks adopt a *lazy transformation* approach [48, 75]. Spark often uses lazy transformations (e.g., `map`), and calls to these transformations only create a data structure called RDD with *lineage* (the sequence of transformations on data records). Actual transformations are only triggered when materialization operations (e.g., `collect`/`count`) are called. Collecting operations trigger transformations only along lineages, so unnecessary computations are avoided. **Objective 1** will leverage lazy transformation to create a fast data flow tracking technique: Reference Propagation (§2.1).

1.2 Software-based privacy techniques

Data Flow Tracking (DFT) is a powerful mandatory access control technique for preventing sensitive data leakage [22]. DFT attaches a tag to a variable (or object), and this tag will propagate during computations on the variable at runtime. DFT is used in various areas, including preventing sensitive data (e.g.contacts) leakage in smart phones (TaintDroid [22]), web services [50], and server programs [36]. No DFT system exists for big-data computing, so **Objective 1** (§2.1) will create the first DFT system for big-data.

Complimentary to DFT, statistical techniques, including K-anonymization methods [41, 62] and differential privacy [43, 45, 53], allow the aggregation of sensitive data while adding random noise on inputs or query results to preserve individual privacy. However, statistical techniques are either not secure (K-anonymization) or suffering from great losses of accuracy (differential privacy). Recent work [29] reports more than 30% loss of accuracy. For query results, low accuracy means bad utility. For instance, a K-Means program will return centroids far from the actual ones, because the accuracy loss rate is much larger than the training error rate (usually less than 10% in practice).

A key reason for this bad utility problem is that differential privacy cannot track how sensitive data fields flow to query results, so they have to take a coarse-grained approach, which conservatively adds noise to all data fields or all query results. **Objective 2** (§2.2) will propose a novel fine-grained differential privacy technique, which combines the strengths of DFT and differential privacy.

1.3 Hardware-based privacy techniques

Trusted Execution Environment (TEE) is a promising technique for protecting computation in a public cloud even if the cloud's operating systems and hypervisors compromise. For instance, Intel-SGX [30], a popular commercial TEE product, runs a program in a hardware-protected *enclave*, so code and data are protected from outside. Compared with the approach of computing on encrypted data (§1.5), TEE is 100X to 1000X faster. For instance, a SGX-based system Opaque [76] incurs a moderate performance overhead of 30% compared to native, insecure big-data queries.

However, to practically run Java big-data queries with SGX, two open challenges remain. First, existing SGX-based systems [76] require computation providers to manually rewrite the readily pervasive Java queries into SGX-compatible C++, a time-consuming and error-prone process. Second, existing SGX-based systems for big-data have too large Trusted Computing Base (TCB). Existing systems (e.g., SGX-BigMatrix [55]) run a whole language interpreter (e.g., JVM and Python runtime) in enclaves, causing a too large (and too dangerous) TCB: JVM code comes from many different parties/vendors and extremely hard to be verified. **Objective 3** (§2.3) tackles these two challenges by building a new just-in-time compiler.

1.4 Motivation of objectives

Data leakage (or breach), defined as the leakage of sensitive customer or organization data to unauthorized users [35], is a top security threat [7, 33] in clouds. From a data provider's perspective, both computation providers (e.g., the 2017 iCloud account leakage caused by third-parties [72]) and cloud providers (e.g., the 2013 Yahoo Cloud compromise [64]) have caused severe data leakage and huge financial loss. This proposal aims to preserve the data provider's privacy by going two directions. First, we will propose two novel complimentary techniques in **Objective 1** (KAKUTE) and **Objective 2** (fine-grained differential privacy) to protect privacy against the computation providers in private clouds. Second, we will propose **Objective 3** (a new privacy-preserving compiler) to protect privacy against the (public) cloud providers. By integrating the outcomes from all three objectives, data privacy will be effectively preserved.

1.5 Related work by others

Computing on encrypted data. Homomorphic encryption [26] is a technique for computing on encrypted data in untrusted environments. There are two kinds of homomorphic encryption methods: Fully homomorphic encryption (FHE) and Partially Homomorphic Encryption (PHE). An evaluation [25] on FHE shows a slowdown of orders of magnitudes, unacceptable in practice. Systems that adopts PHE (e.g., Monomi [66] and CryptDB [52]) report reasonable overhead, but PHE has limited expressiveness (e.g., supports only a small subset of arithmetic instructions) or requires extra trusted servers. Seabed [49] uses asymmetric encryption schemes to reduce the performance overhead of AHE, but its expressiveness is still quite limited. Therefore, in commercial clouds, most data is processed in plaintext.

SGX-based systems. Intel SGX is a promising technique to provide privacy-preserving analytic in public clouds. Compared with software-based solutions, hardware-based solutions incur much lower overhead. TrustedDB [9] is a hardware-based secure database. VC3 [54] proposes a secure distributed analytic platform with SGX protection on MapReduce [19]. Opaque [76] supports secure and oblivious SQL operators on SparkSQL [8]. However, all these systems have limited expressiveness (e.g., SQL operators), and VC3 even needs to rewrite the program with C++. A recent work [47] proposes an oblivious machine leaning framework on trusted processors. SGX-BigMatrix [55] proposes an oblivious and secure vectorization abstraction on Python, but its trusted components are too big (an entire Python runtime).

Big-data privacy systems. Airavat [53], PINQ [43], and GUPT [45] propose to apply differential privacy [21] in MapReduce in order to prevent leakage from malicious queries, but existing differential privacy techniques often produce inaccurate results. MrLazy [6] preserves the privacy of bog-data queries with static DFT. Compared to dynamic DFT (§1.2), static DFT is imprecise and may suffer from false positives.

Data and software integrity (orthogonal). This proposal focuses on data privacy; integrity is an orthogonal area and not the focus of this proposal. Prior work is effective on preserving the integrity of data [68], big-data frameworks (e.g., IntegrityMR [71]), and compilers (e.g., JITGuard [24]). These orthogonal integrity-preserving techniques can be directly used in our proposed systems.

1.6 Related work by the PI and co-I

The PI is an expert on secure and reliable distributed systems [13–17, 23, 32, 69, 70, 73, 74]. The PI's work is published in top conferences and journals on systems (OSDI, SOSP, SOCC, TPDS, and ACSAC) and programming languages (PLDI and ASPLOS). The PI has built tools [15, 73] to detect various new security vulnerabilities in widely used real-world software, including data loss [15], buffer overflows [73], and Linux kernel compromises (CVE-2017-7533 [2] and CVE-2017-12193 [3]). The co-I is an expert on high-performance computing [4, 11, 34, 44, 56, 57, 77], big-data computing [23, 38], security [40, 63], and Java compilers [37, 58, 67]. The co-I's work is published in top systems conferences (Cluster, SC, and ICPADS) and journals (JPDC, TPDS, and IEEE Transactions on Computers). As preliminary results for this proposal, the PI has presented KAKUTE [32] in ACSAC '17, and the PI and co-I have collaborated to present CONFLUENCE [23] in TPDS '17.

2 Research Plan and Methodology

2.1 Objective 1: preventing big-data computation leakage with KAKUTE

This section presents major challenges in existing DFT systems (§2.1.1) and KAKUTE (§2.1.2).

2.1.1 Challenges: existing DFT systems are too slow and incomplete for big-data

Although DFT is a powerful access control technique, existing DFT systems incur high performance overhead, especially for data-intensive computations. For instance, we ran a recent DFT system Phosphor [10] in Spark with a WordCount algorithm on a small dataset of merely 200MB, and we observed 128X longer computation time compared with the native Spark execution [32]. The second challenge is completeness: big-data frameworks usually contain *shuffle* operations, which redistribute data and results across computers. However, most existing DFT systems ignore data flows across computers. For the few [50] who support cross-host data flows, transferring all tags in shuffles consumes excessive network bandwidth.

2.1.2 KAKUTE: a fast, precise DFT system for big-data

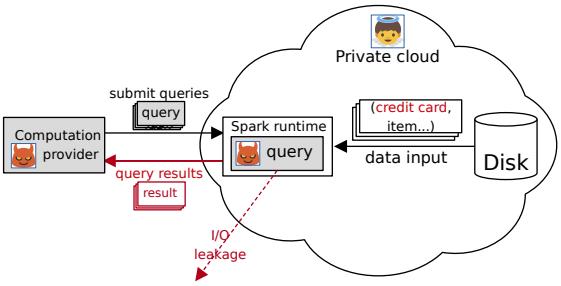


Figure 1: Threat model of KAKUTE. Red colors means sensitive data or leaking channels. Shaded (grey) components may leak data, and KAKUTE is designed to defend against them.

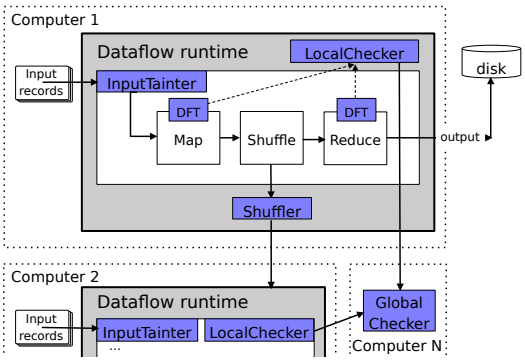


Figure 2: KAKUTE architecture. KAKUTE's key components are shaded (and in blue).

We present KAKUTE, the first precise and complete DFT system for big-data frameworks. Our key insight to address the DFT performance challenge is that multiple fields of a record often have the same tags with the same sensitivity level. For example, in a Taobao order record ⟨`time`, `userId`, `productID`⟩, only the `userId` field is sensitive, while the other fields are insensitive and they can share the same tag. Leveraging this insight, we present two new techniques, Reference Propagation and Tag Sharing. Reference Propagation avoids unnecessary tag combinations by only keeping the *lineage of tags* in the same self-defined queries, while Tag Sharing reduces memory usage by sharing tags among multiple fields in each record. To tackle the completeness challenge, KAKUTE completely captures inter-computer data flows (shuffles), and it efficiently reduces the amount of transferred DFT tags using Tag Sharing. Both techniques are illustrated in Appendix (c) of this proposal.

Figure 1 defines KAKUTE's threat model. Figure 2 shows KAKUTE's design. The InputTainter component provides easy-to-use APIs for data providers to automatically tag sensitive fields. The DFT component is enabled in self-defined functions. The Local- and Global-Checker detect and prevent illegal flows of sensitive fields (e.g., credit cards flow to IO functions in self-defined functions). Shuffle operations across computers are intercepted and tags are added. Therefore, DFT is completely captured across computers.

We plan to implement KAKUTE and integrate it with Spark. We will leverage Phosphor [10], an efficient DFT system working in the Java byte-code level. KAKUTE instruments computations of a Spark worker process to capture data flows inside self-defined-functions. KAKUTE provides different granularities of tracking with two types of tags: INTEGER and OBJECT tags. INTEGER provides 32 distinct tags for identifying 32 sensitivity levels, suitable for detecting data leakage and performance bugs. OBJECT provides an arbitrary number of tags, which is suitable for data provenance [31] and debugging big-data queries [27].

Preliminary results. We have implemented a KAKUTE [32] prototype and evaluated it on six popular big-data queries, including three text processing queries WordCount [61], WordGrep [39] and TwitterHot [61], two graph queries TentativeClosure [61] and ConnectComponent [61], and one medical query MedicalGroup [51]. We ran them with large, realistic datasets [12, 27, 31]. Our evaluation shows that: (1) KAKUTE incurred merely 32.3% overhead (Figure 3) with INTEGER tag, two orders of magnitudes faster than a recent DFT system Phorspor [10]; and (2) KAKUTE effectively detected 13 real-world security and performance bugs in other papers [18, 27, 53]. These promising preliminary results are presented in ACSAC 2017 [32] and TPDS 2017 [23].

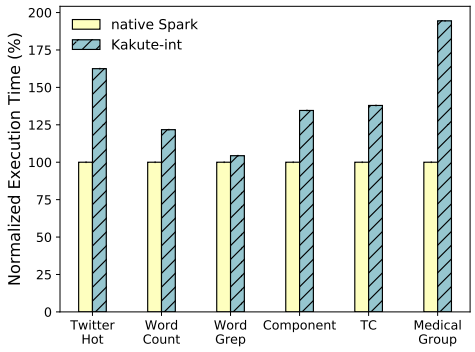


Figure 3: KAKUTE execution time normalized to native Spark executions. 100% means no performance overhead.

Future directions. We plan to extend KAKUTE in three directions. First, we will port KAKUTE onto more big-data frameworks, including PIG [48] and HADOOP [28]. Second, we will extend KAKUTE to detect broader types of real-world security bugs. Third, we will apply KAKUTE to augment other complementary privacy techniques, including differential privacy (i.e., **Objective 2**), K-anonymity [62], and L-diversity [41], which will likely lead to the inventions of diverse privacy-preserving techniques and systems.

2.2 Objective 2: developing the Fine-grained Differential Privacy (FDP) technique

KAKUTE (**Objective 1**) strictly prevents sensitive data flowing to IO functions or query results, but in some scenarios it is still desirable to let computation providers acquire aggregation results (e.g., the sum of citizens who have got cancer in a country) on sensitive fields as long as individual information is not leaked. Differential privacy [20] can enforce statistical bounds on aggregation results and prevent individual information leakage, so it is complementary to DFT and has attracted much attention recently.

2.2.1 Challenge: existing differential privacy techniques are coarse-grained and thus inaccurate

Existing differential privacy techniques often suffer from low accuracy for query results. To prevent computation providers revealing individual data, differential privacy mainly adds noise either on input data records or query results. However, due to the lack of precisely tracking how inputs are computed and propagated to outputs, to enforce statistical guarantee on outputs, differential privacy often conservatively adds roughly the same noise to all fields of a data record and to all records, causing inaccurate results. For instance, prior work [29] reports more than 30% loss of accuracy when the security guarantee is high (the probability of leakage is low). Therefore, a KMeans program will return centroids far from the accurate ones. This low accuracy makes results useless as it is much larger than the KMeans training error rate (a few percents).

2.2.2 FDP and its new aggregation algorithm

Our key insight is that DFT and differential privacy can complement each other, getting the best of both worlds. Considering each data record, DFT can precisely track how sensitive data fields flow to which query result, so differential privacy needs only add noise to the sensitive input fields or results. Considering all data records, DFT can also distinguish which records have higher security sensitivity levels, then we can add more noise accordingly.

This insight nurtures Fine-grained Differential Privacy (FDP). In FDP, each record belongs to a user who assigns a security tag to all its data records. A security tag that is related to the privacy budget ε (or accuracy level). When the privacy budget is high, the probability of leakages is high. $\Phi(x)$ returns the ε of a particular record x. The threat model of FDP is the same as KAKUTE's (Figure 1), because FDP aims to defend against malicious computation providers in private clouds. Figure 4 shows the workflow of FDP with five steps.

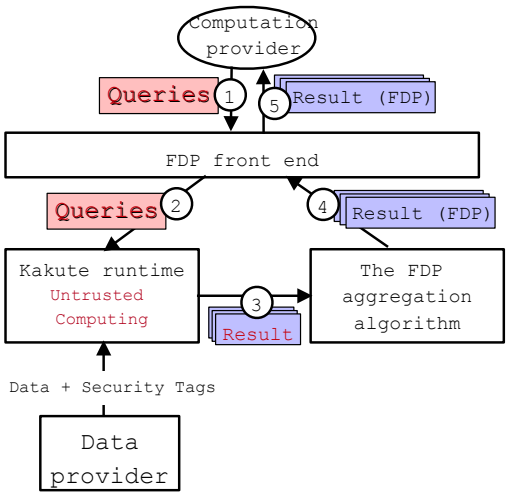


Figure 4: The workflow of FDP with five steps.

Algorithm 1: The FDP aggregation algorithm

Input: D: Dataset, N: dataset size, ε_k: privacy budget for level k, (min, max): output range

n = a suitable number of partition

for $p \leftarrow 1$ **to** n **do**

  $O_p \leftarrow Query(D_i)$;

  if $O_p >$ max, $O_p \leftarrow$ max

  if $O_p <$ min, $O_p \leftarrow$ min

$O \leftarrow \frac{1}{n}\sum_{p=1}^{n} O_p$

for *output record* O_i *of output* O **do**

  for *dimension* j *of a output* O_i **do**

    k $\leftarrow getTagLevel(O_{ij})$;

    $O_{ij} \leftarrow O_{ij} + Lap(\frac{max_j - min_j}{n\varepsilon_k})$

Output: O

Definition 2.1. *Differential Privacy For two neighboring datasets D and D′ differing at record x, a mechanism* $\mathscr{M}(y)$ *is differentially private with the following condition:*

$$Pr[\mathscr{M}(D) \in S] \leq e^{\Phi(x)} \times Pr[\mathscr{M}(D') \in S], S \subseteq Range(\mathscr{M}) \tag{1}$$

Intuitively, Differential Privacy guarantees that the probability of producing different result with neighboring datasets is low. To realize a differential privacy technique, FDP first needs to determine privacy budgets and security levels (`getTagLevel` in **Algorithm 1**). We use a common differential privacy model with 6 security levels (from insensitive to absolutely sensitive): insecure, dp_1, dp_2, dp_3, dp_4 and non-released.

Theorem 2.1. *Laplace Mechanism [20] Adding noise with Laplace distribution* $Laplace(\frac{\Delta f}{\varepsilon})$ *enforces Differential Privacy, and global sensitivity* Δf *is defined as*

$$\Delta f = \max_{D,D'} ||f(D) - f(D')||_1 \tag{2}$$

To enforce differential privacy, one approach is to use the ε inferred by the highest security level of each dimension and to add noise to the output directly, but this approach is too naive for the diversity of security levels. Instead, we adopt the sample-and-aggregate approach in previous work [60]. In this approach, data is partitioned into multiple parts to reduce Δf. Each field of a record x has a security level of ε_k. **Algorithm 1** adds noise to aggregated result of the query from all partitions according to security level of x's fields.

Theorem 2.2. *For any output record O, the mechanism of* ***Algorithm 1*** *is* $max_{i\leq k}(\varepsilon_i)$*-differentially private.*

Proof. For each dimension, we can divide the dataset D as k disjoin partitions D_1, D_2, ..., D_k according to their security levels. According to prior work[59], each $\mathscr{M}(D_i)(i \leq k)$ is ε_i-differentially private. For each D_i, we have $Pr[\mathscr{M}(D_i) \in S] \leq e^{\varepsilon_i} Pr[\mathscr{M}(D_i') \in S]$, suppose $\varepsilon_{max} = max_{i\leq k}(\varepsilon_i)$ and the differing record x of D and D' is in partition D_x,

$$\begin{aligned} Pr[\mathscr{M}(D)] &= Pr[\mathscr{M}(D_1 + D_2 + \ldots + D_k)] \\ &= Pr[\mathscr{M}(D_1)] + Pr[\mathscr{M}(D_2)] + \ldots + Pr[\mathscr{M}(D_k)] \\ &\leq Pr[\mathscr{M}(D_1)] + \ldots + e^{\varepsilon_x} Pr[\mathscr{M}(D_x')] + \cdots + Pr[\mathscr{M}(D_k)] \\ &\leq e^{\varepsilon_{max}}(Pr[\mathscr{M}(D_1)] + \ldots + Pr[\mathscr{M}(D_x')] + \ldots + Pr[\mathscr{M}(D_k)]) \\ &= e^{\varepsilon_{max}} Pr[\mathscr{M}(D')] \end{aligned} \tag{3}$$

Therefore, $\mathscr{M}(D)$ (i.e., **Algorithm 1**) is $max_{i\leq k}(\varepsilon_i)$-differentially private. □

Future directions. We will enrich our FDP technique by going along two directions. First, we will further optimize the algorithm to reduce its added noise. In the last line of **Algorithm 1**, the added noise on O_j comes from two parts: the Laplace noise added in each partition and the sum of noise added to all partitions. When the number of partitions increases, the Laplace per partition decreases (each partition is smaller), but the sum increases. This brings an interesting optimization problem on minimizing added noise, and we plan to create other algorithms (e.g., algorithms with hill-climbing or simulated-annealing styles) to compute the optimal number of partitions for minimizing added noise. Second, we will conduct an extensive study on diverse real-world big-data queries and realistic datasets in order to quantify the accuracy improvements of FDP and its new algorithms compared to existing differential privacy techniques (e.g., GUPT [45]).

2.3 Objective 3: creating a privacy-preserving compiler for big-data queries in public clouds

Recent real-world privacy breaches have shown that sensitive data are often leaked while being processed in public clouds, including clouds compromises on external attacks [72] and insider attacks [7]. Trusted Execution Environment (TEE) is a promising technique to protect computation on public clouds even if the cloud's operating system is compromised. For example, Intel-SGX [30] runs programs in an enclave, so code and data are protected and cannot be seen by the attackers. Meanwhile, SGX is good fit for big-data queries because these queries are data-intensive in userspace and they hardly invoke system calls (OS kernel can easily break SGX's security on memory). A latest big-data analytic system Opaque [76] reports only 30% overhead compared to native, insecure executions.

2.3.1 Challenges: existing SGX-based systems require rewriting queries and have too-large TCB

Despite recent SGX-based systems (Opaque [76], VC3 [54], Azure/Coco [1], and SGX-BigMatrix [55]) for big-data are promising, two major challenges remain. First, these systems require completely rewriting the readily pervasive and familiar Java big-data queries into C++ [1, 54, 76], a time-consuming and error-prone process. Second, to easy implementation, existing systems often run an entire language runtime (e.g., JVM or Python runtime [55]) within SGX, causing the Trusted Computing Base (TCB) to be too large (JVM has millions of lines of code developed by many companies, so it is vulnerable to bugs or insider attacks [7]).

2.3.2 MAAT: a just-in-time (JIT), privacy-preserving Java compiler for big-data queries

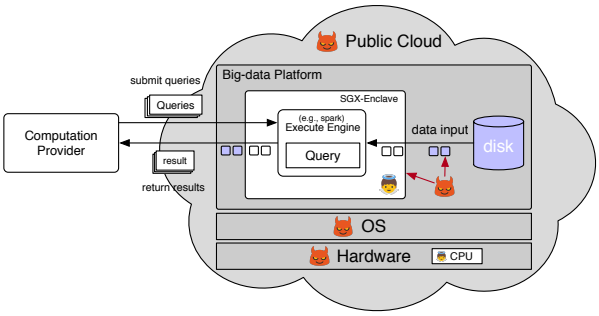


Figure 5: Threat model of MAAT. Data records with blue color are encrypted, and white color are plaintext. Shaded (grey) components may leak data, and MAAT is designed to defend against them.

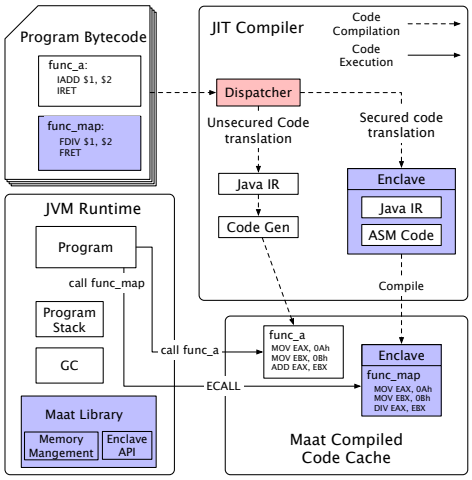


Figure 6: MAAT compiler architecture. Key components are shaded (and blue).

We propose MAAT, the first compiler that runs unmodified Java big-data queries in SGX enclaves securely with minimal TCB (i.e., the TCB contains only SGX and self-defined code itself). MAAT works as a Java JIT compiler which automatically compiles self-defined big-data functions (e.g., `map`/`reduce`) into SGX-compatible assembly instructions. Therefore, the JVM itself does not run in MAAT's enclaves.

MAAT's goal is to preserve the privacy (confidentiality) of data in public clouds. Other attacks such as changing the execution paths (i.e., integrity) of big-data frameworks or language runtimes have been well defended (e.g., IntegrityMR [71] and JITGuard [24]), and MAAT can directly use these techniques.

Figure 5 shows MAAT's threat model: both SGX and computation providers are trusted, and cloud providers are malicious. By integrating the outcomes from **Objective 1 or 2** into MAAT, computation providers no longer need to be trusted. Figure 6 shows the architecture of MAAT: both the translation of self-defined code and the execution of code are protected by enclaves. Even if a cloud's OS compromises, the OS can neither observe the data being queried nor inject malicious code into the queries during MAAT's translation. MAAT's code cache is for reusing translated code at runtime.

Two MAAT software components run in SGX: our JIT translator and our own management library. The JIT translator is a thin layer which translates a Java bytecode instruction into a number of SGX-compatible assembly instructions. For instance, in Figure 6, an FDIV Java bytecode instruction translates to two MOV and one DIV assembly instructions). The memory management library is for our own use of encryption/decryption on data records and maintaining SGX memory for the queries at runtime. We will proactively implement these two components to be easy to verify (use as few as function recursive calls and loops) as in other verification practice [46], and we plan to use state-of-the-art verification techniques [46] to assure that they both are functionally correct (i.e., free of bugs [46] and malicious code [5]). Then, we do not need to include them in MAAT's TCB, greatly reducing its TCB.

A subtle performance challenge for MAAT is that it should have reasonable performance overhead compared to native executions. When a program calls into (i.e., ECALL) and gets out of (i.e., OCALL) a function in enclaves, enclave transitions are invoked. Such transitions are several hundred times slower than user function calls. Moreover, encryption and decryption on data records are invoked during such transitions. Our study on a SGX-based system Opaque [76] shows that it incurs 3.4k enclave transitions for processing only 10k data (with two queries `select` and `groupBy`), which confirms the challenge.

To mitigate this challenge, we will create a new enclave runtime abstraction called Data-locality-aware Asynchronous Enclave calls (DAE). DAE converts the synchronous enclave calls (similar to Java function calls) to asynchronous, data-locality-aware calls into enclaves. Specifically, DAE will run a number of n processes (E_1 to E_n) in an enclave on each computer. When a JVM process P calls a big-data query function, the call and its parameters are appended to a queue to DAE, and DAE arranges a process E_i with good data locality (e.g., according to prior arrangements and the decrypted data held by E_i) to execute the call. The call result is appended to a return queue of the DAE for process P. We expect that DAE will achieve reasonable performance, data locality, and parallelism.

Future directions. By realizing a privacy-preserving Java JIT compiler for public clouds, MAAT has broad applications in other security areas, and we will further extend it along three directions. First, we will fully implement it and evaluate its efficacy on defending against diverse real-world privacy attacks launched by cloud providers. Second, we will further enhance DAE to support well isolated, secured operating system calls (e.g., library operating system calls [65]), so that MAAT will not only benefit big-data queries, but other distributed computing paradigms (e.g., graph queries [42]). Third, we will augment the translator to automatically transform the big-data queries with dangerous access patterns on particular data into those without (i.e., oblivious executions [55]). This will nurture new theory and algorithms on oblivious executions.

2.4 Research plan

This project will need two PhD students S1 and S2 to work for three years. In the first year, S1 will fully implement the KAKUTE system (part of **Objective 1**), and S2 will evaluate its performance and security strengths (part of **Objective 1**). In the second year, S1 will use KAKUTE to develop the proposed FDP technique (part of **Objective 2**), and S2 will develop the proposed aggregation algorithm for FDP (part of **Objective 2**). In the third year, S1 will build MAAT (part of **Objective 3**), and S2 will extensively study the privacy guarantee and performance of MAAT on real-world big-data frameworks (part of **Objective 3**).

(b) (ii) A one-page Gantt Chart showing the research activities

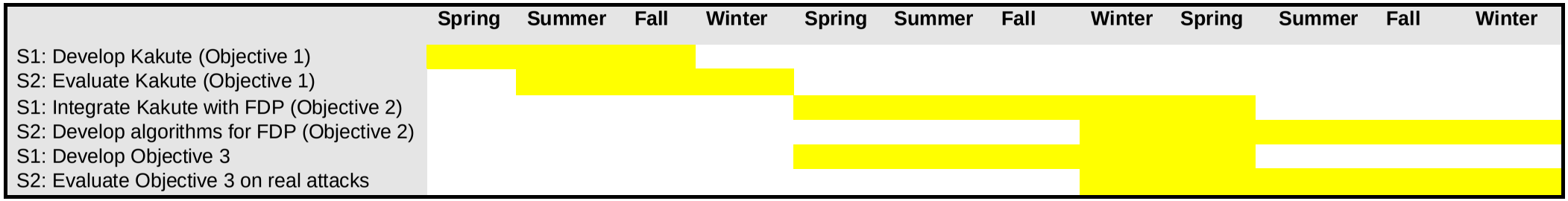
Attached 1 pages(s) as follows

HK RGC GRF Proposal (Ref No. 17202318)

New Systems and Algorithms for Preserving Big-data Privacy in Clouds

Project Timeline in Twelve Seasons (Three Years)

S1: PhD Student 1
S2: PhD Student 2
Proposed project start date: Jan 1, 2019
Proposed project end date: Dec 31, 2021



(c) A maximum of two non-text pages of attached diagrams, photos, charts and table etc, if any.

Attached 1 pages(s) as follows

Appendix (C): Non-text figures

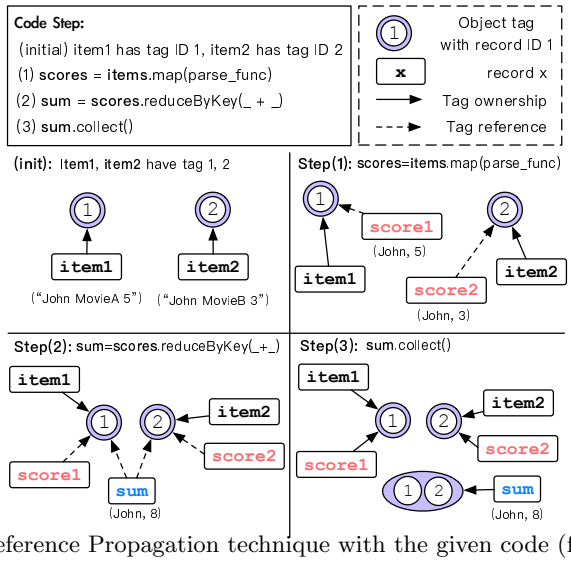


Figure 1: The Reference Propagation technique with the given code (for **Objective 1**).

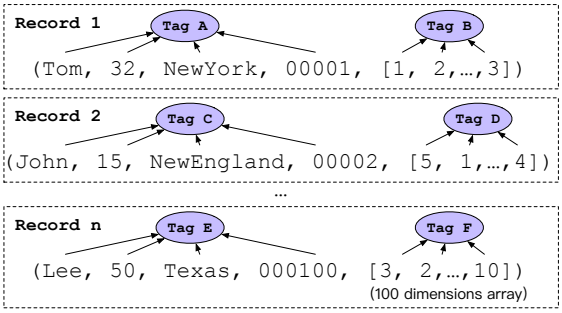


Figure 2: The Tag Sharing technique between fields in each record (for **Objective 1**).

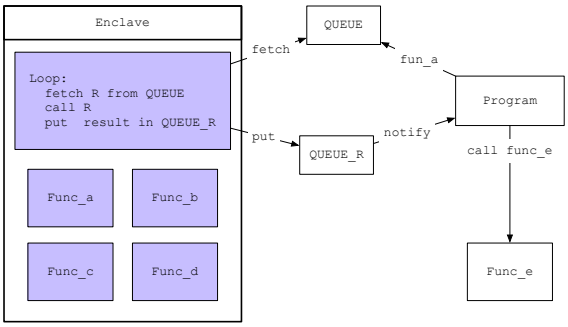


Figure 3: The Data-locality-aware Asynchronous Enclave (DAE) call abstraction (for **Objective 3**).

(d) Reference (a maximum of three pages for references is allowed for listing the publications cited in Section 1-2. All full references should be provided, including all authors of each reference.)

References

[1] GitHub - Azure/coco-framework. https://github.com/Azure/coco-framework.

[2] CVE-2017-7533. http://seclists.org/oss-sec/2017/q3/240.

[3] CVE-2017-12193. https://access.redhat.com/security/cve/cve-2017-12193.

[4] K. A, V. P. M. Shaaban, and W. C.L. Heterogeneous computing: Challenges and opportunities. In *IEEE Computer*, 1993.

[5] F. Adelstein, M. Stillerman, and D. Kozen. Malicious code detection for open firmware. In *Computer Security Applications Conference, 2002. Proceedings. 18th Annual*, pages 403–412. IEEE, 2002.

[6] S. Akoush, L. Carata, R. Sohan, and A. Hopper. Mrlazy: Lazy runtime label propagation for mapreduce. In *Proceedings of the 6th USENIX Conference on Hot Topics in Cloud Computing*, HotCloud'14, pages 17–17, Berkeley, CA, USA, 2014. USENIX Association. URL http://dl.acm.org/citation.cfm?id=2696535.2696552.

[7] C. S. Alliance. Top threats to cloud computing v1.0. https://cloudsecurityalliance.org/topthreats/csathreats.v1.0.pdf, Feb. 2010.

[8] M. Armbrust, R. S. Xin, C. Lian, Y. Huai, D. Liu, J. K. Bradley, X. Meng, T. Kaftan, M. J. Franklin, A. Ghodsi, et al. Spark sql: Relational data processing in spark. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1383–1394. ACM, 2015.

[9] S. Bajaj and R. Sion. Trusteddb: A trusted hardware based database with privacy and data confidentiality. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*, SIGMOD '11, pages 205–216, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0661-4. doi: 10.1145/1989323.1989346. URL http://doi.acm.org/10.1145/1989323.1989346.

[10] J. Bell and G. Kaiser. Phosphor: Illuminating dynamic data flow in commodity jvms. In *Proceedings of the 2014 ACM International Conference on Object Oriented Programming Systems Languages & Applications*, OOPSLA '14, pages 83–101, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2585-1. doi: 10.1145/2660193.2660212. URL http://doi.acm.org/10.1145/2660193.2660212.

[11] C. B.W.L., W. C.L., and K. Hwang. A migrating-home protocol for implementing scope consistency model on a cluster of workstations. In *PDPTA*, 1999.

[12] Z. Chothia, J. Liagouris, F. McSherry, and T. Roscoe. Explaining outputs in modern data analytics. *Proceedings of the VLDB Endowment*, 9(12):1137–1148, 2016.

[13] H. Cui, J. Wu, C.-C. Tsai, and J. Yang. Stable deterministic multithreading through schedule memoization. In *Proceedings of the Ninth Symposium on Operating Systems Design and Implementation (OSDI '10)*, Oct. 2010.

[14] H. Cui, J. Wu, J. Gallagher, H. Guo, and J. Yang. Efficient deterministic multithreading through schedule relaxation. In *Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP '11)*, pages 337–351, Oct. 2011.

[15] H. Cui, G. Hu, J. Wu, and J. Yang. Verifying systems rules using rule-directed symbolic execution. In *Eighteenth International Conference on Architecture Support for Programming Languages and Operating Systems (ASPLOS '13)*, 2013.

[16] H. Cui, J. Simsa, Y.-H. Lin, H. Li, B. Blum, X. Xu, J. Yang, G. A. Gibson, and R. E. Bryant. Parrot: a practical runtime for deterministic, stable, and reliable threads. In *Proceedings of the 24th ACM Symposium on Operating Systems Principles (SOSP '13)*, Nov. 2013.

[17] H. Cui, R. Gu, C. Liu, and J. Yang. Paxos made transparent. In *Proceedings of the 25th ACM Symposium on Operating Systems Principles (SOSP '15)*, Oct. 2015.

[18] A. Dave and M. Zaharia. Arthur: Rich post-facto debugging for production analytics applications.

[19] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. In *OSDI'04: Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation*, pages 10–10, 2004.

[20] C. Dwork. Differential privacy. *Lecture Notes in Computer Science*, 26(2):1–12, 2006.

[21] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Third Conference on Theory of Cryptography*, TCC'06, pages 265–284, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN 3-540-32731-2, 978-3-540-32731-8. doi: 10.1007/11681878_14. URL http://dx.doi.org/10.1007/11681878_14.

[22] W. Enck, P. Gilbert, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth. TaintDroid: an information-flow tracking system for realtime privacy monitoring on smartphones. In *Proceedings of the Ninth Symposium on Operating Systems Design and Implementation (OSDI '10)*, pages 1–6, 2010.

[23] L. Feng, L. F.C.M., C. Heming, and W. Cho-Li. Confluence: Speeding up iterative distributed operations by key-dependency-aware partitioning. In *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 2017.

[24] T. Frassetto, D. Gens, C. Liebchen, and A.-R. Sadeghi. Jitguard: Hardening just-in-time compilers with sgx. 2017.

[25] C. Gentry, S. Halevi, and N. P. Smart. Homomorphic evaluation of the aes circuit. In *Advances in Cryptology–CRYPTO 2012*, pages 850–867. Springer, 2012.

[26] C. Gentry et al. Fully homomorphic encryption using ideal lattices. In *STOC*, volume 9, pages 169–178, 2009.

[27] M. A. Gulzar, M. Interlandi, S. Yoo, S. D. Tetali, T. Condie, T. Millstein, and M. Kim. Bigdebug: Debugging primitives for interactive big data processing in spark. In *Proceedings of the 38th International Conference on Software Engineering*, ICSE '16, pages 784–795, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-3900-1. doi: 10.1145/2884781.2884813. URL http://doi.acm.org/10.1145/2884781.2884813.

[28] Hadoop. Hadoop. http://hadoop.apache.org/core/.

[29] X. Hu, M. Yuan, J. Yao, Y. Deng, L. Chen, Q. Yang, H. Guan, and J. Zeng. Differential privacy in telco big data platform. *Proceedings of the VLDB Endowment*, 8(12):1692–1703, 2015.

[30] Intel. Software guard extensions programming reference. https://software.intel.com/sites/default/files/329298-001.pdf.

[31] M. Interlandi, K. Shah, S. D. Tetali, M. A. Gulzar, S. Yoo, M. Kim, T. Millstein, and T. Condie. Titian: Data provenance support in spark. *Proc. VLDB Endow.*, 9(3):216–227, Nov. 2015. ISSN 2150-8097. doi: 10.14778/2850583.2850595. URL http://dx.doi.org/10.14778/2850583.2850595.

[32] J. Jianyu, Z. Shixiong, A. Danish, W. Yuexuan, C. Heming, L. Feng, and G. Zhaoquan. Kakute: A precise, unified information flow analysis system for big-data security. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC '17)*, 2017.

[33] S. D. I. John and A. O. Osonde. Privacy preservation in the age of big data. In *RAND '16*, 2016.

[34] H. K., J. H., C. E., W. C.L., and X. Z. Designing ssi clusters with hierarchical checkpointing and single i/o space. In *IEEE Concurrency*, 1999.

[35] M. Kazim and S. Y. Zhu. A survey on top security threats in cloud computing. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 2015.

[36] V. P. Kemerlis, G. Portokalidis, K. Jee, and A. D. Keromytis. Libdft: Practical dynamic data flow tracking for commodity systems. In *Proceedings of the 8th ACM SIGPLAN/SIGOPS Conference on Virtual Execution Environments*, VEE '12, pages 121–132, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1176-2. doi: 10.1145/2151024.2151042. URL http://doi.acm.org/10.1145/2151024.2151042.

[37] L. King-Tin, S. Jinghao, H. Dominic, W. Cho-Li, L. Zhiquan, Z. Wangbin, and Y. Youliang. Rhymes: A shared virtual memory system for non-coherent tiled many-core architectures. In *ICPADS 2014*, 2014.

[38] Z. Lai, K. T. Lam, C.-L. Wang, J. Su, Y. Yan, and W. Zhu. Latency-aware dynamic voltage and frequency scaling on many-core architectures for data-intensive applications. In *Cloud Computing and Big Data (CloudCom-Asia), 2013 International Conference on*, pages 78–83. IEEE, 2013.

[39] D. Logothetis, S. De, and K. Yocum. Scalable lineage capture for debugging disc analytics. In *Proceedings of the 4th annual Symposium on Cloud Computing*, page 17. ACM, 2013.

[40] T. Ma, L. Chen, C.-L. Wang, and F. C. Lau. G-pass: An instance-oriented security infrastructure for grid travelers. In *Computation and Currency: Practice and Experience*. IEEE, 2006.

[41] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on*, pages 24–24. IEEE, 2006.

[42] G. Malewicz, M. H. Austern, A. J. Bik, J. C. Dehnert, I. Horn, N. Leiser, and G. Czajkowski. Pregel: a system for large-scale graph processing. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 135–146. ACM, 2010.

[43] F. McSherry. Privacy integrated queries. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data (SIGMOD)*. Association for Computing Machinery, Inc., June 2009. URL https://www.microsoft.com/en-us/research/publication/privacy-integrated-queries/.

[44] M. M.J.M., W. C.L., and L. F.C.M. Jessica: Java-enabled single-system-image computing architecture. In *Journal of Parallel and Distributed Computing*, 2000.

[45] P. Mohan, A. Thakurta, E. Shi, D. Song, and D. Culler. Gupt: Privacy preserving data analysis made easy. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, SIGMOD '12, pages 349–360, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1247-9. doi: 10.1145/2213836.2213876. URL http://doi.acm.org/10.1145/2213836.2213876.

[46] L. Nelson, H. Sigurbjarnarson, K. Zhang, D. Johnson, J. Bornholt, E. Torlak, and X. Wang. Hyperkernel: Push-button verification of an os kernel. In *Proceedings of the 26th Symposium on Operating Systems Principles (SOSP '17)*.

[47] O. Ohrimenko, F. Schuster, C. Fournet, A. Mehta, S. Nowozin, K. Vaswani, and M. Costa. Oblivious multi-party machine learning on trusted processors. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 619–636, Austin, TX, 2016. USENIX Association. ISBN 978-1-931971-32-4. URL https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/ohrimenko.

[48] C. Olston, B. Reed, U. Srivastava, R. Kumar, and A. Tomkins. Pig latin: a not-so-foreign language for data processing. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1099–1110. ACM, 2008.

[49] A. Papadimitriou, R. Bhagwan, N. Chandran, R. Ramjee, A. Haeberlen, H. Singh, A. Modi, and S. Badrinarayanan. Big data analytics over encrypted datasets with seabed. In *OSDI*, pages 587–602, 2016.

[50] V. Pappas, V. P. Kemerlis, A. Zavou, M. Polychronakis, and A. D. Keromytis. Cloudfence: Data flow tracking as a cloud service. In *Proceedings of the 16th International Symposium on Research in Attacks, Intrusions, and Defenses - Volume 8145*, RAID 2013, pages 411–431, New York, NY, USA, 2013. Springer-Verlag New York, Inc. ISBN 978-3-642-41283-7. doi: 10.1007/978-3-642-41284-4_21. URL http://dx.doi.org/10.1007/978-3-642-41284-4_21.

[51] pigmix. https://cwiki.apache.org/confluence/display/PIG/PigMix.

[52] R. A. Popa, C. Redfield, N. Zeldovich, and H. Balakrishnan. Cryptdb: protecting confidentiality with encrypted query processing. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, pages 85–100. ACM, 2011.

[53] I. Roy, S. T. V. Setty, A. Kilzer, V. Shmatikov, and E. Witchel. Airavat: Security and privacy for mapreduce. In *Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation*, NSDI'10, pages 20–20, Berkeley, CA, USA, 2010. USENIX Association. URL http://dl.acm.org/citation.cfm?id=1855711.1855731.

[54] F. Schuster, M. Costa, C. Fournet, C. Gkantsidis, M. Peinado, G. Mainar-Ruiz, and M. Russinovich. Vc3: Trustworthy data analytics in the cloud using sgx. In *Security and Privacy (SP), 2015 IEEE Symposium on*, pages 38–54. IEEE, 2015.

[55] F. Shaon, M. Kantarcioglu, Z. Lin, and L. Khan. Sgx-bigmatrix: A practical encrypted data analytic framework with trusted processors. In *Proceedings of the 17th ACM conference on Computer and communications security (CCS '10)*, 2017.

[56] D. Sheng and W. Cho-Li. Error-tolerant resource allocation and payment minimization for cloud system. In *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 2013.

[57] D. Sheng, R. Yves, V. Frederic, K. Derrick, W. Cho-Li, and C. Franck. Optimization of cloud task processing with checkpoint-restart mechanism. In *SC '13*, 2013.

[58] D. Sheng, K. Derrick, and W. Cho-Li. Optimization of composite cloud service processing with virtual machines. In *IEEE Transactions on Computers*, 2014.

[59] A. Smith. Efficient, differentially private point estimators. *arXiv preprint arXiv:0809.4794*, 2008.

[60] A. Smith. Privacy-preserving statistical estimation with optimal convergence rates. In *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing*, STOC '11, pages 813–822, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0691-1. doi: 10.1145/1993636.1993743. URL http://doi.acm.org/10.1145/1993636.1993743.

[61] Spark example. https://spark.apache.org/examples.html.

[62] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

[63] F. Tang, M. Guo, M. Li, C.-L. Wang, and M. Dong. Secure routing for wireless mesh sensor networks in pervasive environments. In *INTERNATIONAL JOURNAL OF INTELLIGENT CONTROL AND SYSTEMS*. IEEE, 2007.

[64] S. Technology. 7 most infamous cloud security breaches. https://www.storagecraft.com/blog/7-infamous-cloud-security-breaches/, Feb. 2017.

[65] C.-C. Tsai, D. E. Porter, and M. Vij. Graphene-sgx: A practical library os for unmodified applications on sgx. In *2017 USENIX Annual Technical Conference (USENIX ATC)*, 2017.

[66] S. Tu, M. F. Kaashoek, S. Madden, and N. Zeldovich. Processing analytical queries over encrypted data. In *Proceedings of the VLDB Endowment*, volume 6, pages 289–300. VLDB Endowment, 2013.

[67] Z. W, W. Cho-Li, , and L. F.C.M. Jessica2: A distributed java virtual machine with transparent thread migration support. In *IEEE Fourth International Conference on Cluster Computing (Cluster2002)*, 2002.

[68] B. Wang, B. Li, and H. Li. Panda: Public auditing for shared data with efficient user revocation in the cloud. *IEEE Transactions on services computing*, 8(1):92–106, 2015.

[69] C. Wang, J. Yang, N. Yi, and H. Cui. Tripod: An efficient, highly-available cluster management system. In *Proceedings of the 7th ACM SIGOPS Asia-Pacific Workshop on Systems*, APSys '16, 2016.

[70] C. Wang, J. Jiang, X. Chen, N. Yi, and H. Cui. Apus: Fast and scalable paxos on rdma. In *Proceedings of the Eighteenth ACM Symposium on Cloud Computing*, pages 17–28. ACM, 2017.

[71] Y. Wang, J. Wei, M. Srivatsa, Y. Duan, and W. Du. Integritymr: Integrity assurance framework for big data analytics and management applications. In *Big Data, 2013 IEEE International Conference on*, pages 33–40. IEEE, 2013.

[72] N. World. Leaked icloud credentials obtained from third parties, apple says. https://www.networkworld.com/article/3184471/security/leaked-icloud-credentials-obtained-from-third-parties-apple-says.html, Feb. 2017.

[73] J. Wu, Y. Tang, G. Hu, H. Cui, and J. Yang. Sound and precise analysis of parallel programs through schedule specialization. In *Proceedings of the ACM SIGPLAN 2012 Conference on Programming Language Design and Implementation (PLDI '12)*, pages 205–216, June 2012.

[74] J. Yang, H. Cui, J. Wu, Y. Tang, and G. Hu. Determinism is not enough: Making parallel programs reliable with stable multithreading. *Communications of the ACM*, 2014.

[75] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker, and I. Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*, pages 2–2. USENIX Association, 2012.

[76] W. Zheng, A. Dave, J. G. Beekman, R. A. Popa, J. E. Gonzalez, and I. Stoica. Opaque: An oblivious and encrypted distributed analytics platform. In *NSDI*, pages 283–298, 2017.

[77] L. Zhiquan, L. King-Tin, W. Cho-Li, , and S. Jinshu. Powerock: Power modeling and flexible dynamic power management for many-core architectures. In *IEEE Systems Journal*, 2016.

RGC Ref No. 17202318

PROJECT FUNDING

3. Cost and justification

(a) Estimated cost and resource implications:

[Detailed justifications should be given in order to support the request for each item below]

(a maximum of 500 words for each box)

	Year 1 ($)	Year 2 ($)	Year 3 ($)	Year 4 ($)	Year 5 ($)	Total ($)
(A) One-line Vote Items						
(i) Supporting Staff Costs						
[please read Section 3(a)(A)(i) of the Explanatory Notes GRF2 carefully]						
Types						
Monthly salary x Nos. x Months						
Research Assistant						$1,193,400
16,575 * 2 * 12	397,800					
16,575 * 2 * 12		397,800				
16,575 * 2 * 12			397,800			

Justification:

This project will require two PhD students S1 and S2 to work for three years. In the first year, S1 will design and fully implement the KAKUTE system (part of Objective 1), and S2 will evaluate its performance and security strength on various real-world big-data queries (part of Objective 1). In the second year, S1 will use KAKUTE to fully develop the proposed Fine-grained Differential Privacy technique (part of Objective 2), and S2 will implement the algorithm proposed for this technique (part of Objective 2). In the third year, S1 will build the secure big-data compiler (part of Objective 3), and S2 will extensively study the privacy guarantee and performance of this compiler on real-world big-data frameworks (part of Objective 3).

(ii) Equipment Expenses

[please itemize and provide quotations for each item costing over $200,000]

Justification:

The PI feels that this proposal does not need to purchase extra equipment. The PI's research group in HKU already has a datacenter consisting of 20 fast computers and networks: each computer has 24 CPU cores, 128GB memory, 6TB hard disk, and 40Gbps network interface card. The HKU CS department's labs also have plenty of personal computers to do research. Therefore, the PI feels that current equipment are already sufficient to conduct this GRF proposal.

Quotation Provided: **Yes** ☐ **No** ☑

(iii) Outsourcing Expenses of Research Work Outside Hong Kong
[please itemize your cost estimation with justification and provide quotations for work costing over $200,000; and provide detailed justification of sample sizes and costs for surveys conducted outside Hong Kong.]

Justification:

N.A.

Quotation Provided: **Yes** ☐ **No** ☑

(iv) General Expenses
[please itemize and provide quotations for services/purchase costing over $200,000; and provide detailed justification of sample sizes and costs for surveys conducted in Hong Kong.]

Oversea visits to implement and study the objectives in this proposal.	20,000	20,000	20,000	0	0	60,000

Justification:

The PI plans to visit Microsoft Research Asia, CMU, and UC Berkeley during the summers to perform a real-world study on the implemented objectives for this proposal. Each visit will take about one month.

Quotation Provided: **Yes** ☐ **No** ☑

(v) Conference Expenses

Attending international conferences to present papers.	20,000	20,000	20,000	0	0	60,000

Justification:

Support for registration fee, traveling expenses, and accommodations for attending bconferences (e.g., ACSAC, SOCC, NSDI, SOSP, OSDI, PLDI, and ASPLOS) held outside Hong Kong.

Sub-total for (A) (One-line Vote Items): **$** 1,313,400

(B) Earmarked Items

(vi) Costs for Employment of Relief Teacher
[see Enclosure III for individual research and Enclosure V for relief support under Humanities and Social Scienecs Panel]

Rank

Monthly salary x Months

Justification:

Current Average Teaching Load: Total 0 classroom hours per academic year [please report UGC-funded programmes only]

N.A.

(vii) Expenses of Research Experience for Undergraduate Student
(see Enclosure VI for Provision of Research Experience for Undergraduate Students)

Justification:

N.A.

(viii) High-performance Computing Services Expenses

Justification:

N.A.

Quotation Provided: **Yes** ☐ **No** ☑

(ix) Research-related Software Licence /Dataset
[Please itemize and provide quotations for each item]

Justification:

N.A.

Sub-total for (B) (Earmarked Items):	**$**	0
(x) Total cost of the proposal (A) + (B)	**$**	1,313,400

(C) Deduction Items

Less :

(xi) University's funding for provision of research experience for undergraduate student	**$**	0
(xii) Other research funds secured from other sources	**$**	0
Sub-total for C (Deduction Items):	**$**	0
(xiii) Amount requested in this application : (A) + (B) - (C)	**$**	1,313,400

(D) Academic Research related to Public Policy Developments

(xiv) Percentage of the total cost of the proposal related to public policy

developments ((A) + (B)) 0%

[see Enclosure VII for Support for Academic Research relating to Public Policy Developments]

(b) Declaration on the Equipment Procurement:

☑ **(i) No procurement of equipment is required**

OR

☐ **(ii) I declare that the equipment indicated in 3(a)(A)(ii) above is not available in the university**

OR

☐ **(iii) I declare that all or some of the equipment (please provide details in the following text box) indicated in Section 3(a)(A)(ii) above is available in the university but cannot be used by me in view of the following reasons (a maximum of 500 words)**

Reasons : (a maximum of 500 words)

N.A.

(c) Declaration on employment of relief teacher:

☑ **(i) No relief teacher is required**

OR

☐ **(ii) I declare that I currently do not hold any grant for employment of relief teacher of any on-going project under UGC/RGC funding schemes**

OR

☐ **(iii) I declare that I hold grant for employment of relief teacher of the following on-going project(s) under UGC/RGC funding schemes (excluding Humanities and Social Sciences Prestigious Fellowship Scheme (HSSPFS)) and undertake to submit the corresponding completion report(s) by 15 April 2018**

(d) Declaration on high-performance computing services:

☑ **(i) No procurement of high-performance computing services is required**

OR

☐ **(ii) I declare that the high-performance computing services indicated in Section 3(a)(B)(viii) above is not available in the university**

OR

☐ **(iii) I declare that all or some of the high-performance computing services (please provide details in the following text box) indicated in Section 3(a)(B)(viii) above is available in the university but cannot be used by me in view of the following reasons(a maximum of 500 words)**

Reasons : (a maximum of 500 words)

N.A.

(e) Declaration on the research-related software licence / dataset:

☑ **(i) No procurement of research-related software licence / dataset is required**

OR

☐ **(ii) I declare that the research-related software licence / dataset indicated in Section 3(a)(B)(ix) above is not available in the university**

OR

☐ **(iii) I declare that all or some of the research-related software licence / dataset (please provide details in the following text box) indicated in Section 3(a)(B)(ix) above is available in the university but cannot be used by me in view of the following reasons (a maximum of 500 words)**

Reasons : (a maximum of 500 words)

N.A.

4. Existing facilities and major equipment available for this research proposal: (a maximum of 400 words)

The PI's research group in HKU already has a datacenter consisting of 20 fast computers and networks: each computer has 24 CPU cores, 128GB memory, 6TB hard disk, and 40Gbps network interface card. The HKU CS department's labs also have plenty of personal computers to do research. Therefore, the PI feels that current equipment are already sufficient to conduct this GRF proposal.

5. Funds secured or to be secured

(a) Other research funds already secured for this research proposal:
[This amount will be deducted from the total cost of the project in Section 3 of Part II above.]

Source	Amount ($)

(b) Other research funds to be or are being sought for this research proposal.
[If funds under this item are secured, the amount of the GRF to be awarded may be reduced]:

Source	Amount ($)

6. Particulars of PI and Co-Is

(a) Investigator(s) information:
Name and Academic Affiliation of Applicant:

	Name	Post	Unit/ Department/ University	Current Member of RGC Council as at the application deadline (Yes or No)	Current Member of RGC Subject Panel as at the application deadline (Yes or No)	Name of RGC Subject Panel
PI	Dr Cui, Heming	Assistant Professor	Department of Computer Science/The University of Hong Kong	No	No	
Co-I(s)	Prof Wang, Cho-Li	Professor (Non-Clinical)	Department of Computer Science/The University of Hong Kong	No	Yes	Engineering

(b) Curriculum vitae (CV) of Applicant(s).
[For the PI and each Co-I, please attach a CV (a maximum of two A4 pages in standard RGC format for attaching PDF documents or a maximum of 800 words for direct input in the text box) per person in the following format.]

i) Name:
ii) Academic qualifications:
iii) Previous academic positions held(with dates):
iv) Present academic position:
v) Previous relevant research work:
vi) Publication records [Please refer to GRF 2 Part II Section 6 for the format required by the RGC]:
Section A - Five most representative publications in recent five years
Section B - Five representative publications beyond the recent five-year period with the latest publication entered first.
vii) Others (please specify):

(c) Plan(s) for collaboration in this application:
[Indicate the role and the specific task(s) the PI and each Co-I , if any, is responsible for.]
[Letter(s) of collaboration should be attached to section 11]

The PI's collaboration role with the co-I and worldwide researchers：
The PI plans to collaborate with researchers from Microsoft Research Asia, CMU, and UC Berkeley. The co-I, Prof. Cho-Li Wang from HKU, is a world-class expert on high-performance computing compilers and big-data computing platforms. The PI has collaborated with the co-I to publish a relevant paper for this proposal in a top journal: TPDS '17. The PI plans to collaborate with the co-I on all the three objectives.

The PI's collaboration tasks with the co-I and worldwide researchers: The PI and his students will conduct the main research role of this project, including developing theoretical models, building tools, and implementing software. The co-I and worldwide collaborators from Microsoft, CMU, and UC Berkeley will provide high-level suggestions on this proposal and apply the systems and algorithms built from this proposal to real-world big-data platforms.
(d) Number of hours per week to be spent by the PI in the proposal: 20 hour(s)

HEMING CUI

Assistant Professor, Computer Science, HKU *heming@cs.hku.hk*

Academic Qualification

- PhD, Computer Science, Columbia University, New York, USA, 2014.
- Master, Computer Science, Tsinghua University, Beijing, China, 2008.
- Bachelor, Computer Science, Tsinghua University, Beijing, China, 2005.

Academic Position

- January 2015 $\sim$ now, Assistant Professor, Computer Science, HKU.
- Website: `http://www.cs.hku.hk/people/profile.jsp?teacher=heming`

Relevant Research Experience

- Secure and fast big-data computing: [Kakute ACSAC '17], [Confluence TPDS '17].
- Dependable distributed systems: [Parrot SOSP '13], [Crane SOSP '15], [Apus SOCC '17].
- Security bug detection: [PLDI '12], [Woodpecker ASPLOS '13].
- Reliable multi-threading: [Loom OSDI '10], [Tern OSDI '10], [Peregrine SOSP '11].

Relevant Publication

- Jianyu Jiang, Shixiong Zhao, Danish Alsayed, Yuexuan Wang, **Heming Cui**, Feng Liang, and Zhaoquan Gu. "Kakute: A Precise, Unified Information Flow Analysis System for Big-data Security". Proceedings of the Annual Computer Security Applications Conference 2017 (**ACSAC '17**).
- Feng Liang, F.C.M. Lau, **Heming Cui**, and C.L. Wang. "Confluence: Speeding Up Iterative Distributed Operations by Key-dependency-aware Partitioning". IEEE Transactions on Parallel and Distributed Systems 2017 (**TPDS '17**).
- Cheng Wang, Jianyu Jiang, Xusheng Chen, Ning Yi, and **Heming Cui**. "APUS: Fast and Scalable PAXOS on RDMA". Proceedings of the ACM Symposium on Cloud Computing 2017 (**SOCC '17**).
- Cheng Wang, Xusheng Chen, Zixu Wang, Youwei Zhu, Heming Cui, and **Heming Cui**. "A Fast, General Storage Replication Protocol for Active-Active Virtual Machine Fault Tolerance". Proceedings of the IEEE 23rd International Conference on Parallel and Distributed Systems 2017 (**ICPADS '17**).
- Cheng Wang, Jingyu Yang, Ning Yi, and **Heming Cui**. "TRIPOD: An Efficient, Highly-available Cluster Management System". Proceedings of the 7th ACM SIGOPS Asia-Pacific Workshop on Systems (**APSys '16**).
- **Heming Cui**, Rui Gu, Cheng Liu, Tianyu Chen, and Junfeng Yang. "Paxos Made Transparent". Proceedings of the 25th ACM Symposium on Operating Systems Principles (**SOSP '15**).
- Junfeng Yang, **Heming Cui**, Jingyue Wu, Yang Tang, and Gang Hu. "Determinism Is Not Enough: Making Parallel Programs Reliable with Stable Multithreading". Communications of the ACM 2014 (**CACM '14**).

- **Heming Cui**, Jiri Simsa, Yi-Hong Lin, Hao Li, Ben Blum, Xinan Xu, Junfeng Yang, Garth Gibson, and Randy Bryant. "Parrot: a Practical Runtime for Deterministic, Stable, and Reliable Threads". Proceedings of the 24th ACM Symposium on Operating Systems Principles (**SOSP '13**).
- **Heming Cui**, Gang Hu, Jingyue Wu, and Junfeng Yang. "Verifying Systems Rules Using Rule-Directed Symbolic Execution". Proceedings of the 18th International Conference on Architecture Support for Programming Languages and Operating Systems (**ASPLOS '13**).
- Jingyue Wu, Yang Tang, Gang Hu, **Heming Cui**, Junfeng Yang . "Sound and Precise Analysis of Parallel Programs through Schedule Specialization". Proceedings of the 33rd ACM SIGPLAN Conference on Programming Language Design and Implementation (**PLDI '12**).
- **Heming Cui**, Jingyue Wu, John Gallagher, Huayang Guo, and Junfeng Yang. "Efficient Deterministic Multithreading through Schedule Relaxation". Proceedings of the 23rd ACM Symposium on Operating Systems Principles (**SOSP '11**).
- **Heming Cui**, Jingyue Wu, Chia-che Tsai, and Junfeng Yang. "Stable Deterministic Multithreading through Schedule Memoization". Proceedings of the Ninth Symposium on Operating Systems Design and Implementation (**OSDI '10**).
- Jingyue Wu, **Heming Cui**, and Junfeng Yang. "Bypassing Races in Live Applications with Execution Filters". Proceedings of the Ninth Symposium on Operating Systems Design and Implementation (**OSDI '10**).

Research Grant (as a PI)

- "GAIA: Strengthening the Reliability of Datacenter Computing via Fast Distributed Consensus", Hong Kong RGC GRF (Ref: HKU 17207117), 2018∼2021, HK $ 500,000.
- "Achieving Strong Fault-tolerance for General Storage Applications via Fast, RDMA-powered PAXOS", the Huawei Innovation Research Program (HIRP), 2017∼2018, HK $ 544,000.
- "FALCON: Modeling, Detecting, and Defending against Concurrency Attacks", HK RGC early career scheme (Ref: HKU 27200916), 2016∼2019, HK $ 618,470.
- "RepBox: Transparent State Machine Replication and its Applications", Croucher Innovation Award, 2016∼2021, HK $ 5,000,000.

(i) Name : **Cho-Li Wang**

(ii) Academic qualifications:

Ph.D. Computer Engineering, University of Southern California, USA, August 1995.
M.S. Computer Engineering, University of Southern California, USA, April 1990.
B.S. Computer Science and Information Engineering, National Taiwan University, June 1985.

(iii) Previous academic positions held (with dates) :

Assistant Professor	Dept. of Computer Science, HKU, 09/1995 to 04/2001
Associate Processor	Dept. of Computer Science, HKU, 17/4/2001 to 30/6/2013.

(iv) Present academic position : Professor

(v) Previous relevant research work :

(vi) Publication records :

Section A - Five most representative publications in recent five years

[1] King Tin Lam, Jinghao Shi, Dominic Hung, **Cho-Li Wang**, Zhiquan Lai, Wangbin Zhu, and Youliang Yan, "Rhymes: A Shared Virtual Memory System for Non-Coherent Tiled Many-Core Architectures," The 20th IEEE International Conference on Parallel and Distributed Systems (ICPADS 2014), Dec. 16 – 19, 2014.

[2] Sheng Di and **Cho-Li Wang**, "Error-Tolerant Resource Allocation and Payment Minimization for Cloud System," IEEE Transactions on Parallel and Distributed Systems (TPDS), Vol. 24, No. 6, pp. 1097-1106, June, 2013.

[3] Sheng Di, Yves Robert, Frederic Vivien, Derrick Kondo, **Cho-Li Wang**, Franck Cappello, "Optimization of Cloud Task Processing with Checkpoint-Restart Mechanism", The International Conference for High Performance Computing, Networking, Storage, and Applications (SC'13), Nov. 17-22, 2013, Denver, Colorado.

[4] Sheng Di, Derrick Kondo, **Cho-Li Wang**, *"Optimization of Composite Cloud Service Processing with Virtual Machines*," *IEEE Transactions on Computers*, June 2014.

[5] Zhiquan Lai, King Tin Lam, **Cho-Li Wang**, and Jinshu Su, "PoweRock: Power Modeling and Flexible Dynamic Power Management for Many-core Architectures," IEEE Systems Journal, Issue: 99, pp. 1-13, 20 January 2016.

Section B - Five representative publications beyond the recent five-year period

[1] W. Zhu, **C.L. Wang**, and F.C.M. Lau, ``JESSICA2: A Distributed Java Virtual Machine with Transparent Thread Migration Support,'' *IEEE Fourth International Conference on Cluster Computing* (Cluster2002) Chicago, USA, Sept. 23-26, **2002**, pp.381-388. (GoogleScholar Citation: 162)

[2] M.J.M. Ma, **C.L. Wang**, F.C.M. Lau, ``JESSICA: Java-Enabled Single-System-Image Computing Architecture,'' *Journal of Parallel and Distributed Computing*, Vol.60, No.10, pp.1194-1222, Oct. **2000**. (GoogleScholar Citation: 123)

[3] K. Hwang, H. Jin, E. Chow, **C.L. Wang**, and Z. Xu; ``Designing SSI Clusters with Hierarchical Checkpointing and Single I/O Space,'' *IEEE Concurrency*, Vol.7, No.1, pp.60-69, Jan-Mar., 1999. (GoogleScholar Citation: 83)

[4] B.W.L. Cheung, C.L. Wang, K. Hwang, A Migrating-Home Protocol for Implementing Scope Consistency Model on a Cluster of Workstations. PDPTA, 1999. (GoogleScholar Citation: 52)

[5] A, Khokhar, M. Shaaban, V.K. Prasanna, and **C.L. Wang**; "Heterogeneous Computing: Challenges and Opportunities," *IEEE Computer*, Vol.26, No.6, pp.18-27, June 1993. (GoogleScholar Citation: 360)

(vii) Others (please specify) :

Editorial board membership of scholarly journals

[1] IEEE Transactions on Cloud Computing (IEEE TCC) (2013-)
[2] IEEE Transactions on Computers, 2006-2010.
[3] Multiagent and Grid Systems , IOS Press, 2005-present
[4] International Journal of Pervasive Computing and Communications (JPCC), 2005-2011.
[5] Journal of Information Science and Engineering (JISE), 2009- present.
[6] ICST Transactions on Scalable Information Systems (SIS), 2009- present.
[7] International Journal of Intelligent Engineering Informatics (IJIEI), 2010- present.
[8] Human-centric Computing and Information Sciences (HCIS), Springer, 06/2011- present.
[9] Journal of Information Technology and Applications, Chung Hua University, 2006- present.

Invited/keynote talks in international/regional conferences

[1] Mar. 18, 2005 (keynote): "High Performance Computing on Clusters: The Distributed JVM Approach" in *The 11th Workshop on Compiler Techniques for High-Performance Computing* (CTHPC), March 17-18, 2005, Tunghai University, Taiwan. (Regional)

[2] Oct. 26, 2005 (keynote): "A New Compositional Adaptation Technique for Pervasive Computing" in *The First Chinese Workshop on Pervasive Computing* (PerCompChina2005), Oct. 26-28, 2005, Kunming, China. (Regional)

[3] Dec. 14, 2007 (invited talk): "Computation Migration Techniques for Grid Computing" in *The 4th Workshop on Grid Technologies and Applications* (WoGTA'07), Taiwan. (Regional)

[4] Oct. 25, 2008 (keynote): "Scaling Horizontally - The Distributed JVM Approach" in *The 7th International Conference on Grid and Cooperative Computing* (GCC 2008), October 24-26, 2008, Shenzhen, China. (International)

[5] Dec. 10, 2008 (keynote): "Scaling Java Program on Clusters: The Distributed JVM Approach," *The 50th Anniversary of Journal of Computer Research and Development*, Beijing, China. (Regional)

[6] May 30, 2010 (keynote): "Towards Easy-to-use PGAS Parallel Programming – The Distributed JVM Approach" in *the Third International Joint Conference on Computational Sciences and Optimization* (CSO 2010), Huangshan Anhui, China. (International)

[7] Aug. 5, 2010 (keynote): "Towards High Productivity Computing - The Distributed JVM Approach" in *the IET International Conference on Frontier Computing – Theory, Technologies and Applications* (FC2010), Taichung, Taiwan, August 4-6, 2010. (International)

[8] Aug. 31, 2011 (keynote): "Towards Agile Cloud Computing by Computation Migration" in *the Third Cloud Computing Technologies and Applications Workshop*, National Center for High-performance Computing (NCHC), Hsinchu, Taiwan. (Regional)

[9] Jan. 16-17,2013 (invited talk): "System Software Challenges for Big Data Computing" in 2nd ICT Industry and Technology Trend Forum, organized by Huawei, Shenzhen China. (International)

[10] Dec. 16-19, 2013 (keynote): 2013 International Conference on Cloud Computing and Big Data (CloudCom-Asia), Fuzhou, China

Best Paper Awards:

[1] ``Smart Instant Messenger in Pervasive Computing Environments," *The First International Conference on Grid and Pervasive Computing* (GPC2006), May 3-5, 2006.
[2] 'Handoff Performance Comparison of Mobile IP, Fast Handoff and mSCTP in Mobile Wireless Networks', *International Symposium on Parallel Architectures, Algorithms, and Networks* (I-SPAN), May 7-9, 2008.
[3] "eXCloud: Transparent Runtime Support for Scaling Mobile Applications" in *2011 IEEE International Conference on Cloud and Service Computing* (CSC2011), Dec. 12-14, 2011.
[4] "Defeating Network Jitter for Virtual Machines", *The 4th IEEE International Conference on Utility and Cloud Computing* (UCC 2011), Dec. 5-8, 2011. (Best Student Paper Award)

RGC Ref No. 17202318

DECLARATION OF SIMILAR OR RELATED PROPOSALS & GRANT RECORD

[Please refer to GRF2 for information required and implications for non-disclosure of similar or related proposals]

7. Re-submission of a proposal not supported previously

(a) Is this proposal a re-submission or largely similar to a proposal that has been submitted to but not supported by the UGC/RGC or other funding agencies?

Yes ☐ **No** ☑

If yes, please state the funding agency(ies) and the funding programme(s):

Reference No(s). [for UGC/RGC projects only]:

Project title(s) [if different from Section 1(a) of Part I above]:

Date(month/year) of application:

Outcome:

(b) If this application is the same as or similar to the one(s) submitted but not supported previously, what were the main concerns / suggestions of the reviewers then?

(c) Please give a brief response to the points mentioned in Section 7(b) above, highlighting the major changes that have been incorporated in this application.

8. Grant Record of Investigator(s)

(a) PI - Details of research projects undertaken and proposals submitted by the PI (in a PI/PC or Co-I/Co-PI capacity) including (i) completed research projects funded from all sources (irrespective of whether from UGC/RGC) in the past five years; (ii) on-going research projects funded from all sources (irrespective of whether from UGC/RGC); (iii) terminated projects funded by UGC/RGC in the past five years; (iv) unsuccessful proposals or withdrawn projects submitted to UGC/RGC in the past five years; and (v) proposals pending funding approval. If you have any research project(s) / proposal(s) (irrespective of whether submitted to/funded by UGC/RGC and not limited to the past five years) which is/are similar or related to this application, please include in the table below

and provide an explanation on the differences between that/those project(s)/proposal(s) and this application (a maximum of 400 words). [If you have difficulty in making the declaration, please explain.] Please add a new table for each project/proposal.

Project/Proposal Ref No.	17207117
Name of Investigator(s)	Cui, Heming
Project Title	GAIA：Strengthening the Reliability of Datacenter Computing via Fast Distributed Consensus
Project Objective (not applicable for unsuccessful proposals or withdrawn projects)	1. [Create a fast distributed consensus protocol]. We will develop an RDMA-powered distributed consensus algorithm and its implementation protocol APUS. This new protocol aims to be one or two orders of magnitude faster than traditional consensus protocols. 2. [Construct a first datacenter scheduler for improving application availability]. This scheduler will replicate essential application components by integrating APUS with popular schedulers. We will develop a new scheme to seamlessly manage both resource allocation and replication logic, then this scheduler can efficiently support applications. 3. [Build a new fault-tolerant VM for improving application availability]. We will leverage the VM hypervisor layer to transparently enforce same application inputs across VM replications. Compared to existing VM fault-tolerance techniques (e.g., primary-backup), our VM will greatly reduce the amount of transferred memory across VM replications, saving most time and network bandwidth. We will also develop a new algorithm to efficiently track minor different memory across replications.
Status	On-going
Capacity	PI
Funding Source(s) and Amount($)	GRF $ 500,000

UGC/RGC Funding (Yes or No)	
Start Date (if applicable)	01-09-2017
Estimated Completion Date (if applicable)	31-08-2020
Number of Hours Per Week Spent by the PI in Each On-going Project*	20
Similar or related to the current application	NA
If yes, please explain the differences	

Project/Proposal Ref No.	
Name of Investigator(s)	Cui, Heming
Project Title	Achieving Strong Fault-tolerance for General Storage Applications via Fast, RDMA-powered PAXOS
Project Objective (not applicable for unsuccessful proposals or withdrawn projects)	State machine replication (SMR) uses PAXOS to enforce the same sequence of inputs for a storage application (e.g., Redis) running on replicas of computer hosts, tolerating various types of failures. This strong fault-tolerance makes PAXOS attractive to build an active-active replication service for general storage applications. Unfortunately, traditional PAXOS protocols incur prohibitive performance overhead for storage applications due to the high PAXOS consensus latency on TCP/IP. Worse, our study finds that the consensus latency of extant PAXOS protocols increases drastically when

	more concurrent client connections or hosts are added. We propose APUS, the first RDMA-powered PAXOS protocol that aims to be fast and scalable to client connections and hosts. APUS automatically runs an unmodified storage application on multiple replicas within the same datacenter, intercepts inbound socket calls (e.g., recv()) in this application, and uses fast RDMA primitives to make replicas receive the same input requests sent from wide- or local-area networks.
Status	On-going
Capacity	PI
Funding Source(s) and Amount($)	Huawei Technologies Inc $ 544,000
UGC/RGC Funding (Yes or No)	N
Start Date (if applicable)	01-09-2017
Estimated Completion Date (if applicable)	31-08-2018
Number of Hours Per Week Spent by the PI in Each On-going Project*	10
Similar or related to the current application	NA
If yes, please explain the differences	

Project/Proposal Ref No.	27200916

RGC Ref No. 17202318

Name of Investigator(s)	Cui, Heming
Project Title	FALCON：Modeling, Detecting, and Defending against Concurrency Attacks
Project Objective (not applicable for unsuccessful proposals or withdrawn projects)	1. [To develop a general, rigorous concurrency attack model]. We will conduct an extensive study on real-world multithreaded programs, summarize general elements on how concurrency bugs propagate to attacks, and leverage our expertise on precise program analysis methods to develop the first concurrency attack model. 2. [To construct a systematic concurrency attack detection approach]. With the concurrency attack model, we will construct an approach to detect as many as concurrency attacks for the software testing phase. This approach will leverage recent automated program analysis techniques to identify concurrency bugs in program source code and vulnerable instructions these bugs may propagate to. 3. [To build a runtime defense infrastructure]. To defend against concurrency attacks that may be missed by detection tools, we will build a defense infrastructure for the software deployment phase. This infrastructure will leverage recent advanced replication techniques to tolerate concurrency attacks and checkpoint techniques to recover program execution state (e.g., memory and files) from attacks.
Status	On-going
Capacity	PI
Funding Source(s) and Amount($)	ECS $ 568,470
UGC/RGC Funding (Yes or No)	
Start Date (if applicable)	01-09-2016

Estimated Completion Date (if applicable)	31-08-2019
Number of Hours Per Week Spent by the PI in Each On-going Project*	20
Similar or related to the current application	NA
If yes, please explain the differences	

Project/Proposal Ref No.	
Name of Investigator(s)	Cui, Heming
Project Title	RepBox: Transparent State Machine Replication and its Applications
Project Objective (not applicable for unsuccessful proposals or withdrawn projects)	The fault tolerance and the theoretically proven safety of State Machine Replication (SMR) makes it attractive for implementing a principled system for general programs, especially server programs that demand high availability. Unfortunately, existing SMR systems unrealistically assume deterministic code execution when most server programs are nondeterministic multithreaded programs. Moreover, existing SMR systems typically provide narrow state machine interfaces, and orchestrating a server program into these interfaces can be strenuous and error-prone. This proposal presents RepBox, an SMR system that transparently replicates general server programs for high availability. To address nondeterminism, RepBox leverages deterministic multithreading to keep replicas in sync. RepBox addresses a difficult network input timing problem via a new technique called time bubbling. I envision that RepBox has

	broad applications on improving software reliability and security.
Status	On-going
Capacity	PI
Funding Source(s) and Amount($)	Croucher foundation $ 5,000,000
UGC/RGC Funding (Yes or No)	N
Start Date (if applicable)	01-01-2016
Estimated Completion Date (if applicable)	31-12-2021
Number of Hours Per Week Spent by the PI in Each On-going Project*	20
Similar or related to the current application	NA
If yes, please explain the differences	

*** The PI is not required to report on the time spent in the capacity of Co-I in GRF / Joint Research Schemes projects.**

(b) Co-I(s) – Details of on-going research projects funded from all sources (irrespective of whether from UGC/RGC) undertaken by each Co-I (in a PI/PC capacity) and proposals pending funding approval (in a PI/PC capacity). If you have any research project(s)/proposal(s) (irrespective of whether submitted to/funded by UGC/RGC and not

limited to the past five years) which is/are similar or related to this application, please include in the table below and provide an explanation on the differences between that/those project(s)/proposal(s) and this application (a maximum of 400 words). [If you have difficulty in making the declaration, please explain.] Please add a new table for each project/proposal.

Name of Co-I(s) and Capacity	Prof Wang, Cho-Li (PI)
Project Title	Big-Little Heterogeneous Computing with Polymorphic GPU Kernels
Project/Proposal Ref No.	17257416
Status	On-going
Funding Source(s) and Amount($)	GRF HK$ 675,647
Start Date	01-10-2016
Estimated Completion Date	30-09-2019
Similar or related to the current application	NA
If yes, please state the project objectives and explain the differences	

Name of Co-I(s) and Capacity	Prof Wang, Cho-Li (PI)
Project Title	Software Architecture for Fault-Tolerant Multicore Computing with Hybridized Non-Volatile Memories

Project/Proposal Ref No.	17210615
Status	On-going
Funding Source(s) and Amount($)	GRF HK$ 871,036
Start Date	01-09-2015
Estimated Completion Date	31-08-2018
Similar or related to the current application	NA
If yes, please state the project objectives and explain the differences	

ANCILLARY INFORMATION

9. Research Ethics/Safety Approval and Access to Government/ Official/ Private Data and Records

[Please refer to GRF2 Part II Section 9 for the responsibilities and implications]

(a) Research Ethics/Safety Approval

(i) I confirm that the research proposal ☐ **involves /** ☑ **does not involve human subjects.**

(ii) Please tick the appropriate boxes to confirm if approval for the respective ethics and/or safety issues is required and has been / is being obtained from the PI's university. PIs are encouraged to seek necessary approval (except for human research ethics (clinical)) before application deadline as far as possible

	Approval not	Approval being	Approval

	required	sought	obtained
(1) Animal research ethics	☑	☐	☐
(2) Biological safety	☑	☐	☐
(3) Ionizing radiation safety	☑	☐	☐
(4) Non-ionizing radiation safety	☑	☐	☐
(5) Chemical safety	☑	☐	☐
(6) Human research ethics (non clinical)	☑	☐	☐
	Approval not required	**Approval being sought**	**Approval will be sought if funded**
(7) Human research ethics (clinical)	☑	☐	☐

(iii) If approval is required by other authorities, please indicate *below* the names of the authorities and the prospects of obtaining such approval. If not applicable, please put down "N.A.".

N.A.

(b) Access to Government/ Official/ Private Data and Records

(i) Is access to Government / official / private data and records critical to the research proposal?

☐ **Yes**

☑ **No**

If approval is required, please indicate below the names of the agency(ies) of obtaining such approval.

(ii) Please tick in the appropriate boxes to confirm if approval for access to the related data/records has been / is being obtained from the relevant agency(ies). If approval has been obtained, please provide evidence.

List of agency(ies)	Approval not required	Approval being sought	Approval obtained

[Note: PIs are encouraged to seek necessary approval before application deadline as far as possible.]

10. Release of completion report, data archive possibilities and public access of publications resulting from research funded by the RGC

(a) Is the proposed project likely to generate data set(s) of retention value?

Yes ☐ **No** ☑

If yes, please describe the nature, quantity and potential use of the data set(s) in future.

Nil

(b) Are you willing to make the data set(s) available to others for reference twelve months after the publication of research results or the completion of this proposed project?

Yes ☐ **No** ☐

I/We understand that the RGC will release the completion report to the public and only considers data archiving requests after the completion of the RGC-funded project. The RGC has full discretion in funding the archiving requests. Data sets archived with RGC funds will require users to acknowledge the originator and the RGC. The originator will also be provided with copies of all publications derived from the use of the data.

I undertake to include in the project completion report the URL links to the university's repository or the publisher's websites so that the public could have quick and easy access to the manuscripts or journal articles. I will also consider to include in the research completion report the data repository where research data of the project could be accessed and shared, where appropriate.

I undertake that upon acceptance of a paper for publication,

(i) I will check whether the publisher already allows (A) full open access to the publisher's version, or (B) my depositing a copy of the paper (either the publisher's version or the final accepted manuscript after peer-review) in the university's repository for open access;

(ii) if both (i) (A) and (B) are not allowed, I will request the publisher to allow me to place either version in my university's repository for restricted access immediately upon publication or after an embargo period of up to twelve months if required by the publisher; and

(iii) subject to the publisher's agreement on (i) or (ii) above, I will deposit a copy of the publication in my university's repository as early as possible but no later than six months after publication or the embargo period, if any.

11. Education Plan, Technology Transfer Plan, Letters of Collaboration and Supporting

Documents

(A maximum of 20 words for each box to caption each uploaded pdf document)

Appendix 1: Education Plan (up to one A4 page)

Appendix 2: Technology Transfer Plan

Education Plan

This proposal will provide substantial and comprehensive research trainings on security techniques to both postgraduate and undergraduate students. Trainings include literature study, algorithm and software design, systems implementation and evaluation on real-world applications, team work, writing academic articles, and giving technical talks. We believe that these trainings will provide a solid foundation to students for their future academic and industrial careers. We elaborate our proposed training programs below.

1. **Final-year project**. The PI will advise final-year undergraduate students, and a project could be: (1) studying the applicability of our systems and techniques on a variety of real-world big-data applications, including Spark, Pig, and Hadoop; (2) implementing evaluation scripts that can automatically run our systems with real-world big-data algorithms and collect performance results; (3) in-depth analysis of security techniques, including data flow tracking, encryption, SGX hardware, and secure compilers. By conducing these hand-on tasks, undergraduate students can get good experience on understanding and using real-world software, performance diagnoses, and programming with scripting languages. We believe these are great ways to get the students interested in research and build a concrete sense on real-world security software.

2. **Research seminar**. The PI will hold regular weekly research seminars with the postgraduate students as well as undergraduate students. Each seminar will either (1) assign a postgraduate student to give a talk on the design, implementation, evaluation, and writing of a top systems paper, or (2) invite external experts to present their security systems or industrial experience. By participating these seminars, all students can learn the whole process of doing research, implementing practical software, and even paper writings.

3. **One-on-one meeting**. The PI will also have regular one-on-one meetings with each undergraduate student involved in the project. The PI will consult the progress, issues, and questions from each student and give concrete suggestions. The goal is to make sure the PI knows the performance of the research students well and make sure the students are on the right track.

4. **Student mentor system**. Each research postgraduate student will act as a mentor for an undergraduate student researcher. This could help undergraduate student easy to join research teams and solve technical problems smoothly. The postgraduate students can also learn how to act as a team leader in this mentoring process.

5. **Case study**. A few undergraduate students with strong motivation and programming skills will be invited to apply existing top security techniques into our infrastructures. They will learn how to install a tool on one of our replicas, and how to inspect the results reported from these tools, and how to analyze the security implications of these results. This process can help undergraduate students to get familiar with various research fields broadly.

Technology Transfer Plan

Publication. To attract insterests from industry and create impacts, The PI expects to publish his research work in top conferences and journals. The PI has published several papers in top conferences in two major research fields: (1) software systems and security systems, including [Kakute ACSAC '17], [Apus SOCC '17], [Crane SOSP '15], [Parrot SOSP '13], [Peregrine SOSP '11], [Tern OSDI '10], and [Loom OSDI '10]; and (2) programming languages, including [Woodpecker ASPLOS '13] and [Wu PLDI '12]. The PI has also published in systems flag-ship journals, including [Confluence TPDS '17] and [StableMT CACM '14]. We believe that this proposal will continue to produce research impact and be present in these conferences and journals.

Industrial research grant. To set up a direct technology transfer channel with industry, the PI plans to proactively apply for research grants from industry. Such grants can promote lots of technology transfer opportunities, including absorbing more real-world significant problems, accessing realistic data sets, invoking joint patent applications, and turning the outcomes of this proposal into practical products. For instance, the PI's [Apus SOCC '17] paper is funded by a GRF grant (Ref: HKU 17207117), and this paper has received a grant award from the competitive Huawei Innovation Research Program (HIRP) in 2017. The PI believes that the success of this proposal will attract more industrial research grants and launch more technology transfer opportunities, because industry cares about big-data privacy very much.

Open source. As a tradition in the PI's research, the PI would like to share source code and evaluation results with the research community and industry. The PI expects to make the outcomes of this proposal open source after it is done. The PI has made several systems and their evaluation results open source, including:

1. Kakute ACSAC '17: `https://github.com/hku-systems/kakute`
2. Apus SOCC '17: `https://github.com/hku-systems/apus`
3. Crane SOSP '15: `http://github.com/columbia/crane`
4. Parrot SOSP '13: `http://github.com/columbia/smt-mc`
5. LOOM OSDI '10: `http://github.com/columbia/loom`

Media. To attract attention from industry, the PI expects to share his thoughts and results with the media and people out of his research fields. The PI's previous research addresses important problems and he has built several secure and reliable systems. These systems and their results have been featured in various international technology websites, including `seclists.org` (Security Mailing List Archive), `ACM Tech News`, `TG Daily`, and `Phys.org`. One can google "heming cui seclists" and she will find some Linux kernel compromise bugs (CVE-2017-7533 and CVE-2017-12193) detected by the PI's recent security tools. The PI's bug reports have caused many operating systems vendors (e.g., RedHat and Android) to patch their latest operating systems software immediately in 2017. One can also google "heming cui croucher" and she/he will find articles that describe RepBox, a dependable distributed system built by the PI. The PI will continue to share the achievements from this proposed project with media.