EVA: Efficient Versatile Auditing Scheme for IoT-Based Datamarket in Jointcloud

Ke Huang, *Student Member, IEEE*, Xiaosong Zhang, Yi Mu, *Senior Member, IEEE*, Fatemeh Rezaeibagha, Xiaofen Wang, Jingwei Li, Qi Xia, and Jing Qin

***Abstract*—Cloud storage offers convenient outsourcing services to users, and it serves as a basic platform to drive Internet-of-Things (IoT) where massive devices are connected to the cloud storage and interact with each other. However, cloud storage is more than a data warehouse. In the literature, data market was proposed as a novel model to empower IoT, where data are circulated as merchandise in the digital marketplace with financial activities. When storing IoT data in cloud storage, security and efficiency rules should be applied. Meanwhile, data dynamics is counted as a critical factor to the feasibility of datamarket as data are supposed to be manipulated through circulation and exploitation for IoT. Another issue is the single-point-of-failure (SPoF) of cloud server in which the initiative of jointcloud was suggested. Since providing data security, efficiency, and dynamics simultaneously is challenging, in this article, we propose a versatile auditing scheme (EVA) as a solution to problems. Our proposal ensures that data are securely, efficiently, and dynamically stored in the jointcloud meanwhile supported by data trades via blockchain. We give a comprehensive security analysis based on our security definitions and experiments to support our claims. The evidence has shown that our EVA is efficient for processing large files when proper parameters are chosen.**

***Index Terms*—Auditing, data dynamic, jointcloud, security.**

I. Introduction

THE cloud storage is a popular platform to store massive outsourced data at large scale. With advances in system design, more and more heterogeneous devices (such as mobile devices, sensing devices, etc.) are deployed to facilitate people's daily life. Meanwhile, the notion of Internet-of-Things (IoT) [1] seeks to enable interactions and connections between these devices and exploit data derived for better decision-making. In addition, single-point-of-failure (SPoF) is another issue to consider if data services are supported by a single server only. According to [2], once such server has been breached, it will cause severe losses to users. Therefore, unifying cloud servers together as a jointcloud to offer stable services is crucial [3].

Cloud storage is not merely a data warehouse; it can be used to boost IoT (as pointed out by Truong and Dustdar [4]). This requires further exploiting data dynamics which indicates both changing data structure [5] dynamically and circulating data as a merchandise in marketplace [6]. As a novel idea, Mišura and Žagar [7] proposed the notion of datamarket to power IoT by setting up a digital marketplace to facilitate data trading activities. It involves financial incentives to break the reluctance of users to share their data for IoT use. With the recent emergence of blockchain [8] (a decentralized and public trust layer), such an incentive mechanism is easy to achieve [9]. In this article, we give a concrete design for datamarket to drive IoT [7].

When storing IoT data in datamarket, generic rules regarding data integrity and efficiency are applied. For the former rule, Ateniese *et al.* proposed provable data possession (PDP) [10] as an auditing mechanism which ensures that data are intact on the server without retrieving it. The latter one often implies encrypted deduplication where the data are encrypted at all times and will be deleted if it is redundant (i.e., when a duplicated copy is detected). Although some works proposed solutions to the above issues in one framework (like [11] and [12]), they can only work in static archive where the data structure is fixed and manipulation is not allowed. As discussed earlier, data dynamics is important for the availability of datamarket. Therefore, it calls for consideration of data security, efficiency, and dynamics under one infrastructure. As we noted in our previous work [13], a trivial combination of static and dynamic settings result in contradiction. Therefore, devising an inclusive scheme to capture the above matters for datamarket is a challenging work.

Based on the above, we focus on a comprehensive auditing scheme to drive datamarket. It is designed to capture security, efficiency, and dynamics for IoT in a jointcloud. Our contributions can be highlighted as follows.

Manuscript received June 14, 2019; revised August 18, 2019; accepted September 29, 2019. Date of publication October 8, 2019; date of current version February 11, 2020. This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB0802300, in part by the National Natural Science Foundation of China under Grant 61602092, Grant U1833122, and Grant 61872087, in part by the open project of the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences under Grant 2019-MS-05, and in part by the foundation from the State Key Laboratory of Integrated Services Networks, Xidian University under Grant ISN18-09. *(Corresponding author: Xiaosong Zhang.)*

K. Huang, X. Zhang, J. Li, and Q. Xia are with the Center for Cyber Security, College of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: kh936@uowmail.edu.au; johnsonzxs@uestc.edu.cn; jwli@uestc.edu.cn; xiaqi@uestc.edu.cn).

Y. Mu is with the Fujian Provincial Key Laboratory of Network Security and Cryptology, College of Mathematics and Informatics, Fujian Normal University, Fuzhou 350007, China (e-mail: ymu.ieee@gmail.com).

F. Rezaeibagha is with Murdoch University, Perth, WA 6150, Australia.

X. Wang is with the Center for Cyber Security, College of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China, and also with the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China (e-mail: wangxuedou@sina.com).

J. Qin is with Shandong University, Jinan 250100, China.

Digital Object Identifier 10.1109/JIOT.2019.2945921

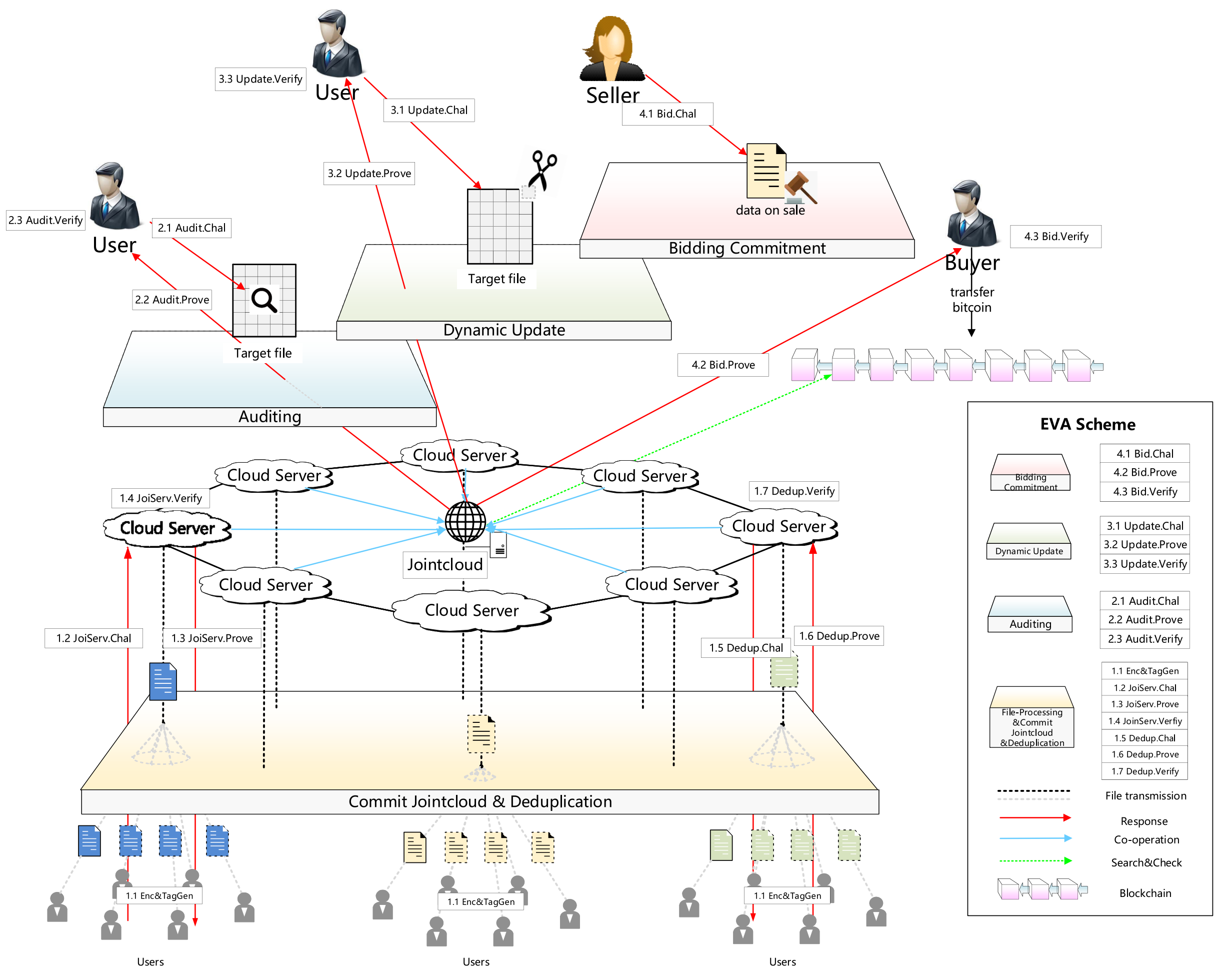


Fig. 1. Framework of datamarket.

1) We propose an efficient versatile auditing (EVA) scheme and use it to drive datamarket. Our EVA scheme achieves data auditing, encrypted deduplication, and dynamic update in a jointcloud. We also propose a bidding commitment based on blockchain to enable data trades. It allows data to be sold to the highest bidder. Our EVA scheme achieves the above functions by one set of parameters while assuming contradictions brought by deduplication and update [14] is trivially solved.
2) We give security definitions and analysis on our proposed EVA scheme for each function. The hardness assumptions guarantee that our security is hard to break.
3) We conduct comprehensive experiments on our proposed EVA; the results showed that our EVA is efficient to deduplicate large files, and performance of auditing and dynamic update scales with proper parameters. Meanwhile, our bidding commitment is acceptably efficient.

II. Preliminary

A. Complexity Assumptions

Let G be a cyclic multiplicative group generated by g with prime order q. We informally state the following assumptions.

Discrete Logarithm Problem: Given $g^a \in G$, where $a \in_R Z_q$, computing a is hard.

Decisional Diffie–Hellman Problem: Given $g, g^a, g^b, g^c \in G$, where $a, b, c \in_R Z_q$, deciding whether $c = ab$ is hard.

Computational Diffie–Hellman Problem: Given $g, g^a, g^b \in G$, where $a, b \in_R Z_q$, computing g^{ab} is hard.

In a gap Diffie–Hellman (GDH) group, computational Diffie–Hellman problem (CDHP) is hard and decisional Diffie–Hellman problem (DDHP) is easy on it. We say $\langle g, g^a, g^b, g^c\rangle$ is a Diffie–Hellman tuple if $c = ab \bmod q$.

Bilinear Pairing: Given two multiplicative groups G and G^T with the same group order q and generator g, denote $\hat{e} : G \times G \to G^T$ as the symmetric bilinear map, where $\hat{e}(x^a, y^b) = \hat{e}(x, y)^{ab}$ for all $x, y \in G$ and $a, b \in_R Z_q$. Additionally, computing $\hat{e}(g, g) \neq 1$ is efficient.

III. Definitions of EVA

A. System Model of Datamarket

The framework of our datamarket (Fig. 1) is dominated by our EVA scheme which consists of six protocols, including file processing, commit jointcloud, deduplication, auditing, dynamic update, and bidding commitment. There are two major typical participants, including user and cloud server.

Specifically, our datamarket allows multiple servers to unify as a jointcloud (not a substantial entity but a symbol of the union) to offer services unitarily. To clarify, when a user wishes to outsource file to cloud servers. it first runs algorithm file processing, then executes commit jointcloud protocol to negotiate with designated servers for data services. After that, the other protocols can be executed to offer users with efficient data outsourcing, integrity check, data dynamics, and data sales. Our EVA covers the most popular data services witnessed in the current cloud business.

B. Security Requirement of EVA

Privacy: Our proposed EVA is private if it is indistinguishable against chosen distribution attacks when assuming the message is unpredictable (PRV\$-CDA [15]). Suppose $\mathcal{A}$ as an efficient adversary against our scheme, we require that $\mathcal{A}$ cannot win the experiment PRV\$-CDA$^{\mathcal{A}}_{\mathsf{EVA}}(\lambda)$ with no negligible advantage, i.e., PRV\$-CDA$^{\mathcal{A}}_{\mathsf{EVA}}(\lambda) \leq \nu(\lambda)$, where ν is a negligible function.

Experiment: PRV\$-CDA$^{\mathcal{A}}_{\mathsf{EVA}}(\lambda)$

$\mathcal{M} \leftarrow \mathcal{A}$

Here, $\mathcal{M}$ denotes an unpredictable block source [15]. It is a polynomial time algorithm which on input λ, outputs (M, Z). The guessing probability of $\mathsf{M} : \mathsf{GP}_{\mathsf{M}}$ is negligible

$\mathsf{param}_{\mathsf{EVA}} \leftarrow \mathsf{Setup}(\lambda)$

$b \in \{0, 1\} \leftarrow \$\mathcal{A}$

$(\mathsf{M}, Z) \leftarrow \mathcal{M}(\lambda)$

If $b = 0$, set $m_0 = \mathsf{M}$

If $b = 1$, set $m_1 = \{0, 1\}^{|m_0|}$

Run algorithm $(c_b, \sigma) \leftarrow \mathsf{Enc\&TagGen}(\mathsf{param}_{\mathsf{EVA}}, m_b)$ to derive c_b as file ciphertext and corresponding tags $\sigma = \{\sigma_i\}_{0 \leq i \leq n}$.

$b^* \leftarrow \mathcal{A}^{H_1, H_2}(\mathsf{param}_{\mathsf{EVA}}, c_b, \sigma, Z)$

Return 1 if $b = b^*$

else, return 0.

Auditing Soundness: Auditing soundness asks that no cheating prover can produce a valid proof to pass the verification of auditing without actually storing the challenged file. Similarly, this is captured by the experiment Aud$^{\mathcal{A}}_{\mathsf{EVA}}$.

Experiment: Aud$^{\mathcal{A}}_{\mathsf{EVA}}(\lambda)$

$\mathsf{param}_{\mathsf{EVA}} \leftarrow \mathsf{Setup}(\lambda)$

On a challenge achal $\leftarrow \mathsf{Audit.Chal}()$ where file $c[1]|| \cdots ||c[n]$ is outside of $\mathcal{A}$'s view

$(aprf*) \leftarrow \mathcal{A}^{H_1}(\mathsf{param}_{\mathsf{EVA}}, \text{achal})$

Here, H_1 is random oracle.

Return 1 if $1 \leftarrow \mathsf{Audit.Verify}(\mathsf{param}_{\mathsf{EVA}}, \text{achal}, aprf^*, k)$

else, return 0.

Deduplication Soundness: Update soundness asks that no cheating prover can forge a proof to pass the verification of PoW.Verify (i.e., proof-of-ownership (PoW) [16]) without actually possessing the file (i.e., $\mathcal{A}$ may get hold of a small fraction of the file but never the entire one). Similarly, this is captured by the experiment PoW$^{\mathcal{A}}_{\mathsf{EVA}}(\lambda)$.

Experiment: PoW$^{\mathcal{A}}_{\mathsf{EVA}}(\lambda)$

$\mathsf{param}_{\mathsf{EVA}} \leftarrow \mathsf{Setup}(\lambda)$

$(ssk, spk) \leftarrow UKeyGen(\lambda)$

On a challenge $dchal \leftarrow \mathsf{PoW.Chal}()$ on file $m = m[1]|| \cdots ||m[n]$ which is out of $\mathcal{A}$'s view

$(dprf*) \leftarrow \mathcal{A}^{H_1, \mathsf{MHT}}(\mathsf{param}_{\mathsf{EVA}}, dchal, \ldots,)$

Here, H_1 and MHT are random oracles, and MHT answers queries for the ith block of m by returning c_i for some $\phi < 1$ (percentage of file leakage).

Return 1 if $1 \leftarrow \mathsf{PoW.Verify}(dchal, dprf^*, \ldots,)$

else, return 0.

Update Soundness: Update soundness asks that no cheating prover can produce a valid proof to pass update verification without actually performing the update operation. It further requires satisfying collision resistance which asks that it is hard to find a collision without private key x. Similarly, this is captured by the experiment Upd$^{\mathcal{A}}_{\mathsf{EVA}}(\lambda)$.

Experiment: Upd$^{\mathcal{A}}_{\mathsf{EVA}}(\lambda)$

$\mathsf{param}_{\mathsf{EVA}} \leftarrow \mathsf{Setup}(\lambda)$

$\{x_i, y_i\}_{1 \leq i \leq k} \leftarrow \mathsf{SKeyGen}(\mathsf{param}_{\mathsf{EVA}})$

$y \leftarrow \mathsf{JKeyGen}(\mathsf{param}_{\mathsf{EVA}}, \{x_i, y_i\}_{1 \leq i \leq k})$

On a challenge $uchal \leftarrow \mathsf{PoW.Chal}()$

$(uprf*) \leftarrow \mathcal{A}^{H_1, Prove}(\mathsf{param}_{\mathsf{EVA}}, uchal)$

Here, H_1 and *Prove* are random oracles where *Prove* returns update proof by running algorithm Update.Prove.

Return 1 if $1 \leftarrow \mathsf{PoW.Verify}(uchal, uprf^*, \ldots,)$ and $uprf*$ has never been queried

else, return 0.

Bidding Soundness: Bidding soundness asks that no cheat adversary could forge a bidding proof without private key x. Similarly, this is captured by the experiment Bid$^{\mathcal{A}}_{\mathsf{EVA}}(\lambda)$.

Experiment: Bid$^{\mathcal{A}}_{\mathsf{EVA}}(\lambda)$

$\mathsf{param}_{\mathsf{EVA}} \leftarrow \mathsf{Setup}(\lambda)$

$\{x_i, y_i\}_{1 \leq i \leq k} \leftarrow \mathsf{SKeyGen}(\mathsf{param}_{\mathsf{EVA}})$

$y \leftarrow \mathsf{JKeyGen}(\mathsf{param}_{\mathsf{EVA}}, \{x_i, y_i\}_{1 \leq i \leq k})$

On a challenge $bchal \leftarrow \mathsf{Bid.Chal}()$

$(bprf^*) \leftarrow \mathcal{A}^{H_1, \mathsf{PrfVer}}(\mathsf{param}_{\mathsf{EVA}}, dchal)$

Here, H_1 and *Prove* are random oracles where *Prove* returns update proof to query by running algorithm Bid.Verify.

Return 1 if $1 \leftarrow \mathsf{Bid.Verify}(\mathsf{bchal}, \epsilon', \mathsf{bprf}^*, \Delta\mathsf{t}, \mathsf{t})$, where $bprf^*$ has never been queried.

else, return 0.

IV. Proposed EVA

A. System Setup

Setup *$(\lambda) \rightarrow (\mathsf{param}_{\mathsf{EVA}})$:* On input a security parameter λ, choose two groups G and G_T with prime order p and generator g. Set bilinear map as $\hat{e} : G \times G \rightarrow G_T$. Randomly select s elements $u_1, \ldots, u_s \xleftarrow{R} G$. Set hash functions as

$$H_1 : \{0, 1\}^* \rightarrow G, \ H_2 : \{0, 1\}^* \rightarrow Z_p, \ H_3 : G \rightarrow \{Z_p\}^s.$$

Output $\mathsf{param}_{\mathsf{EVA}} = \{G, G^T, g, p, \hat{e}, u_1, \ldots, u_s, H_1, H_2, H_3\}$ as system parameters.

SKeyGen *$(\mathsf{param}_{\mathsf{EVA}}) \rightarrow (x_i, y_i)$:* On input system parameters $\mathsf{param}_{\mathsf{EVA}}$, to generate private and public keys for a cloud server (say i), pick a random number $x_i \xleftarrow{R} Z_p^*$ as the private key, and compute $y_i = g^{x_i}$ as the public key. The algorithm outputs (x_i, y_i).

UKeyGen$(\mathsf{param}_{\mathsf{EVA}}) \rightarrow (ssk, spk)$: On input system parameters $\mathsf{param}_{\mathsf{EVA}}$, generate a private signing key *ssk* and public verification key *spk* for the user. The algorithm outputs (ssk, spk).

B. Joint Key Generation

To invite $w \geq 2$ cloud servers to unify as a jointcloud, run algorithms below to generate a joint key $y = g^x$, where $x = x_1, \ldots, x_w$ and x_i denotes the private key of cloud server i.

JKeyGen$(\mathsf{param}_{\mathsf{EVA}}) \rightarrow (y, L_{\text{ring}})$: On input system parameters $\mathsf{param}_{\mathsf{EVA}}$, negotiate a joint key y for service as follows. w servers form a ring based on the idea in [17]. Record the ring membership in a list L_{ring}. Based on the sequence of the ring, the ith ring member relays g^{x_i} to the next member, meanwhile, it receives $g^{x_{i-1}}$ from the $(i-1)$th member and forward $g^{x_{i-1} \cdot x_i}$ to the next member as well. Denoting the ring-based computation as (RBC). After w circles, each member could derive join key $y = g^{x_1, \ldots, x_w}$. The algorithm outputs joint key and a list of ring membership (y, L_{ring}).

C. File Processing

The user runs algorithm $\mathsf{Enc\&TagGen}$ to preprocess file m and generate encrypted file c and metadata $\{u, k, \mathsf{CID}, \sigma, \text{aux}\}$ as follows.

Enc&TagGen$(\mathsf{param}_{\mathsf{EVA}}, m) \rightarrow (c, u, k, \mathsf{CID}, \sigma, aux)$: On input system parameters $\mathsf{param}_{\mathsf{EVA}}$, a customized identity CID [18], and a file $m \in \{0, 1\}^*$ proceed as follows.

1) Compute file master key $k_0 = H_2(m)$ and file tag $\sigma_0 = g^{k_0} \in G$. Divide file m into n blocks and s sectors such that each $m[i][j] \in Z_p$. Denote $m = \{m[i][j]\}_{1 \leq j \leq s, 1 \leq i \leq n}$
2) Choose a customized identity $\mathsf{CID} \in \{0, 1\}^*$. Compute $h = H_1(\mathsf{CID}) \in G$. For each $1 \leq i \leq n$, compute block key $k_i = g^{k_0} \cdot h^{H_2(m[i])} \in G$. Denote $k = \{k_i\}_{0 \leq i \leq n}$.
3) Encrypt each block as $c[i] = m[i] \oplus H_3(k_i)$ for $1 \leq i \leq n$, derive ciphertext file $c = \{c[i][j]\}_{1 \leq j \leq s, 1 \leq i \leq n}$.
4) Choose a random element $u \xleftarrow{R} G$. For each $1 \leq i \leq n$, compute block tag $\sigma_i = (k_i \cdot \prod_{j=1}^{s} u_j^{c[i][j]})^{k_0} \in G$. Compute auxiliary information $\text{aux}_i = \hat{e}(k_i, g^{k_0})$. Denote $\sigma = \{\sigma_i\}_{0 \leq i \leq n}$, $\text{aux} = \{\text{aux}_i\}_{1 \leq i \leq n}$.

The algorithm outputs $\{c, u, k, \mathsf{CID}, \sigma, \text{aux}\}$.

The user keeps $k = \{k_i\}_{0 \leq i \leq n}$ privately at local side for spot-check-based auditing [19].

D. Commit Jointcloud

To upload $f \geq 2$ distinct processed files and metadata $\{c_l, \text{meta}_l\}_{1 \leq l \leq f}$ to a jointcloud, where each file is associated with a master key $k_{0,l}$, the user designates a number of servers and requests outsourcing service as follows.

JoiServ.Chal$(\{k_{0,l}\}_{1 \leq l \leq f}) \rightarrow (jchal)$: On input f file master keys $\{k_{0,l}\}_{1 \leq l \leq f}$, where each $k_{0,l} = H_2(m_l)$ identifies a target file $m_l \in \{0, 1\}^*$ to be outsourced, compute $\varphi = g^{(\sum_{1 \leq l \leq f} k_{0,l} \mod p)}$. Denote L_{ring} as a list which records ring membership of w designated servers for service (as a jointcloud). The algorithm outputs $jchal = (\varphi, L_{\text{ring}})$ as a challenge for jointcloud service.

JoiServ.Prove$(jchal) \rightarrow (jprf)$: On input a challenge $jchal = (\varphi, L_{\text{ring}})$, w servers computes: $\delta = \varphi^x = \varphi^{\prod_{i \in [1,w]} x_i}$ as a jointcloud based on the idea of RBC as mentioned by algorithm $\mathsf{JKeyGen}$. The algorithm outputs a proof for joint service by $jprf = \{\delta\}$.

JoiServ.Verify$(jchal, jprf, \{k_{0,l}\}_{1 \leq l \leq f}, y) \rightarrow (0,1)$: On input challenge and proof *jchal*, *jprf* for jointcloud service, w file master keys $\{k_{0,l}\}_{1 \leq l \leq f}$, and a joint key y (derived from $\mathsf{JKeyGen}$), check whether: $\delta \stackrel{?}{=} y^{\sum_{1 \leq l \leq f} k_{0,l}}$. If yes, output 1; else, output 0.

E. Deduplication

Before outsourcing, a jointcloud runs PoW [16] protocol with the user to authenticate file ownership, it runs EqTest to check tag consistency.

EqTest$(\mathsf{param}_{\mathsf{EVA}}, c, \sigma, u) \rightarrow (0 \text{ or } 1)$: On input system parameter $\mathsf{param}_{\mathsf{EVA}}$, an encrypted file $c = \{c[i][j]\}_{1 \leq j \leq s, 1 \leq i \leq n}$, file master key $\sigma_0 = g^{k_0}$, and a set of tags $\{\sigma_i\}_{0 \leq i \leq n}$, check whether equation holds for each $1 \leq i \leq n$

$$\hat{e}(\sigma_i, g) = \text{aux}_i \cdot \hat{e}\left(\prod_{j=1}^{s} u_j^{c[i][j]}, \sigma_0\right).$$

If all hold, output 1; else, output 0.

PoW.Chal$() \rightarrow (dchal)$: No input, pick a set $Q_d \subseteq [1, n]$. Output a deduplication challenge $dchal = \{Q_d\}$.

PoW.Prove$(\mathsf{param}_{\mathsf{EVA}}, dchal, c, \mathsf{CID}, ssk) \rightarrow (dprf)$: On input system parameter $\mathsf{param}_{\mathsf{EVA}}$, deduplication challenge *dchal*, encrypted file $c = \{c[i][j]\}_{1 \leq j \leq s, 1 \leq i \leq n}$, customized identity $\mathsf{CID} \in \{0, 1\}^*$, and user's private signing key *ssk* proceed as follows. Compute $h = H_1(\mathsf{CID})$. Then, compute $node_i = g^{k_0} \cdot h^{H_2(c[i])}$ for each $i \in Q_d$ and use each $node_i$ as leaf node to construct a merkle hash tree (MHT) (we denote it as chameleon-based MHT also known as C-MHT). Compute the root of C-MHT as R'. Let $node = \{node_i\}_{i \in Q_d}$. Then, generate each sibling path Ω_i for the challenged ith node where $i \in Q_d$ to root. Denote $\Omega = \{\Omega_i\}_{i \in Q_d}$. Finally, compute and sign the derived root as $SIGN_{ssk}(R')$.

The algorithm outputs a deduplication proof as $dprf = \{\text{node}, \Omega, SIGN_{ssk}(R')\}$.

PoW.Verify$(\mathsf{param}_{\mathsf{EVA}}, dchal, dprf, spk) \rightarrow (0, 1; \text{ or } \perp)$: On input system parameter $\mathsf{param}_{\mathsf{EVA}}$, deduplication challenge *dchal*, deduplication proof *dprf*, and user's public verification key *spk* proceed as follows. Use *spk* to verify the validity of $SIGN_{ssk}(R)$. If pass, proceed; else, output $\perp$.Then, use each $node_i$ in t to construct a C-MHT and derive root R'. Check whether $R' = R$ holds. If yes, output 1; else, output 0. The algorithm output 0, 1, or $\perp$.

F. Auditing

Audit.Chal$(\mathsf{param}_{\mathsf{EVA}}) \rightarrow (achal)$: On input system parameter $\mathsf{param}_{\mathsf{EVA}}$, randomly pick a set $\subseteq [1, n]$. Choose $v_i \in Z_p$ for each $i \in Q_a$. Generate an auditing challenge as $\text{achal} = \{i, v_i\}_{i \in Q_a}$. Output achal.

Audit.Prove$(achal, c, \sigma) \rightarrow (aprf)$: On input auditing challenge achal, encrypted file $c = c[1]|| \cdots ||c[n]$, a set of tags $\sigma = \{\sigma_i\}_{1 \leq i \leq n}$, and compute $\mu_j = \sum_{i \in Q} v_i \cdot c[i][j] \in Z_p$ for each $1 \leq j \leq s$ and $\sigma = \prod_{i\in Q} \sigma_i^{v_i} \in G$. Finally, compute $z = \hat{e}(\sigma^x, g)$ between a ring L_{ring} of cloud servers such that $x = \prod_{i\in L_{\text{ring}}} x_i$ based on RCB as mentioned earlier. Output an auditing proof as $aprf = \{\mu, \sigma, z\}$.

***Audit.Verify**($\mathsf{param}_{\mathsf{EVA}}$, achal, aprf, k, y) $\rightarrow$ ($\perp$ 0 or 1):* On input system parameter $\mathsf{param}_{\mathsf{EVA}}$, auditing challenge achal, auditing proof $aprf = \{\mu, \sigma, z\}$, a set of block keys $k = \{k_i\}_{0\leq i\leq n}$, file tag τ, and a joint key y, first check whether equation holds: $z \overset{?}{=} \hat{e}(\sigma, y)$. If yes, proceed; else, output $\perp$ to indicate an error. Next, check whether equation holds

$$\hat{e}(\sigma, g) \overset{?}{=} \hat{e}\left(\prod_{i\in Q} {k_i}^{v_i} \cdot \prod_{j=1}^{s} u_j^{\mu_j}, \sigma_0\right).$$

If yes, output 1; else, output 0.

G. Dynamic Update

The user runs the update protocol with the server as described below; we only give an instance for block modification and omit block insertion and deletion due to space limitations. Meanwhile, we only consider the target file to be updated is privately owned by a user, thus, contradiction caused by dynamic update and deduplication is omitted in this article [14].

***Update.Chal**($\mathsf{param}_{\mathsf{EVA}}$, m_i, CID, k_0, y, σ_i $\rightarrow$ (uchal)):* On input system parameter $\mathsf{param}_{\mathsf{EVA}}$, a target block plaintext $m[i]$ (and $m[i]'$ if it refers to block modification), customized identity CID, file master key k_0, joint key of designated servers y, and block tag σ_i proceed as follows.

To modify the ith block from $c[i]$ to $c[i]'$, set $op = \mathcal{M}$. Compute $h = H_1(\mathsf{CID}) \in G$, block key $k_i = g^{k_0} \cdot h^{H_2(m[i])} \in G$, old block ciphertext $c[i] = m[i] \oplus H_3(k_i)$, new block ciphertext $c[i]' = m[i]' \oplus H_3(k_i)$, and new block tag $\sigma_i' = \left(k_i \cdot \prod_{j=1}^{s} u_j^{c[i][j]'}\right)^{k_0}$. Generate the trapdoor $r = (g^{k_0}, y^{k_0})$, where y denotes a joint key of designated servers. Set update challenge for block modification as $uchal = \{op, i, c_i, c_i', \mathsf{CID}, \sigma_i', r, \hbar\}$. Output $uchal$.

***Update.Prove**($\mathsf{param}_{\mathsf{EVA}}$, uchal, $\{x_i\}_{i\in L_{\text{ring}}}$, y) $\rightarrow$ (uprf):* On input system parameter $\mathsf{param}_{\mathsf{EVA}}$, an update challenge $uchal$, a set of private keys of designated servers $\{x_i\}_{i\in L_{\text{ring}}}$, and a joint key y proceed as follows.

Compute $h = H_1(\mathsf{CID}) \in G$. Compute new trapdoor r' based on the idea of RBC earlier mentioned

$$\begin{aligned} r' &= \left(g^{k_0'}, y^{k_0'}\right) \\ &= \left(g^{k_0} \cdot h^{\left[(H_2(c[i]) - H_2(c[i]'))\right]}, g^{k_0} \cdot h^{x\left[H_2(c[i]) - H_2(c[i]')\right]}\right) \end{aligned}$$

where $x = \prod_{i\in L_{\text{ring}}} x_i$. Replace old block tag σ_i with σ_i' (consistency check is omitted here), and old block ciphertext $c[i]$ with $c[i]'$. Generate node sibling path Ω_i for block c_i' (there is no need to compute merkle root in this case as our C-MHT treats each leaf node as chameleon hash where hash value remains unchanged via a found collision). Return an update proof for block insertion as $uprf = \{r'\}$. Output $uprf$.

***Update.Verify**($\mathsf{param}_{\mathsf{EVA}}$,uchal,uprf) $\rightarrow$ (0 or 1):* On input system parameter $\mathsf{param}_{\mathsf{EVA}}$, update challenge $uchal$, update proof $uprf$, and a joint key y proceed as follows. Parse $uprf = \{r'\}$, where $r' = (g^{k_0'}, y^{k_0'})$. Check whether $(g, g^{k_0'}, y, y^{k_0'})$ is a Diffie–Hellman tuple and whether equation $g^{k_0} \cdot h^{H_2(m[i])} = g^{k_0} \cdot h^{H_2(m[i]')}$ holds. If yes, output 1; else, output 0.

H. Bidding Commitment for Data Trade

We characterize data trade by bidding protocols where cloud server (or jointcloud) issues a base price based on which user will compete on bidding. The bidding is made through bitcoin and each user will receive corresponding bidding commitment and trapdoor (i.e., $\{\hbar, bprf\}$) as a receipt after successful transferring bitcoins to a designated account (as bid). Later, the jointcloud or any other third parties can check blockchain and verify by executing an algorithm $\mathsf{Bid.Chal}$ to find the highest bidder for an item in a period of time (bidding duration). Refer to Section V-D for security guarantee.

***Bid.Chal**($\mathsf{param}_{\mathsf{EVA}}$, pk_s, σ_0, $price_{base}$, y, t) $\rightarrow$ (bchal):* On input system parameter $\mathsf{param}_{\mathsf{EVA}}$, public key of seller pk_s, file tag of data to be sold $\sigma_0 = g^{k_0}$, base price $\text{price}_{\text{base}}$, and a joint key of designated servers y and a current time t, compute coefficients $\eta = H_1(\epsilon)$ and $\theta = H_2(\epsilon)$. Then, generate bidding information $\epsilon = pk_s||\text{price}_{\text{base}}||\tau$ which includes the identity of seller, base price of item (data to be sold), and file tag of item. Next, compute bidding trapdoor $\psi = (g^{k_0 t}, y^{k_0 t})$ and bidding commitment $\hbar = g^{k_0 t}\eta^{H_2(\epsilon)t} = (g^{k_0}\eta^{\theta})^t$. Output a bidding challenge $bchal = \{\epsilon, \hbar, \psi\}$.

***Bid.Prove**(ϵ', $\{x_i\}_{i\in L_{\text{ring}}}$, Δt) $\rightarrow$ (bprf):* On input a new bidding information $\epsilon' = pk_\pi||\text{price}_{\max}||\tau$ (where pk_π denotes the public key of the bidding winner and $\text{price}_{\max}$ denotes the highest price outbids others), a set of private keys of designated servers $\{x_i\}_{i\in L_{\text{ring}}}$ and a bidding duration Δt. Then, compute $\theta' = H_2(\epsilon')$. Finally, compute new bidding trapdoor based on RBC as mentioned earlier: $\psi' = (g^{k_0'(t+\Delta t)}, y^{k_0'(t+\Delta t)}) = (g^{k_0 t}\eta^{\theta(-\Delta t)}, y^{k_0 t}\eta^{x\theta(-\Delta t)})$. The algorithm outputs a proof of bid $bprf = \{\psi'\}$.

***Bid.Verify**(bchal, ϵ', bprf, Δt, t) $\rightarrow$ (0 or 1):* On input a bidding challenged $bchal$, new bidding information ϵ', an old bidding trapdoor ψ, a proof of bid $bprf = \{\psi'\}$, where ψ' indicates a new bidding trapdoor, and a bidding duration Δt and current time t, compute $\theta' = H_2(\epsilon')$ and $\eta = H_1(\epsilon)$. Check whether $g^{k_0 t}\eta^{\theta t} = g^{k_0'(t+\Delta t)}\eta^{\theta'(t+\Delta t)}$ holds. If yes, output 1; else, output 0.

I. File Retrieval

$\mathsf{Decryption}$*(c, k_0, σ) $\rightarrow$ (m):* On input system parameter $\mathsf{param}_{\mathsf{EVA}} = \{G, G^T, g, p, \hat{e}, u_1, \ldots, u_s, H_1, H_2, H_3\}$, an encrypted file $c = \{c[i][j]\}_{1\leq j\leq s, 1\leq i\leq n}$, a file master key k_0, and a set of block tag $\sigma = \{\sigma_i\}_{0\leq i\leq n}$ proceed as follows.

For each $1 \leq i \leq n$, retrieve block key by computing $k_i = {\sigma_i}^{{k_0}^{-1}} \cdot \prod_{j=1}^{s} u_j^{-c[i][j]} \in G$. Parse $c = c[1]||\cdots||c[n]$. For each $1 \leq i \leq n$, decrypt each block as $m[i] = c[i] \oplus H_3(k_i)$. Derive $m = \{m[i][j]\}_{1\leq j\leq s, 1\leq i\leq n}$. Output $m = \{m[i][j]\}_{1\leq j\leq s, 1\leq i\leq n}$.

V. Security Analysis of EVA

We give security analysis for each dominant definition as given in Section III-B. The security of joint key commitment in Section IV-D is obviously based on the intractability of the discrete logarithm problem (DLP) and is omitted due to space limitations.

A. Auditing Soundness

We briefly show how to construct an algorithm $\mathcal{B}$ which interacts with $\mathcal{A}$ to solve CDHP as below.

Given a CDHP instance (g, g^a, g^b), $\mathcal{A}$ first runs $\mathsf{param}_{\mathsf{EVA}} \leftarrow \mathsf{Setup}(\lambda)$ and achal $\leftarrow \mathsf{Audit.Chal}$. Then, $\mathcal{B}$ relays $(\mathsf{param}_{\mathsf{EVA}}, \text{achal})$ to $\mathcal{A}$.

$\mathcal{B}$ sets $\sigma_0 = g^a$ and denotes $k_0 = a$ as file master key. Then, for each $1 \le j \le s$, $\mathcal{B}$ samples $\beta_j, \gamma_j \xleftarrow{R} Z_p$, and computes $\mu_j = g^{\beta}(g^b)^{\gamma}$. Next, $\mathcal{B}$ samples $\delta_i \xleftarrow{R} Z_p$ for each $1 \le i \le n$. $\mathcal{B}$ returns answer to each query made by $\mathcal{A}$ on a customized identity $\mathsf{CID} \in \{0,1\}^*$ and message block $m[i] \in \{0,1\}^*$ as

$$H_1(\mathsf{CID}) = g^{\delta} / \left[g^{\sum_{j=1}^{s} \beta_j m[i][j]} \left(g^b\right)^{\sum_{j=1}^{s} \gamma_j m[i][j]} g^a \right]^{H_2(m[i])^{-1}}.$$

So, $\mathcal{B}$ can generate block tag of message block $m[i]$ correctly because: $\sigma_i = [\sigma_0 H_1(\mathsf{CID})^{H_2(m[i])} \cdot \sum_{j=1}^{s} u_j^{m[i][j]}]^a = (g^a)^{\delta_i}$. Suppose $aprf^*$ is a successful forgery on the given challenge achal, and $aprf$ is a valid proof, where $aprf \ne aprf^*$. We have the following equations:

$$\hat{e}(\sigma, g) = \hat{e}\left(\prod_{i \in Q_a} \left(k_i^{v_i}\right) \cdot \prod_{j=1}^{s} u_j^{\mu_j}, \sigma_0\right)$$
$$\hat{e}\left(\sigma^*, g\right) = \hat{e}\left(\prod_{i \in Q_a} \left(k_i^{v_i}\right) \cdot \prod_{j=1}^{s} u_j^{\mu_j^*}, \sigma_0\right).$$

As $aprf^* \ne aprf$, we can divide above equations and derive an answer to CDHP as

$$\left(\sigma^* \cdot \sigma \cdot \sigma_0^{-\sum_{j=1}^{s} \beta_j \Delta\mu_j}\right)^{\frac{1}{\sum_{j=1}^{s} \gamma_j \Delta\mu_j}} = g^{ab}.$$

B. Deduplication Soundness

To win the experiment $\mathrm{PoW}_{\mathsf{EVA}}^{\mathcal{A}}(\lambda)$, $\mathcal{A}$ will need to forge a proof $dprf*$ to pass *dedupverify*, this implies breaking soundness of MHT, collision resistance of chameleon hash, and unforgeability of signing scheme SIGN adopted. We focus on the former two security requirements as follows.

The soundness of MHT has been discussed by [16] based on the information theory. We can deduce $\mathcal{A}$'s probability to pass MHT verification by: $P(succ_{Q_d}) = 1 - \epsilon(1-p)^{|Q_d|}$. Furthermore, with proper choice on ϵ and p (denotes the success probability of guessing a random block), we can bound the above equation by $\lceil (\lambda ln2/\epsilon(1-p)) \rceil$, where λ denotes a security parameter we have chosen. More details can be found in [20].

The collision resistance of leaf node (characterized by $l_i = g^{k_0} \cdot h^{H_2(c[i])}$) can be reduced to the CDHP problem. Briefly, if there exists an efficient adversary (say $\mathcal{B}$) who can break the above collision resistance, we can therefore construct an algorithm $\mathcal{C}$ to solve CDHP by interacting with $\mathcal{B}$ as below. On given a CDHP instance (g, g^a, g^b), $\mathcal{C}$ sets $hk = g^a$ as hash key, $tk = a$ as trapdoor key, and $\hat{h} = g^b$. Then, $\mathcal{C}$ relays system parameter $<g, G, p, H_1, H_2>$ to $\mathcal{B}$. $\mathcal{B}$ can then query a forging oracle Forge controlled by $\mathcal{C}$ adaptively on a customized identity $\mathsf{CID} \in \{0,1\}^*$ and two different file blocks $c[i] \ne c[i]'$ for some i, then, $\mathcal{C}$ returns $r' = (g^{k_0}, y^{k_0}) = (g^{k_0} h^{H_1(c[i]) - H_1(c[i]')}, y^{k_0} h^{x(H_1(c[i]) - H_1(c[i]'))})$ for each distinct query where $h = H_1(\mathsf{CID})$. Finally, $\mathcal{B}$ outputs a forged collision on CID where the following equation holds: $g^{\alpha} \hat{h}^{H_1(c[i])} = g^{\alpha'} \hat{h}^{H_1(c[i]')}$ for some $c[i] \ne c[i]'$, where $\hat{h} = H_1(\mathsf{CID}*)$ and CID^* has never been queried previously. Thus, $\mathcal{C}$ can then extract a solution to CDHP from it by computing

$$\left(\frac{hk^{\alpha'}}{hk^{\alpha}}\right)^{H_1(c[i]) - H_1\left(c[i]'\right)^{-1}} = g^{ab}.$$

Refer to [17] for more details.

Based on the above, our EVA is deduplication sound if MHT [16] is sound, chameleon hash [17] is collision resistant, and signature SIGN adopted is unforgeable.

C. Update Soundness

Our EVA satisfies update soundness if no adversary $\mathcal{A}$ can forge an update proof $uprf*$ to pass $\mathsf{Update.Verify}$ as defined by experiment $\mathrm{Upd}_{\mathsf{EVA}}^{\mathcal{A}}(\lambda)$. Concretely, $uprf*$ is parsed by $\{r'\}$ and $\{\Omega, \phi\}$ for block modification and insertion, deletion, respectively. A successful forgery on the proof of block modification (i.e., $op = \mathcal{M}$) implies finding an r'' such that equation $g^{k_0} \cdot h^{H_2(c[i])} = g^{k_0} \cdot h^{H_2(c[i]')}$ holds. Suppose r' is the valid result, if $r'' = r'$, this indicates collision resistance as we briefly discussed earlier (Section V-B). If $r'' \ne r'$, this indicates breaking an even stronger notion called "uniqueness" (as identified by [21]), where two collisions were found (r' and r'') to hold the following equations: $g^{k_0} \cdot h^{H_2(c[i])} = g^{k_0'} \cdot h^{H_2(c[i])'}$ and $g^{k_0} \cdot h^{H_2(c[i])} = g^{k_0''} \cdot h^{H_2(c[i]'')}$. This also implies breaking the collision resistance as discussed in Section V-B.

D. Bidding Soundness

Our EVA satisfies bidding soundness if no efficient adversary $\mathcal{A}$ can forge a proof of bid $bprf^*$ to pass $\mathsf{Bid.Chal}$ as defined in experiment $\mathrm{Bid}_{\mathsf{EVA}}^{\mathcal{A}}(\lambda)$.

Concretely, parse $bprf^* = \psi^*$. Suppose $bprf = \psi'$ is the valid proof for bidding challenge *bchal*. We have: $g^{k_0 t} \eta^{\theta t} = g^{k_0'(t+\Delta t)} \eta^{\theta'(t+\Delta t)}$ and $g^{k_0 t} \eta^{\theta t} = g^{k_0^*(t+\Delta t)} \eta^{\theta^*(t+\Delta t)}$. Thus, we can derive another collision such that $g^{k_0'(t+\Delta t)} \eta^{\theta'(t+\Delta t)} = g^{k_0^*(t+\Delta t)} \eta^{\theta^*(t+\Delta t)}$. This implies breaking uniqueness of chameleon-hash-based commitment (as identified by [21]) which is a stronger notion than collision resistance. Based on the earlier discussions in Section V-C, we can reduce our bidding soundness to the security of collision resistance (and uniqueness). Since we have reduced these securities to the intractability of CDHP, analogically, we can reduce our bidding soundness to intractability of CDHP. Due to space limit, details are omitted.

VI. Performance Evaluations

In this section, we give comprehensive analysis on each function of our proposed EVA scheme. For abbreviations of cost, we denote T_m as group multiplication; T_e as group exponentiation; T_i as group inversion; T_p as bilinear pairing operation; T_h as hashing operation of SHA-256 (if it is not negligible in the cost); T_{sig} as generation of BLS signature [22]; T_{ver} as verification of BLS signature [22]; and T_{mht} as constructing an MHT.

TABLE I
COMPLEXITY OF FILE PROCESSING

Algorithm	Enc&TagGen	Decryption
Our EVA	$(1+2n)T_m+$ $3nT_e+nT_p$	nT_m+2nT_i $+2nT_e$
BL-MLE [12]	nT_m+nT_p $+(2n+1)T_e$	nT_m+2nT_i $+2nT_e$

Suppose each file m is partitioned into n blocks, while each block is further partitioned into s sectors. Each sector $c[i][j]$ is an element of group Z_p.

TABLE II
OFFLINE COST AT USER-SIDE

Algorithms	Number of Blocks per File (File size)			
	128 (4 KB)	512 (16 KB)	2048 (64 KB)	8192 (256 KB)
Enc&TagGen	0.693 s	2.937 s	9.728 s	36.188 s
Decryption	0.611 s	2.163 s	8.538 s	34.792 s

Size of each sector (i.e., $m[i][j] \in Z_p$ for some i and j) is 256 bit.

For simulations, we code simulations on C language and implement PBC-0.5.13 for cryptographical operations. We select super supersingular curve $y^2 = x^3+x$ with an embedding degree of 2. Our experiments are carried out on a laptop with 3.5-GHz 4-cores CPU, 8-GB RAM, and 256 SSD (with much faster loading speed than conventional hard disc drive). The operating system is 32-bit Windows 7 SP1. We use OpenSSL as a means of secure communication. Bandwidth environments are 10 and 50 Mb/s (will be used for comparison). Each result is derived from a mean of ten trials. Other parameters will be specified right after when used.

A. Performance of File Processing

We compare our EVA with a relevant work [12] on processing complexity in Table I. Two works both adopt message-locked encryption (MLE) [15] to process file to enable deduplication, therefore, identical file from different use will lead to same encrypted file. We omit measuring cost of symmetric encryption as it is generally fast. As is show in Table I, our EVA is less efficient as our metadata involves more computations to support block dynamics. Specifically, each block key k_i in our EVA is computed by $g^{k_0}h^{H_2(m_i)}$ (Pedersen commitment [23]) instead of a mere hash in BL-MLE [12].

Additionally, we note that a portion of block keys $\{k_i\}_{1\leq i\leq n}$ should be kept at the auditor side for spot-check-based auditing in our EVA. The number of block keys $\{k_i\}$ to be kept at user-side depends on what type of auditing to be achieved (for static check, 460 block keys will suffice to guarantee over 99% detection rate [10]; for dynamic check, a whole set of blocks should be kept, it depends on the number of blocks n by which a file m is partitioned).

For quantitative analysis, we conduct several experiments to mainly test computational cost and show results in Table II. The statics suggest that our EVA is as efficient as BL-MLE. However, the performance of our EVA is not comparable to which of any symmetric encryption. But we consider it as a one-time cost, as it will be compensated by savings brought by deduplication and update later.

TABLE III
COMPLEXITY OF AUDITING

Algorithm	Audit.Prove	Audit.Verify
Our EVA	$(\lvert Q_a\rvert+w)T_e+$ $(2\lvert Q_a\rvert-1)T_m$	$(\lvert Q_a\rvert+1)T_e+$ $(\lvert Q_a\rvert-1)T_m+2T_p$
DPAF [24]	$\lvert Q_a\rvert T_e+$ $(2\lvert Q_a\rvert-1)T_m$	$(\lvert Q_a\rvert+1)T_e+2T_p$ $+\lvert Q_a\rvert T_m$
ID-RDIC [25]	$(\lvert Q_a\rvert+1)T_e+T_i+$ $(3\lvert Q_a\rvert-2)T_m+2T_p$	$(2\lvert Q_a\rvert+4)T_e+$ $(\lvert Q_a\rvert)T_p$
Compact PoR [26]	$\lvert Q_a\rvert T_e+$ $(2\lvert Q_a\rvert-1)T_m$	$(\lvert Q_a\rvert+1)T_e$ $\lvert Q_a\rvert T_m+2T_p$
CPVPA [27]	$\lvert Q_a\rvert T_e+$ $2\lvert Q_a\rvert T_m$	$(3\lvert Q_a\rvert+2)T_e+$ $5\lvert Q_a\rvert+2)T_m+4T_p$

Denote $|Q_a|$ as number of challenged blocks, w as number of servers to form as a jointcloud.

TABLE IV
COMPLEXITY OF RELEVANT PoW

Algorithm	PoW.Prove (User-side)	PoW.Prove (Server-side)
Our EVA (C-PoW)	$O(\lvert m\rvert)\cdot(T_h+T_e+T_m)$	$O(\lvert m\rvert)\cdot T_h+O(1)$
b-PoW [16]	$O(\lvert m\rvert)\cdot T_h$	$O(\lvert m\rvert)\cdot T_h+O(1)$
s-PoW [20]	$O(\lvert m\rvert)\cdot T_h$	$0(\lvert Q_d\rvert\cdot\kappa)\cdot T_{PRF}$

Denote $|m|$ as size of message m, T_{PRF} as pseudo-random function operation, κ as security parameter, $|Q_d|$ as number of challenged blocks.

B. Performance of Auditing

We compare our EVA scheme with relevant schemes [24]–[27] on complexity of auditing in Table III. In comparison with peer works, our EVA scheme is as efficient as other schemes in verification. However, we additionally involve computing $z = \hat{e}(\sigma^x, g)$ in order to commit jointcloud service. Specifically, the computation of σ^x is based on a ring sequence, where each cloud server contributes their private key x_i to generate $x = \prod_{i\in L_{\text{ring}}} x_i$ [17]. To evaluate the performance of our auditing scheme under different factors, we vary challenged blocks $|Q_a|$ from 100 to 1000, and adopt different w (number of servers as a jointcloud) by 10, 20, 50, and 100, respectively, for analysis. As is shown in Table III and Figs. 2 and 3, the cost of our EVA in generating an auditing proof on each server scales fine with variables. Need to point out, if w surges radically, the cost on each server to generate auditing proof will increase dramatically as well. Therefore, it is needed to set w reasonably to optimize auditing performance for massive files.

What is more, our scheme is as efficient as work of [26] in verification where the scheme of [26] is commonly used as a baseline for comparison.

C. Performance of Deduplication

Our EVA adopts the concept of PoW [16] to authenticate file structure for deduplication, we denote it as C-PoW. We compare our scheme with relevant works regarding complexity in Table IV. As the complexity of MHT is linear with the size of file and way to partition the file into blocks, we therefore only give complexity degree. As is shown in Table IV, our C-PoW is less efficient than standard PoW (b-PoW) as we treat each leaf node more carefully by further computing a chameleon hash value (Pedersen commitment [23]). This provides us with a basis to commit dynamic update without

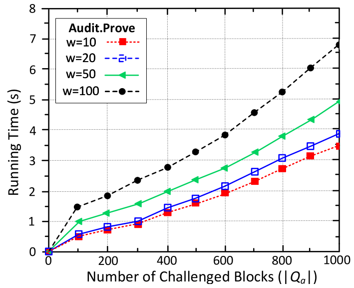


Fig. 2. Auditing cost of each server.

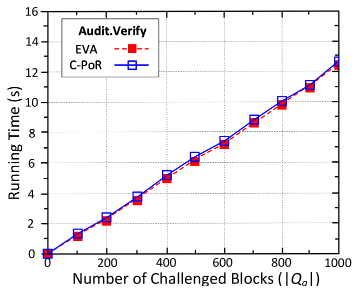


Fig. 3. Auditing cost at user-side.

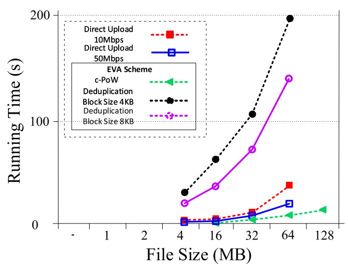


Fig. 4. Small file deduplication.

varying hash value [17], but introduces more complexities as a result.

To quantify the performance of small-file deduplication (from 2 to 256 KB), we use SHA-256 to authenticate file structure based on MHT [16]. We denote the result as the cost of C-PoW. Meanwhile, we also accumulate the result of c-PoW with file processing (as evaluated in Section VI-A), and denote it as cost of deduplication. We also apply direct upload (under 10- and 50-Mb/s bandwidth, respectively) as baselines. As shown in Fig. 4, although our C-PoW is efficient, our EVA is still poor in performing deduplication on small file. However, we did witness the fact that larger block costs less than smaller ones because small-sized block results in less leaf nodes to build an MHT. Consequently, our deduplication is meaningless for small files since it is slower than direct upload.

To test deduplication on large files, we randomly sample files from (256–2048 MB) and divide them with different block sizes. For instance, to partition a 256-MB large file by 16-KB block length, there will be 16 384 blocks (where each block contains 512 sectors, each sector is 256 bits large). Meanwhile, direct uploads are also used as baselines for comparison. As is shown in Fig. 5, our EVA is generally more

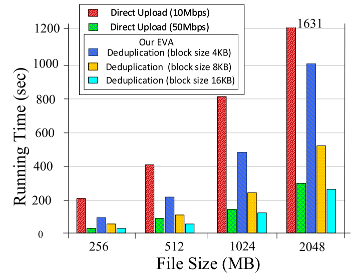


Fig. 5. Large file deduplication.

TABLE V
COMPLEXITY OF UPDATE (MODIFICATION)

Algorithm	Update.Prove	Update.Verify
Our EVA	$2wT_e + T_{mht}$ $+2T_m$	$2T_e + 2T_m$ $+1T_{mht}$
EPAD [28]	$T_e + T_m$ $+2T_p + 1T_{ver}$	$(\lvert Q_u\rvert + 3)T_e + 4T_p$ $\lvert Q_u\rvert T_m + T_{ver}$

Denote $|Q_u|$ as number of challenged blocks, w as number of servers as jointcloud. We use T_m to denote multiplication in either G or Z_p for ease of comparison.

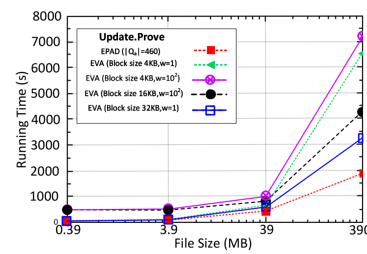


Fig. 6. Update cost at server-side.

efficient than 10-Mb/s bandwidth direct upload. To win 50-Mb/s bandwidth, it is suggested to break the file into larger blocks for deduplication so that fewer leaf nodes are generated to construct C-MHT.

D. Performance of Dynamic Update

We compare the complexity of dynamic update (block modification) for relevant schemes in Table V. As is shown in Table V, our scheme is affected by the number of designated servers w in generating update proof. To verify update proof, our EVA scheme only requires authenticating root of C-PoW for entire file (since each leaf node is unchanged due to character of chameleon hash [17]), whereas EPAD [28] requires performing a spot check on updated file (and it is affected by the number of challenged nodes). For quantitative analysis, we conduct experiments under various parameters (number of servers w and block size) for algorithm Update.Prove and Update.Verify, respectively. We incrementally set file size (from 0.39 to 390 MB, where 0.39 MB denotes the size of the file consists of 100 4-KB blocks, analogically, 390 MB for 10^5 4-KB blocks). We also include w as a factor for comparison. As is shown in Fig. 6, the cost of dynamic update at the server side is dominated by block size, i.e., in the smaller block, more time is required to generate an update proof. In addition, as it is shown in Fig. 7, the cost of verifying an update proof is simply dominated by the size of the file (where we assume

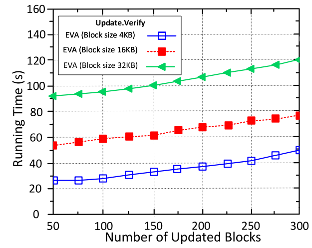


Fig. 7. Update cost at user side.

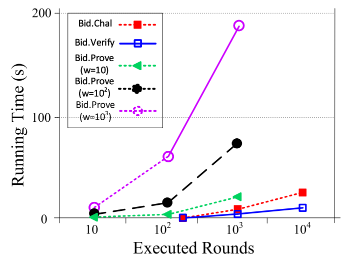


Fig. 8. Cost of bidding commitment.

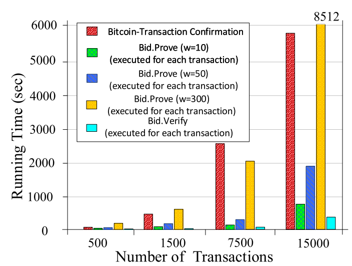


Fig. 9. Transaction confirmation cost.

each file consists of 50–300 blocks where block length is a variable). Therefore, the cost update verification in our EVA will gradually increase with file length if the block length is fixed.

E. Performance of Bidding Commitment

We list the complexity of our bidding commitment in Table VI. As is shown in Table VI, the cost is constant and negligible for each transaction except for the cost of algorithm Bid.Prove which is linearly factored by w. To evaluate performance, we establish a blockchain and make numerous transactions to a designated account for bidding. This allows us to simulate the bidding scenario with ease since bitcoin [8] and other cryptocurrencies are widely used nowadays. We use Geth 1.8.23 to simulate bitcoin transfer and set mining difficulty to very low since there is only one device for use. We incrementally set 500, 1500, 7500, and 15 000 as checkpoints and calculate the time for executing corresponding algorithms to derive an upper bound for evaluation. Suppose algorithm Bid.Prove is executed for each transaction, we also use a number of servers w as a variable for evaluation. As is shown in Fig. 8, when the number of servers w is small, the cost of executing Bid.Prove (at server-side) is negligible. In addition, the performance of algorithms Bid.Chal and Bid.Verify is fine.

In Fig. 9, we suggest that it is needed to keep the gap between transaction confirmation cost and cost of Bid.Prove as wide as possible; otherwise, there will be not enough time to process bidding information before duration time Δt ends. In other words, we should select a proper w to optimize the performance of block-based bidding commitment so that the server can process as much as bidding actions if given a very short bidding duration Δt (such as live auction).

TABLE VI
COMPLEXITY OF BIDDING COMMITMENT

Bid.Chal	Bid.Prove	Bid.Verify
$4T_e + 3T_m$	$2(w+1)T_e + 3T_m$	$2T_e + 4T_m$

Denote w as number of servers as a jointcloud.

TABLE VII
REVIEW OF DATA AUDITING SCHEME

Scheme	Public Auditing	Dynamics	ID-based
PDP [10]	√	×	×
PoR [19]	×	×	×
Compact PoR [26]	√	×	×
DPDP [30]	×	√	×
MuR-DPA [31]	√	√	×
EPAP [35]	√	√	×
Panda [36]	√	×	×

Based on the above analysis, we conclude that our EVA suffices to support multiple data services efficiently for large files across multiple servers meanwhile trade these data with cryptocurrencies efficiently for IoT use.

VII. RELATED WORK

A. Data Auditing

Cloud auditing enables the user to check the validity of data stored on the cloud server without retrieving it. Ateniese *et al.* [10] proposed the first and formal notion of PDP. Juels and Kaliski [19] proposed the notion of proof-of-retrievability (PoR) to ensure the retrievability of data. Later on, this topic fast expands to identity-based infrastructure [25], data privacy [25], public verifiability [27], [29], data dynamics [28], etc. Specifically, data dynamics refers to availability to change data structure without compromising auditing [30], [31].

Note that the auditing scheme does not necessarily require encryption to be applied on data. However, if it does, it is suggested to encapsulate the decryption key in block tag for the sake of decryption (as suggested by Chen *et al.* [12]).

We compare some related schemes in Table VII, in terms of public auditing, dynamics, and ID-based properties. As it is Table VII, while the majority of auditing schemes support public auditing, less than half of them support dynamics.

B. Message-Locked Encryption

To achieve confidentiality, current outsourced data are generally encrypted at a remote server. Generally, symmetric encryption is applied on massive large files due to its fast performance. Another advantage of symmetric encryption is to enable encrypted deduplication such that identical files can be deleted for storage savings while data confidentiality is guaranteed.

TABLE VIII
REVIEW OF DATAMARKET-RELATED WORKS

Work	Collection	Protection	Analytics	Pricing	Trading
Our EVA	√	√	×	√	√
[37]	√	√	√	√	√
[38]	×	×	×	√	√
[39]	×	×	√	√	×
[40]	√	×	×	×	×
[41]	√	×	×	√	√
[42]	×	×	×	√	×

Douceur *et al.* [32] first proposed the notion of convergent encryption (CE) to practically encrypt a file by hash-as-a-key for deduplication. To formally exploit what security can be achieved, Bellare *et al.* [15] proposed MLE as an answer. Chen *et al.* extended MLE by proposing BL-MLE [12] to further partition file into blocks and encapsulating metadata into one set for fine-grained savings. Later, the notions of MLE-2 [33] and I-MLE [34] were proposed as primitives to seek stronger security.

However, encryption and data dynamics are fundamentally contradicted with each other, as the former is usually used for static data archive while the latter deals with dynamic storage and data structure [13].

C. Datamarket

Datamarket is a concept which seeks to commercialize data and data-driven services [43]. According to [37] survey, data market covers the following aspects: data collection, protection, analytics, pricing, and trading of data. We review some related works [37]–[40] and categorize them by associated aspects in Table VIII. As is shown in Table VIII, our proposed EVA scheme creates a concrete design for most aspects of datamarket except data analytics. Despite extensive survey conducted by Liang *et al.* [37], majority of related work focus on specific one or two aspects, such as pricing and trading proposed in [38] and [41], and data collection proposed in [40] and [41].

Particularly, our EVA scheme captures data collection feature by employing jointcloud proposed in [2] based on data outsourcing proposed in [4], data protection feature by MLE proposed in [15], and data pricing and trading by bidding-based commitment and deduplication (as briefly discussed in Section V-D). In short, we allow the same set of system parameters which was generated for data auditing to be utilized for the above functions to drive datamarket.

VIII. CONCLUSION

In this article, we proposed a scheme called EVA to drive IoT-based datamarket in jointcloud by enabling data auditing, encrypted deduplication, and dynamic update simultaneously. Additionally, we proposed the bidding commitment based on blockchain as a financial incentive to encourage data trade. We provided a security analysis on each proposed protocol and conducted comprehensive experiments on the proposed scheme. The evidence showed that our EVA scheme is generally efficient for large files in practice and if proper parameters are chosen.

REFERENCES

[1] L. Atzori, A. Iera, and G. Morabito, “The Internet of Things: A survey,” *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, 2010.
[2] J. Gantz and D. Reinsel, “The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east,” *IDC iView IDC Anal. Future*, pp. 1–16, 2012.
[3] D.-G. Cao, B. An, P.-C. Shi, and H.-M. Wang, “Providing virtual cloud for special purposes on demand in jointcloud computing environment,” *J. Comput. Sci. Technol.*, vol. 32, no. 2, pp. 211–218, 2017.
[4] H.-L. Truong and S. Dustdar, “Principles for engineering IoT cloud systems,” *IEEE Cloud Comput.*, vol. 2, no. 2, pp. 68–76, Mar./Apr. 2015.
[5] D. Puschmann, P. Barnaghi, and R. Tafazolli, “Adaptive clustering for dynamic IoT data streams,” *IEEE Internet Things J.*, vol. 4, no. 1, pp. 64–74, Feb. 2017.
[6] A. Bröring *et al.*, “Enabling IoT ecosystems through platform interoperability,” *IEEE Softw.*, vol. 34, no. 1, pp. 54–61, Jan./Feb. 2017.
[7] K. Mišura and M. Žagar, “Data marketplace for Internet of Things,” in *Proc. IEEE Int. Conf. Smart Syst. Technol. (SST)*, 2016, pp. 255–260.
[8] S. Nakamoto *et al.*, *Bitcoin: A Peer-to-Peer Electronic Cash System*, 2008. [Online]. Available: https://static.coinpaprika.com/storage/cdn/whitepapers/215.pdf
[9] P. Missier, S. Bajoudah, A. Capossele, A. Gaglione, and M. Nati, “Mind my value: A decentralized infrastructure for fair and trusted IoT data trading,” in *Proc. ACM 7th Int. Conf. Internet Things*, Linz, Austria, 2017, p. 15.
[10] G. Ateniese *et al.*, “Provable data possession at untrusted stores,” in *Proc. ACM 14th ACM Conf. Comput. Commun. Security*, Alexandria, VA, USA, 2007, pp. 598–609.
[11] J. Yuan and S. Yu, “Secure and constant cost public cloud storage auditing with deduplication,” in *Proc. IEEE Conf. Commun. Netw. Security (CNS)*, 2013, pp. 145–153.
[12] R. Chen, Y. Mu, G. Yang, and F. Guo, “BL-MLE: Block-level message-locked encryption for secure large file deduplication,” *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 12, pp. 2643–2652, Dec. 2015.
[13] K. Huang, X. Zhang, X. Wang, X. Du, and R. Zhang, “EBD-MLE: Enabling block dynamics under BL-MLE for ubiquitous data,” in *Proc. IEEE Int. Symp. Parallel Distrib. Process. Appl. IEEE Int. Conf. Ubiquitous Comput. Commun. (ISPA/IUCC)*, Guangzhou, China, 2017, pp. 1281–1288.
[14] K. Huang, X.-S. Zhang, and X.-F. Wang, “Block-level message-locked encryption with polynomial commitment for IoT data,” *J. Inf. Sci. Eng.*, vol. 33, no. 4, pp. 891–905, 2017.
[15] M. Bellare, S. Keelveedhi, and T. Ristenpart, “Message-locked encryption and secure deduplication,” in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.*, 2013, pp. 296–312.
[16] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg, “Proofs of ownership in remote storage systems,” in *Proc. 18th ACM Conf. Comput. Commun. Security*, Chicago, IL, USA, 2011, pp. 491–500.
[17] K. Huang *et al.*, “Building redactable consortium blockchain for industrial Internet-of-Things,” *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3670–3679, Jun. 2019.
[18] G. Ateniese and B. de Medeiros, “Identity-based chameleon hash and applications,” in *Proc. Int. Conf. Financ. Cryptography*, 2004, pp. 164–180.
[19] A. Juels and B. S. Kaliski, Jr., “Pors: Proofs of retrievability for large files,” in *Proc. 14th ACM Conf. Comput. Commun. Security*, Alexandria, VA, USA, 2007, pp. 584–597.
[20] R. Di Pietro and A. Sorniotti, “Boosting efficiency and security in proof of ownership for deduplication,” in *Proc. 7th ACM Symp. Inf. Comput. Commun. Security*, Seoul, South Korea, 2012, pp. 81–82.
[21] J. Camenisch, D. Derler, S. Krenn, H. C. Pöhls, K. Samelin, and D. Slamanig, “Chameleon-hashes with ephemeral trapdoors,” in *Proc. 20th IACR Int. Conf. Public Key Cryptography II*, 2017, pp. 152–182.
[22] D. Boneh, B. Lynn, and H. Shacham, “Short signatures from the weil pairing,” in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Security*, 2001, pp. 514–532.
[23] T. P. Pedersen, “Non-interactive and information-theoretic secure verifiable secret sharing,” in *Proc. Annu. Int. Cryptol. Conf.*, 1991, pp. 129–140.
[24] H. Jin, H. Jiang, and K. Zhou, “Dynamic and public auditing with fair arbitration for cloud data,” *IEEE Trans. Cloud Comput.*, vol. 6, no. 3, pp. 680–693, Jul./Sep. 2018.
[25] Y. Yu *et al.*, “Identity-based remote data integrity checking with perfect data privacy preserving for cloud storage,” *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 4, pp. 767–778, Apr. 2017.

[26] H. Shacham and B. Waters, "Compact proofs of retrievability," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Security*, 2008, pp. 90–107.
[27] Y. Zhang, C. Xu, X. Lin, and X. S. Shen, "Blockchain-based public integrity verification for cloud storage against procrastinating auditors," *IEEE Trans. Cloud Comput.*, to be published.
[28] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 5, pp. 847–859, May 2011.
[29] A. Fu, S. Yu, Y. Zhang, H. Wang, and C. Huang, "NPP: A new privacy-aware public auditing scheme for cloud data sharing with group users," *IEEE Trans. Big Data*, to be published.
[30] C. C. Erway, A. Küpçü, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," *ACM Trans. Inf. Syst. Security*, vol. 17, no. 4, p. 15, 2015.
[31] C. Liu, R. Ranjan, C. Yang, X. Zhang, L. Wang, and J. Chen, "MuR-DPA: Top-down levelled multi-replica Merkle hash tree based secure public auditing for dynamic big data storage on cloud," *IEEE Trans. Comput.*, vol. 64, no. 9, pp. 2609–2622, Sep. 2015.
[32] J. R. Douceur, A. Adya, W. J. Bolosky, P. Simon, and M. Theimer, "Reclaiming space from duplicate files in a serverless distributed file system," in *Proc. IEEE 22nd Int. Conf. Distrib. Comput. Syst.*, 2002, pp. 617–624.
[33] M. Abadi, D. Boneh, I. Mironov, A. Raghunathan, and G. Segev, "Message-locked encryption for lock-dependent messages," in *Proc. Annu. Cryptol. Conf.*, 2013, pp. 374–391.
[34] M. Bellare and S. Keelveedhi, "Interactive message-locked encryption and secure deduplication," in *Proc. IACR Int. Workshop Public Key Cryptography*, 2015, pp. 516–538.
[35] J. Shen, J. Shen, X. Chen, X. Huang, and W. Susilo, "An efficient public auditing protocol with novel dynamic structure for cloud data," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 10, pp. 2402–2415, Oct. 2017.
[36] B. Wang, B. Li, and H. Li, "Panda: Public auditing for shared data with efficient user revocation in the cloud," *IEEE Trans. Services Comput.*, vol. 8, no. 1, pp. 92–106, Jan.–Feb. 2015.
[37] F. Liang, W. Yu, D. An, Q. Yang, X. Fu, and W. Zhao, "A survey on big data market: Pricing, trading and protection," *IEEE Access*, vol. 6, pp. 15132–15154, 2018.
[38] Z. Zheng, Y. Peng, F. Wu, S. Tang, and G. Chen, "An online pricing mechanism for mobile crowdsensing data markets," in *Proc. 18th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2017, p. 26.
[39] Y.-C. Tsai *et al.*, "Time-dependent smart data pricing based on machine learning," in *Proc. Can. Conf. Artif. Intell.*, 2017, pp. 103–108.
[40] F. Schomm, F. Stahl, and G. Vossen, "Marketplaces for data: An initial survey," *ACM SIGMOD Rec.*, vol. 42, no. 1, pp. 15–26, 2013.
[41] C. Williams, M. Suzuki, G. Klenske, K. Graham, and J. Corsi, "Product data file for online marketplace sales channels," U.S. Patent Appl. 10/794 769, Sep. 8, 2005.
[42] C. Li and G. Miklau, "Pricing aggregate queries in a data marketplace," in *Proc. WebDB*, 2012, pp. 19–24.
[43] M. Balazinska, B. Howe, and D. Suciu, "Data markets in the cloud: An opportunity for the database community," *Proc. VLDB Endow.*, vol. 4, no. 12, pp. 1482–1485, 2011.

Ke Huang (S'19) received the M.S. degree from the University of Electronic Science and Technology of China, Chengdu, China, where he is currently pursuing the Ph.D. degree with the College of Computer Science and Engineering.

He is a visiting student with the University of Wollongong, Wollongong, NSW, Australia, from 2017 to 2019. His current research interests include blockchain and Internet of Things.

Xiaosong Zhang received the M.S. and Ph.D. degrees in computer science from the University of Electronic Science and Technology of China, Chengdu, China, in 1999 and 2011, respectively.

He is currently a Professor with the University of Electronic Science and Technology of China. He is the Cheung Kong Scholar Distinguished Professor. His current research interests includes blockchain, big data security, and AI security.

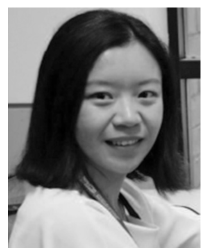
Yi Mu (SM'00) received the Ph.D. degree from the Australian National University, Canberra, ACT, Australia, in 1994.

In 2018, he was a Professor of computer science with the University of Wollongong, Wollongong, NSW, Australia. He is currently a Professor with the College of Mathematics and Informatics, Fujian Normal University, Fuzhou, China. His current research interests include blockchain, cybersecurity, and cryptography.

Prof. Mu was the Editor-in-Chief of the *International Journal of Applied Cryptography* and has served as an Associate Editor for several other international journals.

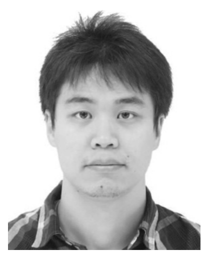
Fatemeh Rezaeibagha received the bachelor's degree in information technology engineering from Islamic Azad University, Tehran, Iran, in 2009, the M.S. degree in information security from the Luleå University of Technology, Luleå, Sweden, in 2013, and the Ph.D. degree in computer science from the University of Wollongong, Wollongong, NSW, Australia, in 2017.

She was an Associate Research Fellow with the University of Wollongong, in 2017. She is a Lecturer of cyber security with Murdoch University, Perth, WA, Australia. Her current research interests include cryptography, blockchain, and cybersecurity.



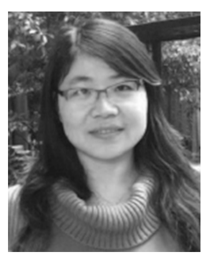
Xiaofen Wang received the M.S. and Ph.D. degrees in cryptography from Xidian University, Xi'an, China, in 2006 and 2009, respectively.

She is currently an Associate Professor with the College of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China. Her current research interests include cryptography and cloud computing.



Jingwei Li received the Ph.D. degree in computer application technology from Nankai University, Tianjin, China, in 2014, and the B.S. degree in information and computing science from Hebei University of Technology, Tianjin, in 2009.

Since 2016, he has been an Associate Professor with the University of Electronic Science and Technology of China, Chengdu, China. His current research interests include applied cryptography, and cloud security and secure deduplication.



Qi Xia received the B.S., M.S., and Ph.D. degrees in computer science from the University of Electronic Science and Technology of China, Chengdu, China, in 2002, 2006, and 2010, respectively.

She was a Visiting Scholar with the University of Pennsylvania, Philadelphia, PA, USA, from 2013 to 2014. She has published over 20 papers. She is the Vice Dean of the Center for Cyber Security, University of Electronic Science and Technology of China, where she is currently an Associate Professor. She is the PI of the National Key Research and Development Program of China in Cyber Security.

Jing Qin received the B.S. degree from Information Engineering University, Zhengzhou, China, in 1982, and the Ph.D. degree from the School of Mathematics, Shandong University, Jinan, China, in 2004.

She is currently a Professor with the School of Mathematics, Shandong University. She has coauthored two books and has published about 30 professional research papers. Her current research interests include information security, and design and analysis of security about cryptologic protocols.

Prof. Qin is a Senior Member of Chinese Association for Cryptologic Research and China Computer Federation.