


Article

Astrape: An Efficient Concurrent Cloud Attestation with Ciphertext-Policy Attribute-Based Encryption

Haihe Ba ^{1,†} , Huaizhe Zhou ^{1,†}, Songzhu Mei ^{1,2,*}, Huidong Qiao ^{1,3}, Tie Hong ¹, Zhiying Wang ^{1,*} and Jiangchun Ren ¹

¹ College of Computer, National University of Defense Technology, Changsha 410073, China; haiheba@nudt.edu.cn (H.B.); huaizhezhou@nudt.edu.cn (H.Z.); qiaohuidong13@nudt.edu.cn (H.Q.); tiehong@nudt.edu.cn (T.H.); jcren@nudt.edu.cn (J.R.)

² Science and Technology on Parallel and Distributed Laboratory, National University of Defense Technology, Changsha 410073, China

³ College of Computer and Communication, Hunan Institute of Engineering, Xiangtan 411100, China

* Correspondence: sz.mei@nudt.edu.cn (S.M.); zywang@nudt.edu.cn (Z.W.)

† These authors contributed equally to this paper.

Received: 28 August 2018; Accepted: 19 September 2018; Published: 21 September 2018



Abstract: Cloud computing emerges as a change in the business paradigm that offers pay-as-you-go computing capability and brings enormous benefits, but there are numerous organizations showing hesitation for the adoption of cloud computing due to security concerns. Remote attestation has been proven to boost confidence in clouds to guarantee hosted cloud applications' integrity. However, the state-of-the-art attestation schemes do not fit that multiple requesters raise their challenges simultaneously, thereby leading to larger performance overheads on the attester side. To address that, we propose an efficient and trustworthy concurrent attestation architecture under multi-requester scenarios, Astrape, to improve efficiency in the integrity and confidentiality protection aspects to generate an unforgeable and encrypted attestation report. Specifically, we propose two key techniques in this paper. The first one—aggregated attestation signature—reliably protects the attestation content from being compromised even in the presence of adversaries who have full control of the network, therefore successfully providing attestation integrity. The second one—delegation-based controlled report—introduces a third-party service to distribute the attestation report to requesters in order to save computation and communication overload on the attested party. The report is encrypted with an access policy by using attribute-based encryption and accessed by a limited number of qualified requesters, hence supporting attestation confidentiality. The experimental results show that Astrape can take no more than 0.4 s to generate an unforgeable and encrypted report for 1000 requesters and deliver a throughput speedup of approximately 30× in comparison to the existing attestation systems.

Keywords: concurrent attestation; ciphertext-policy attribute-based encryption; cloud computing

1. Introduction

Cloud computing is rapidly emerging as a novel computing paradigm and gaining more and more attention among organizations due to cost savings and simplification of the technological infrastructure [1,2]. For example, when migrating applications to clouds, organizations can remove the burden of repair and maintenance, as well as furnish united availability by way of replication and redundancy. It is prevalent for enterprises and users to embrace the cloud. However, security concerns are the strongest barrier to further adoption of cloud computing [3–5], which is complicated by cloud tenants not knowing whether their hosted applications in leased virtual environments work as expected.

Remote attestation has been proposed for this purpose by analyzing the integrity of a remote system to judge its trustworthiness. Typical attestation schemes are designed based on the following steps. First, an attestation challenger (requester) sends a target system (attester) a challenge message in which there is a random nonce against replay attack. Second, the attester responds with an evidence report about the integrity of its components in an unforgeable manner. Finally, the requester verifies the report and determines whether the status of the attested system is acceptable. Attestation systems in cloud computing typically face challenges in integrity and confidentiality protection as the open network in public clouds is a prime attack vector for adversaries. On the one hand, one common approach to guarantee integrity is to employ a standard signature [6] based on a pair of attestation keys in a Trusted Platform Module (TPM) [7,8] that is first introduced by the Trusted Computing Group (TCG). The standard signature is called the quote operation in trusted computing. On the other hand, a secure network connection between a requester and an attester is requisite to satisfy the confidentiality requirement and not to expose the detailed information of the attested system, such as a TLS/SSL-protected connection.

In a cloud environment, a set of requesters may simultaneously raise their requests to challenge the same target, such as attestations on security monitor systems [9–11] or microservices-based cloud systems [12]. While some previous works proposed techniques for TPM-based attestation over a secure connection [6,13,14], they focused on the single-requester scenario. If every requester is to run a standard attestation, the throughput of the attester would be extremely low due to the numerous operations in the signature and encryption. Although an existing solution [9] can reduce overhead in the aspect of integrity protection by aggregating a large number of attestation requests into a single TPM quote, it is still inefficient in the confidentiality protection for simultaneous attestations.

Under multi-requester scenarios, our goal is two-fold: (1) to reduce TPM quote operations, but still with attestation integrity assurance; (2) to save much computation and communication overload in the confidentiality protection. Our key idea is to decouple each requester's nonce from the attestation report, ensuring there is only a signature and encryption operation in the procedure of generating the attestation report, no matter how many requesters raise challenges. Note that the complete detailed configuration of the attested system is included in the report; therefore, only a qualified requester can have the right to access.

To such an end, we present Astrape, a flexible architecture to perform concurrent attestations under multi-requester scenarios. In particular, Astrape implements two key techniques: the first one is the aggregated attestation signature, which reduces numerous quote operations to improve integrity efficiency. The set of requesters' nonces is aggregated through building a Merkle tree to generate a root nonce to stand for all nonces and the corresponding nonce summary for the freshness validation of the requesters' nonces. The root nonce is quoted altogether with the attestation content to provide unforgeability, giving an implementation of aggregating a large number of requests into a single TPM quote. The second key technique is the delegation-based controlled report, which saves computation and communication overload on the attested party, but with confidentiality assurance. To achieve that, Astrape delegates a third party as an attestation proxy, which is responsible for the distribution of the attestation report. To mediate access control on the report, we leverage Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [15] to provide an encrypted report embedded with a policy associated with attributes of qualified requesters. As each nonce summary is directly sent to each requester in plain-text, the nonce summary can be thus excluded from the report, meaning that we only generate an encrypted report for all requesters, which reduces numerous encryption operations due to the confidentiality protection. Notice that our attestation scheme remains secure even in the presence of an attacker pretending to be a proxy as this proxy does not participate in producing an encrypted report even though we rely on its distribution jobs to various requesters.

Our contributions are summarized as follows:

- The design of a flexible architecture, Astrape, to enforce an efficient attestation for a remote cloud system under multi-requester scenarios. Astrape resorts to an attestation proxy for the distribution of the newly-generated encrypted report, in which computation and communication overhead can be drastically reduced on the attester side.
- Providing a concurrent attestation protocol with integrity and confidentiality guarantee in an efficient manner. In this protocol, we leverage nonces' aggregation, attribute-based encryption and delegation-based response to enable controlled attestation for qualified requesters posing a set of attributes to satisfy the access policy.
- Introducing the trust measure into the CP-ABE to build more expressive access policies, whereby we enable the report acquired by qualified requesters. By measuring trust based on behavior evidence, Astrape achieves a more controlled report for the trustworthy attestation.
- A proof-of-concept prototype of the architecture with concurrent attestation in a cloud environment, as well as security and performance evaluations that confirm the effectiveness and efficiency of Astrape.

The rest of this paper is organized as follows. Section 2 reviews the necessary background on the cloud computing, ciphertext-policy attribute-based encryption and remote attestation. Section 3 illustrates the architecture overview of Astrape and the threat model. Section 4 provides a detailed description of our efficient and trustworthy concurrent attestation scheme, including the aggregated attestation signature and delegation-based controlled report. The security and performance evaluations are given in Section 5. Finally, we present related work, discuss limitations and conclude in Sections 6–8, respectively.

2. Background

In this section, we present a basic introduction to cloud computing (Section 2.1), cipher-policy attribute-based encryption (Section 2.2) and remote attestation (Section 2.3). Additionally, we will give an overview of the standard attestation protocol in trusted computing (Section 2.4).

2.1. Cloud Computing

There are various definitions for cloud computing in the literature [16–18], but one unanimous definition is presented by the National Institute of Standards and Technology (NIST) [19]: “Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.” There are three models provided as services by cloud providers:

- SaaS (Software as a Service): The capability offered to cloud users is to make use of applications provided by the cloud provider. The users do not have a right of management or control over the underlying cloud infrastructure to support applications' execution.
- PaaS (Platform as a Service): The capability offered to cloud users is to deploy user-created applications by using programming languages and tools of the cloud provider. The users just have a right of control over its applications and possibly environment configurations.
- IaaS (Infrastructure as a Service): The capability offered to cloud users is to provision fundamental computing resources to deploy arbitrary software. The users have control over operating systems, storage and deployed applications.

Cloud users are able to choose any model above (SaaS, PaaS, IaaS) according to functionality and security concerns. Apparently, the IaaS model is the better choice for providing the most security assurance as the users can enforce their security policies at their will [13].

Cloud computing has become a significant paradigm due to cost savings and simplification of the technological infrastructure [1]. It provides an opportunity to companies with innovative ideas for

their services, which do not require the significant capital outlays in hardware resources any longer. Instead, cloud users can pay for computing, storage and network capability according to business needs. However, the emergence of security concerns is the strongest barrier to further adoption of cloud computing [3–5]. We enable remote attestation in cloud systems, which is proven to increase the confidence of cloud tenants for further use of cloud computing.

2.2. Ciphertext-Policy Attribute-Based Encryption

Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [15] is a new promising public-key cryptography primitive for fine-grained access control on shared sensitive data in one-to-many communications [20–22]. In CP-ABE, it does not rely on a trusted server to store and mediate access control on the sensitive data. Instead, the data are encrypted with access structures on attributes, and each user is assigned a set of attributes embedded into the user's private secret key. A user can decrypt a ciphertext to access if and only if the user possesses a certain set of attributes to satisfy the access structure in the ciphertext. The access structure is generalized as any Boolean formula over threshold gates on positive attributes and negative attributes. A CP-ABE system consists of four fundamental algorithms: Setup, KeyGen, Encrypt and Decrypt.

$Setup(\lambda) \rightarrow (PK, MK)$: the setup algorithm takes the implicit security parameter λ as an input and outputs the public parameter PK and a master key MK .

$KeyGen(MK, S) \rightarrow (SK)$: the key generation algorithm takes as inputs the master key MK and a set of attributes S ; it outputs a private decryption key SK described by S for each user.

$Encrypt(PK, M, \mathcal{T}) \rightarrow (CT)$: the encryption algorithm takes as inputs the public parameter PK , a message M and an access structure tree \mathcal{T} over the universe of attributes; it produces a ciphertext CT , which implicitly contains the access structure.

$Decrypt(SK, CT) \rightarrow (M)$: the decryption algorithm takes as inputs a private key SK and a ciphertext CT ; if the attributes set of the user S associated with SK satisfies the access structure tree \mathcal{T} in the ciphertext CT , then it will decrypt CT to obtain a message M .

Note that the efficiencies of the key generation and encryption algorithms are both fairly straightforward. The key generation algorithm requires two exponentiations for every attribute given to a user, and the encryption algorithm requires two exponentiations for each leaf node in the access structure tree of the ciphertext. These exponentiations can be quite expensive.

Based on CP-ABE, a number of fine-grained access models have been proposed for trustworthy online-data sharing in a cloud environment, such as HABE [20,23], HASBE [24], DAC-MACS [25,26] and RBKD [27]. Unlike past work on CP-ABE, we focus on the combination of remote attestation and CP-ABE for an efficient and trustworthy report under multi-requester scenarios.

2.3. Remote Attestation

TCG specifications [7,8] define an attestation mechanism for a TPM-enabled remote platform to provide the status of its components for a requester. A TCG-based attestation procedure on the attester consists of the following steps:

(1) an attester measures the platform components recursively, which generates a corresponding hash value per measurement and extends into a non-volatile and tamper-proof TPM Platform Configuration Register (PCR);

(2) the attester responds to a requester's challenge message with a digital certificate including an attester's Attestation Identity key (AIK), an AIK-signed PCR value (or multiple PCR values) and a measurement log file that is comprised of a series of integrity hash measurements about the platform components.

These integrity hash measurements enable the requester to determine where the state of the remote platform is in the unmodified (benign) state. Many systems enabled with remote attestation have been designed (e.g., IMA [6], BIND [28], TBVMM [13], DTEM [29], CloudMonatt [10,11], etc.). In the context of cloud computing platforms, a virtual TPM (vTPM) [30] is introduced to offer the

same usage model and services as a hardware TPM as the hardware TPM cannot be directly used in a cloud node.

2.4. Standard Attestation Protocol

Remote attestation needs the support of cryptographic protocols. The most basic protocol is the standard signature scheme, which was originally adopted by [6]. Figure 1 depicts the attestation protocol used by a requester to retrieve measurements and validate the integrity claims of the attester securely.

Due to network attacks (eavesdrop attack, man-in-the-middle attack, and so on), the attestation protocol requires secure communications between the requester and the attester. For secure communications over untrusted networks, the standard attestation system expects the requester and the attester to implement the SSL protocol. SSL first authenticates the two entities in Figure 1 by using the CA-signed digital certificates of their public-private key-pairs that uniquely identify them within the cloud system, then generates a temporary symmetric session key K_s to encrypt the passed messages for each attestation request.

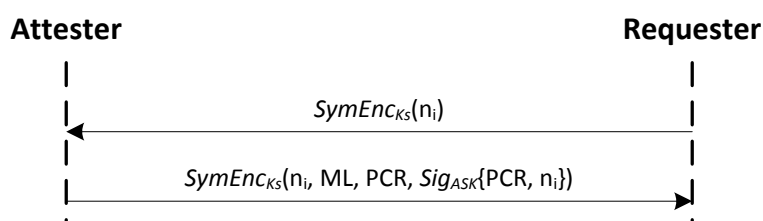


Figure 1. Standard attestation protocol.

To raise a challenge, the request first creates a unique and unpredictable 160-bit random nonce n_i and sends it in a challenge request with the confidentiality protection by K_s . The attester uses the securely-created 2048-bit private part of the attestation key, ASK , to enforce a quote operation inside the TPM chip. As described in the TPM specification [7,8], the chip signs the selected register value, PCR , and the nonce n_i to generate an unforgeable signature, $Sig_{ASK}\{PCR, n_i\}$. To complete an attestation, the attester retrieves the ordered list of all measurements, ML , and places them, together with the nonce, the PCR value and the signature into the challenge response. The response message is also encrypted by the session key, K_s , to guarantee the confidentiality requirement.

From these steps, we notice that the attestation process of the standard attestation system is designed for a single-requester scenario. When a set of requesters launches challenges simultaneously, the attesters have to perform a TPM quote operation and encrypt the corresponding response message for the respective incoming request in sequence. This is inefficient under multi-requester scenarios as the type of standard attestation not only increases the response time of each requester, but also decreases the throughput of the attester.

3. Astrape Architecture

The primary goal of Astrape is to provide an efficient attestation on cloud services under multi-requester scenarios, ensuring the integrity and confidentiality of an evidence report against malicious attackers over an untrusted network. The secondary goal of Astrape is to make the approach of Astrape practical to be deployed for large systems with less overhead.

3.1. Overview

TPM-based attestation: In trusted computing, remote attestation mechanisms need the support of cryptographic engines, which are implemented in either a hardware TPM [7,8] or software TPM [31,32]. Due to the convenience and flexibility of the deployment, it is a better choice to install the software

TPM installed in a privileged domain to act as a trust anchor for the support of trusted services, which is a tradeoff between performance and security. In addition, a hardware TPM cannot be directly used in a cloud node, in which vTPM is used as the TPM in a tenant-leased Virtual Machine (VM).

In this paper, Astrape cooperates with the underlying (hardware or software) TPM that depends on where an attested target system is deployed. For example, an attested security monitor is always directly deployed in a physical server with a hard TPM; however, due to its dynamic and changeable requirement for cloud resources, a micro-services system is hosted in a cloud node with the use of a vTPM instance, which indirectly leverages a software TPM as a trust anchor. Specifically, Astrape relies on the AIK credential to enforce a TPM quote for meeting the integrity demand on an evidence report, verified by a requester to judge whether this received report is actually originated from the attester in an unforgeability manner.

Application-level measurement: A cloud system is bootstrapped through the TCG trusted boot to reach a trustworthy status, which is composed of a set of ordered sequential steps and is only defined up to the bootstrap loader. In this process, it only takes integrity measurement of the system and stores the result in a TPM against a potentially malicious compromise. An existing approach, IMA [6], is used to maintain the chain of trust measurements up to the application layer, which takes integrity measurements as soon as executable content is loaded into the system, but before it is executed. Note that the result of the measurements is kept in an ordered list, protected by the TPM from integrity corruption attacks. To prove to a requester what software stack is loaded, the attester needs to present the TPM-protected PCR value and the ordered list. Due to a large variety of executable content, the list would grow to be significantly large.

Runtime integrity protection: Although the measurement above is loading time, the attested system can leverage a hardware-level security feature of modern CPUs to prevent runtime code corruption attacks, for example an adversary writes malicious instructions in an area of memory intended for data and then arranges to run the instructions. This security feature is called the $W \oplus X$ (Write XOR Execute) policy, stating that a page can be either writable or executable, but not both. With fine-grained page permissions supported by all modern CPUs, enforcing $W \oplus X$ policy is inexpensive and practical.

Architecture overview: Figure 2 shows an overview of the Astrape architecture. This includes three entities: (1) Requester, (2) Attester, (3) Attestation Proxy.

The requester is the initiator and end-verifier in the system and consists of two essential components. The Client is responsible for issuing an attestation with a random nonce and validating the freshness of the evidence report. The job of the Decrypter is to ask an attester to creating in advance a decryption key associated with a set of attributes and decrypt an encrypted report if and only if its key can satisfy the policy presented in the report (white box in Figure 2).

The attester acts as the attestation receiver, who is the challenged target in question. The Attestation Service at the attester-side is responsible for the consolidation of attestation challenges from a wide range of requesters and asking TPM for a TPM quote operation (signature) to generate an unforgeable attestation evidence through the TCG Software Stack. The next step is to drive the Encrypter to encrypt the evidence with the measurement logs from the IMA Module. Since CP-ABE is significantly more inefficient than symmetric encryption, the Encrypter encrypts the data with a randomly generated symmetric key and leverages CP-ABE to encrypt the symmetric key with the master key and public key according to the attributes defined in the policy. In the end, the Attestation Service delegates a proxy to respond to various requesters with the encrypted report, which can save numerous bandwidth workloads of the attester.

The responsibility of the attestation proxy is rather simpler compared to the two entities above. The proxy is a normal cloud service to distribute the newly-encrypted report. In this paper, the proxy is assumed to faithfully fulfill its duty for the attestation response to requesters rather than denial of service. We exclude the proxy from the Trusted Computing Base (TCB) as it does not have complete trustworthiness. For example, an adversary may act as a proxy to expose the detailed information of

the attested system through collusion attacks to decrypt the encrypted report. This is a reasonable assumption in cloud computing as a cloud service has a larger attack surface and open network access even though existing security mechanisms can harden software components against outside hostile attacks.

We introduce an attestation proxy for the report distribution, while the attester is responsible for the report generation. The split of jobs can achieve better scalability as we just enable several proxy replicas to be spawned when there are more incoming requests to respond to. It also implements “separation of duties” such that the server where an attester resides focuses on its responsibility (e.g., security monitoring) and a proxy is dedicated to the attestation distribution.

Typical usage scenarios: We envision a typical usage scenario in Figure 3. Here, a set of requesters sends each challenge to the attester, who can then make a trustworthy attestation on the measurement of system behaviors that the requesters are interested in and respond to them with the evidence report. The attester is a security monitor, who takes charge of the enforcement of the security policy in customers’ leased virtual machines in a cloud environment against hostile behaviors.

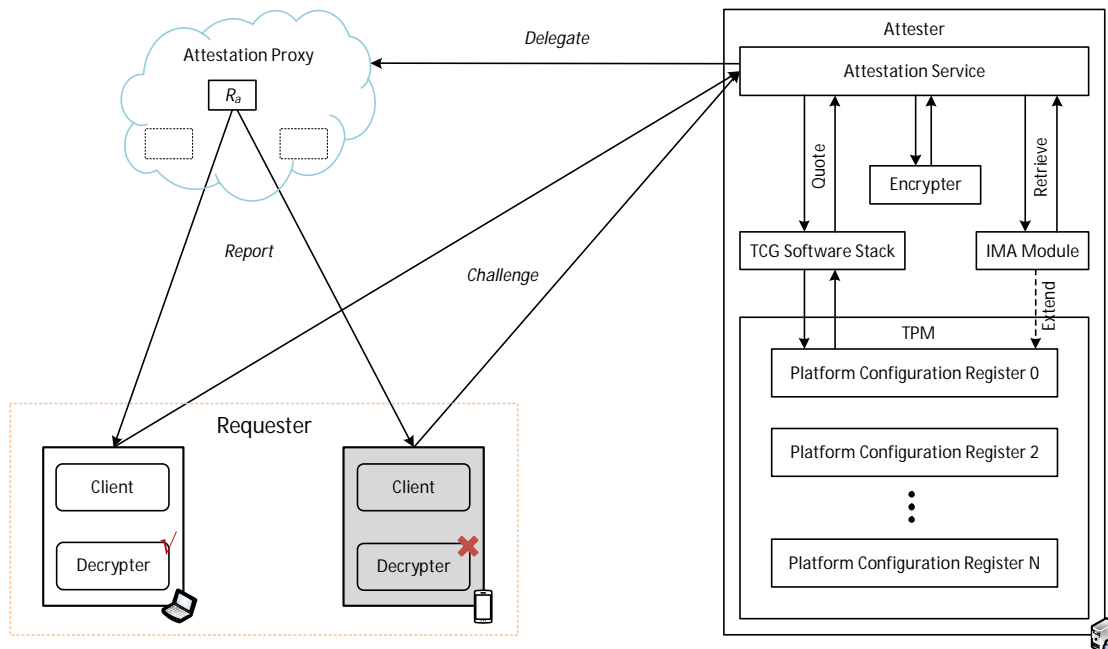


Figure 2. Astrape overview.

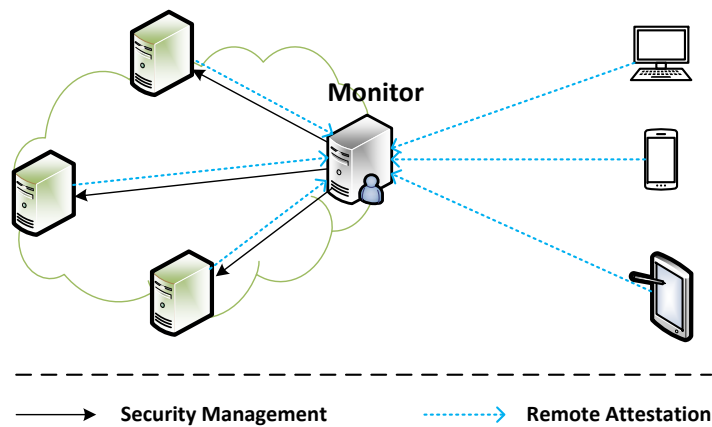


Figure 3. Illustration of how a set of various requesters could simultaneously challenge a security monitor in clouds.

We emphasize in this usage scenario that the overheads introduced by simultaneous attestations are required to be as low as possible so that the monitor can be dedicated to fulfilling its duties to manage the security jobs on the leased virtual machines. Thus, it is particularly important for the attester to generate an unforgeable and confidential evidence report with minimal overheads for a set of qualified requesters. Note that the requesters come from either outside customers or inside VMs.

3.2. Threat Model and Assumptions

Our premise is that an adversary seeks to breach the confidentiality or integrity of an attestation report. An adversary is successful if the report content is accessible to other in addition to qualified requesters.

We focus on two types of adversary's capabilities: (1) an adversary, who acts as an attestation proxy to extract the detailed information from the encrypted report through collusion attacks. Multiple parties try to decrypt the report by combining their attributes. However, we do not consider such a collusion attack that one of these colluders is able to decrypt a ciphertext on its own. (2) An active adversary has full control of the network between different involved entities, as in the standard Dolev–Yao threat model [33,34]. The adversary is capable of eavesdropping the network, as well as falsifying the attestation messages, trying to have the requester receive a forged report without detecting anything suspicious.

We assume that the TPMs are correct, and we do not consider side-channel attacks. These threats are quite valuable, but are not resolved by our Astrape for now. We also assume an adversary without the ability to physically tamper with any hardware of any host machine (i.e., CPU, memory and TPM). Whenever an attester boots a secure software platform whose TCB we assume to be correct, the adversary can no longer exploit vulnerabilities through the interface of the software platform. We do not need to require that the attestation proxy is fully trustworthy. However, we consider that the proxy is assumed to be semi-trustable (with its reputation at stake), meaning that it is always available for attestation content provision.

4. Efficient and Trustworthy Concurrent Attestation

We define the Concurrent Attestation of a remote system in cloud computing as a capability of an attester to respond to various requesters' challenge messages during overlapping time periods (we borrow the term "Concurrent" from concurrent computing notation, to indicate the simultaneity property of requesters' challenges and the efficiency property of an attester's responses in our attestation scheme). This capability implies that the attester can prove the status of the attested target with minimal overhead in a trustworthy manner.

To achieve concurrency, Astrape: (1) aggregates a set of requesters' nonces for a root nonce hash through a Merkle tree and signs the input content with the root nonce instead of requesters' nonces; (2) leverages attribute-based encryption in controlled delegating of an attestation proxy for reliable reporting to a set of qualified requester, in which the newly-generated evidence is only encrypted once due to optimizations.

4.1. Aggregated Attestation Signature

As shown in Section 2.4, when an attester receives a challenge message, it would perform an attestation signature (TPM quote), which signs a PCR value with the unique and unpredictable nonce in the message by using the ASK attestation key (the private part of AIK), $Sig_{ASK}\{PCR, n_i\}$. If every requester is to raise a standard remote attestation, the attester would process the incoming challenge messages and respond to every requester with an evidence report in a "First-In-First-Out (FIFO)" sequence. Notice that the number of signatures is proportional to the number of attestation requests. The throughput of the attester would be extremely low as there are so many signature operations. As for the requesters, they have to wait until the attester completes previous attestations, leading to longer response time for the challenging party.

To solve inefficiency issues, we batch a number of simultaneous attestation requests into a single quote operation by using a Merkle tree, as shown in Figure 4. In the Merkle tree, every leaf node is labeled with the cryptographic hash value of a nonce n_i that belongs to a requester, and every non-leaf node is labeled with the cryptographic hash value of the labels of its child nodes. With the tree, Astrape quotes a batch of N nonces (n_1, n_2, \dots, n_N) expressed as an aggregated root hash h_r (in the Figure 4, $h_r = H(h_0|h_1)$) instead of respectively quoting each nonce n_i for each attestation. In addition, Astrape needs to provide a summary for nonce n_i , $s(n_i)$, for each requester, so that they could validate the received report's freshness indicated by the aggregated root nonce hash h_r in the encrypted report. The number of nonces sent to each requester is proportional to the logarithm of the number of all nonces that the attester receives, $\mathcal{O}(\log(N))$.

Note that we can directly transmit each summary for nonce n_i to each requester in plain-text over the public non-trusted network. If the received summary fails to pass the freshness validation against the root hash in the encrypted report, we can detect suspicious behaviors such as the summary has been damaged or faked. Recall that, under our assumptions, the attestation proxy would faithfully fulfill its duties in distributing the newly-generated report to each requester. To obtain the benign nonce summary, the requester shall establish a secure connection against network adversaries.

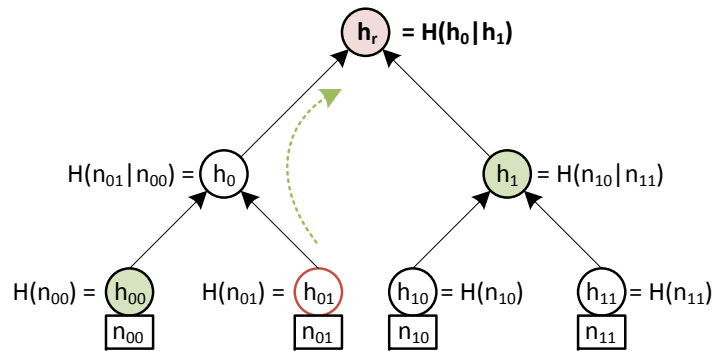


Figure 4. Merkle tree of four nonces. A summary for nonce n_{01} is composed of its near nonce n_{00} and the hash h_1 in the path to the root, $s(n_{01}) = (h_{00}, h_1)$. The root, $h_r = H(h_0|h_1)$, is an aggregated hash to represent a batch of nonces ($n_{00}, n_{01}, n_{10}, n_{11}$).

4.2. Cryptographic Keys Used

For TLS/SSL keys, an attester and each requester must have one pair of long-term public-private keys, $\{SK, VK\}$, that can be used to identify it within a cloud computing environment uniquely. This is minimally what is required for TLS/SSL support and is already present in all involved parties. For secure communications between the attester and each requester, TLS/SSL first authenticates the sender and receiver using their public-private key-pairs, then generates symmetric session keys for encrypting the passed messages.

For attestation keys, an attester owns a public-private attestation identity keys pair, $\{AVK, ASK\}$, which is created by the TPM whenever an attestation report is needed. The private key, ASK , is used to enforce a quote operation, which signs the PCR value and root nonce to generate an unforgeable signature. The public key, AVK , is used to authenticate the attester through a certificate issued by the CA, which indicates that the attestation report is authentically from the challenged attester.

For CP-ABE keys, MK , PK and SK denote master, public and private keys, respectively. Notation $SymEnc_K(M)$ indicates data M encrypted with a symmetric key K ; $Sig_K\{M\}$ indicates data M signed with an asymmetric private key K ; and $Encrypt(PK, M, \mathcal{T})$ indicates data M encrypted with a public key PK and an access structure tree \mathcal{T} under CP-ABE. We represent nonces as n , session keys as K_s and encryption keys as K_e . Nonces, session keys and encryption keys are randomly generated.

4.3. Delegation-Based Controlled Report

To ensure confidentiality for the newly-generated evidence, we generate a random symmetric key for the encryption, namely $SymEnc_{k_e}(ML, PCR, Sig_{ASK}\{PCR, h_r\})$, where ML represents the ordered list of all measurements, PCR is a cumulative hash value to verify the integrity of the list ML and h_r is used to express an aggregated root nonce hash for all requesters' nonces. Instead of sending each requester each encrypted report that has a nonce summary for the respective requester, Astrape decouples the nonce from the attestation report, which encrypts the signed evidence and sends each nonce summary to each requester in plain-text.

The following prime challenge is how to deliver the encryption key to the requesters in a reliable and efficient manner. A simple design is to embed the encryption key in the response message. Due to involving sensitive information of the attested platform in the report, an attester requires evaluating whether each requester is qualified to access the report before releasing the encryption key. However, it needs the attester to participate in every key-releasing operation, thereby introducing a performance bottleneck. An alternative design is to employ a trusted server to hold the encryption key and mediate access control. In this case, if the server is compromised, the report will be exposed in public.

In this paper, we design a key technique called delegation-based controlled report, in which we leverage Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [15] to delegate a proxy for a controlled report. Astrape first generates a pair of keys at setup time: a master key MK and a public key PK . Unlike traditional public key schemes, the public key allows the data to be encrypted and bound to an access policy. A policy is a logical expression that uses conjunction and disjunction operations over a set of terms, which can be displayed as a tree (as shown in Figure 5). Each term tests a condition over an attribute. An attribute could be described as a string or a number, and both types support the equality operation; however, the numeric type also supports inequalities (e.g., $s = x$ or $n > y$). With this CP-ABE scheme, Astrape can then create an arbitrary number of private keys SKs from the same master key, each of which is able to embed a set of attributes specified at key-creation time. The encrypted data can be decrypted only by a private key whose attributes satisfy the policy. For example, in an access tree as shown in Figure 5, if a requester is a security administrator (attribute "SecAdmin") who is in charge of the security monitor system, it can conduct the decryption of the encrypted attestation report. If a requester is an auditor (attribute "Auditor"), but its trust level (attribute "Trust_Level") is less than or equal to 0.8, its attributes cannot satisfy the access policy to decrypt the report. The detail of the trust level as an attribute will be clarified later. Furthermore, the attributes "Service_Provider" and "Cloud_Domain" mean that a requester is to lease a virtual machine protected by the security monitor in the same cloud domain; the "CloudAdmin" attribute is possessed by a cloud administrator whose responsibility is to manage the entire cloud infrastructure, naturally including the security jobs.

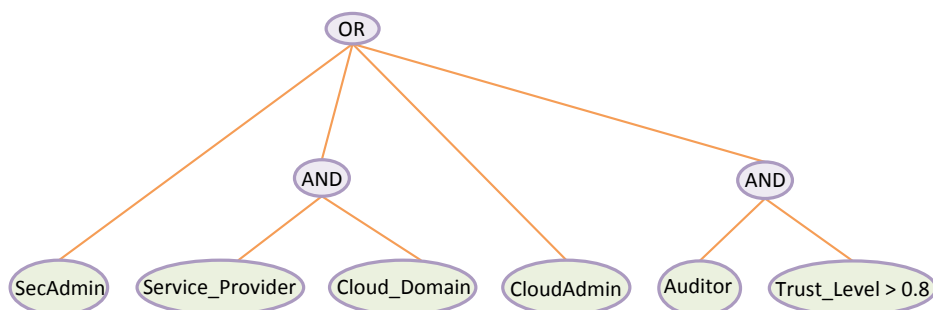


Figure 5. An access tree example for an attestation report in the aforementioned security monitor scenario.

The security of this system relies on the security of the CP-ABE keys. The attester secures the master key by: (1) ensuring it not to be released through any system interfaces; and (2) encrypting

it before storing on disk. Additionally, requesters must protect their private keys from leakage or corruption of key material.

The benefits of using CP-ABE in the Astrape system are three-fold. First, it lets the system support concurrent attestation independently of the number of requesters as the encryption operation is associated with a set of attributes instead of requesters. Second, the chosen CP-ABE scheme is resistant to collusion attacks from unauthorized users such that if there is multiparty collusion, they should only be able to decrypt a ciphertext if at least one of the parties could decrypt it on its own. As a result, Astrape can resort to a semi-trustable third party for the distribution of the encrypted report to reduce communication overheads under multi-requester scenarios. Third, the CP-ABE policy specification language can support the creation of expressive policies to enforce an access control on the encrypted report, which permits only qualified requesters to decrypt the report and read the content.

The cost using CP-ABE is a performance hit when compared to traditional cryptographic schemes. This impact can be minimized in the following two ways. First, as CP-ABE is significantly more inefficient than symmetric encryption, Astrape encrypts the newly-generated report with a randomly-created symmetric key and leverages CP-ABE to encrypt the symmetric key. Second, as generating a new private key is expensive, the attester would cache these keys in the key store so they can be resent to requesters with a set of the same attributes.

4.4. Concurrent Attestation Protocol

In a cloud computing environment where communication is over untrusted public networks, the integrity and the confidentiality are essential requirements to secure a remote attestation: they establish trust between the requester and the attester. Astrape gives a security guarantee to requesters through the unforgeable signature, as well as symmetric encryption operation on the evidence report. Due to simultaneous attestation challenges, we combine trusted computing and attribute-based encryption to provide an efficient and trustworthy attestation protocol in multi-requester scenarios.

Figure 6 shows the concurrent attestation protocol in Astrape. Initially, each requester generates each unique nonce n_i , which is used to prevent replay attacks over each channel between each requester and the attester. It acts as a challenge message to be securely transmitted to the attester. In this step, we use a temporary symmetric session key K_s to prevent unauthorized access to the nonce from hostile adversaries over an untrusted network. A session key is negotiated and created through public identity key certificates between the attester and each requester (TLS/SSL support). The attester batches a set of nonces as an aggregated hash h_r through the use of a Merkle tree, enforces a single quote operation on the root hash and the accumulated value PCR with the private attestation key ASK , $Sig_{ASK}\{PCR, h_r\}$, and then uses the required measurements ML and the quote value to generate the attestation report $(ML, PCR, Sig_{ASK}\{PCR, h_r\})$. To ensure confidentiality, Astrape encrypts this report with a randomly-generated symmetric key K_e and uses CP-ABE to encrypt this symmetric key, $Encrypt(PK, K_e, T)$. After completing these steps, the attester would send the encrypted report to this attestation proxy and respond to each requester with a short message, namely each summary $s(n_i)$ for nonce n_i .

We give a measurable qualitative perspective to demonstrate the higher efficiency of our concurrent attestation protocol. The performance of an attestation scheme can be measured in the following items: the number of digital signature operations, the number of symmetric encryption operations and the number of the CP-ABE encryption operations. In Table 1, we compare our scheme with some of the existing attestation protocol schemes, where \mathcal{N} denotes the number of requesters.

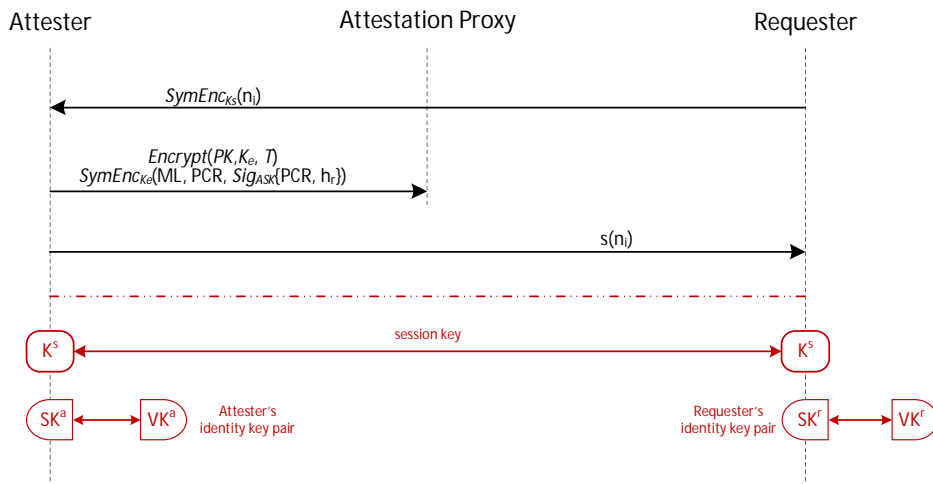


Figure 6. Concurrent attestation protocol in Astrape. We use the notation $SymEnc_K(M)$ for an encryption operation on M with a symmetric key K , $Sig_K\{M\}$ for a digital signature operation on M with an asymmetric private key K and $Encrypt(PK, M, \mathcal{T})$ for a CP-ABE encryption operation on M with a public key PK and an access structure tree \mathcal{T} .

Table 1. Comparison with related works.

	Signature Times	Symmetric Encryption Times	CP-ABE Encryption Times
[6]	$\mathcal{O}(\mathcal{N})$	$\mathcal{O}(\mathcal{N})$	0
[28]	$\mathcal{O}(\mathcal{N})$	$\mathcal{O}(\mathcal{N})$	0
[9]	$\mathcal{O}(1)$	$\mathcal{O}(\mathcal{N})$	0
This work	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$

4.5. Measuring Trust in Clouds

In Section 4.3, we introduce the trust level as an attribute of a third-party auditor (a requester) since the security of a cloud platform depends on the selection of trustworthy auditors so that an auditor worthy of trust can be granted to extract detailed information of the cloud. In this paper, we apply a trust model [35] to measure the trust level of an auditor based on evidence related to the auditor from past interactions. In this model, the trust level E_{Trust} is an expectation value, defined as:

$$E_{Trust} = c \times t + (1 - c) \times f, \tag{1}$$

where $0 \leq t \leq 1$, $0 \leq f \leq 1$, $0 \leq c \leq 1$.

In Equation (1), t denotes the average trust level calculated through the outcomes of past interactions, f denotes the initial trust level to express dispositional trust and c is a certainty value to denote a measure for the representability of the average trust level as an estimate for the outcome of future interactions. The average trust level t and the certainty value c can be calculated using:

$$t = \begin{cases} 0 & \text{if } n_{pos} + n_{neg} = 0, \\ \frac{n_{pos}}{n_{pos} + n_{neg}} & \text{otherwise.} \end{cases} \tag{2}$$

$$c = \frac{N \times (n_{pos} + n_{neg})}{2 \times (N - n_{pos} - n_{neg}) + N \times (n_{pos} + n_{neg})} \tag{3}$$

In Equations (2) and (3), the parameter N is introduced to allow for the definition of a (finite) number of pieces of evidence, which is expected to be sufficient to consider the collected evidence as representative for the behavior of an auditor, and n_{pos} , n_{neg} refer to the number of collected positive and negative pieces of evidence, respectively.

When providing the trust level as an attribute for the attestation protocol, it can determine the status of a requester based on behavior evidence to enforce more expressive access policies for achieving trustworthy attestation. Astrape excludes some less credible requesters by putting a limit on an attestation report so that only qualified requesters can acquire the content.

5. Evaluation

This section describes the setup used to perform the experiments designed to evaluate the performance and security of the proposed Astrape. The results of the experimental validation are presented and discussed.

5.1. Experimentation Setup

Our testbed is a Lenovo ThinkPad T440s, featuring a 1.7-GHz Intel[®] Core i5-4210U CPU, with 12 GB of RAM. The host OS is a Linux Mint 18.3 OS (4.8.0 Linux kernel), used as an attester. We integrated the TPM-emulator [31,32] that is used to emulate the functions of the trusted hardware module and enabled the IMA module in the kernel space to collect the measurement logs of applications. We leveraged the Trousers software stack and its related tools [36] to interact with the software TPM. In addition, we used the OpenSSL crypto library [37] and the CP-ABE toolkit [38] for all cryptographic operations. In all experiments, we repeated each one ten times and recorded the median results, in which the standard deviation was negligible. In the experiments, we compared Astrape with attestation protocols of two widely-used systems: standard attestation [6] and batch attestation [9].

5.2. Performance Evaluation

We first evaluate the nonces aggregation performance (Section 5.2.1), aggregated attestation signature performance (Section 5.2.2) and report symmetric encryption performance (Section 5.2.3). We then evaluate the performance overhead of CP-ABE operations for its three main activities: generating CP-ABE private keys (Section 5.2.4) and CP-ABE encryption and decryption (Section 5.2.5). Finally, we evaluate the full system performance by performing the generation of attestation reports with various attributes (Section 5.2.6). Table 2 depicts the length of the keys used by Astrape.

Table 2. Length of cryptographic keys used by Astrape.

n_i	h_r	PCR	VK	SK	AVK	ASK	K_s	K_e
160-bit	160-bit	160-bit	2048-bit	2048-bit	2048-bit	2048-bit	256-bit	256-bit

5.2.1. Nonces' Aggregation

Generating an aggregated root nonce hash is to build a Merkle tree based on a set of nonces from requesters. Its overhead depends on the number of nonces (each requester raises an attestation with each nonce). We measured the time to aggregate a set of nonces to generate a root nonce, in which we varied the number of requesters from 10–1000. This range seemed reasonable to characterize the concurrency of attestation requests.

Figure 7 shows the results under various requester scenarios, which shows a significant finding that although we performed each SHA-1 hash operation on each leaf node in a built Merkle tree, the overhead introduced by the hash operation is acceptable in comparison to the batch attestation. For example, an aggregation with 940 nonces took 0.43 milliseconds in batch attestation and 0.67 milliseconds in Astrape, respectively, in which there was a maximum difference of 0.24 milliseconds. However, with these additional hash operations, Astrape can send each corresponding nonce summary to each requester in plain-text unlike the batch attestation, which is over a secure connection.

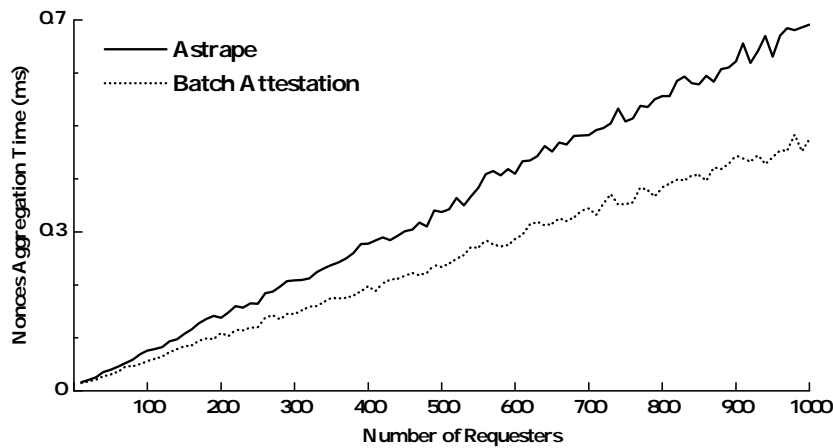


Figure 7. Performance overhead of nonces' aggregation. Time to generate an aggregated root hash using a Merkle tree as we vary the number of requesters.

5.2.2. Aggregated Attestation Signature

We measured the performance overhead of the attestation signature in standard attestation, batch attestation and Astrape by varying the number of requesters whose range was in $\{r \mid r \in \mathbb{N}, \text{ and } 100 \leq r \leq 1000, \text{ and } 100|r\}$. In the experiment, we used two various sizes of measurement list ML (512 KB and 2048 KB), a core part of attestation content (see Section 4.4). Figures 8 and 9 shows the result of the attestation signature in three various systems. From the two figures, we make two observations:

(1) For standard attestation, the overheads of the signature operations grow with the number of requesters as performing the same signature operations as the number of requesters. However, for batch attestation and our Astrape, the overheads remain almost invariable due to the nonces' aggregation, which makes multiple attestation requests become a single quote operation by using a Merkle tree. When the number of requesters is 1000, the signature time in the standard attestation is approximately 1.41 s, whose throughput is 0.71 quote/s. However, the signature time of Astrape is approximately 0.002 s for 1000 requests, and the time of the related nonces' aggregation operation is approximately 0.001 s. This means that the overheads of signature and nonces' aggregation in Astrape are far less than the overheads of the signature operations in the standard attestation.

(2) The overhead of the signature in batch attestation and Astrape is independent of attestation content size. The signature time in Figure 8 is almost two milliseconds, for which the same observation could be seen in Figure 9 as they performs the signature operation once due to the nonces' aggregation.

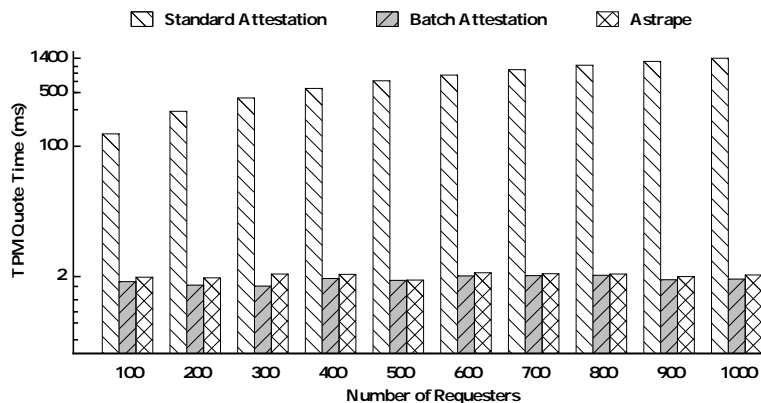


Figure 8. Performance overhead of the attestation signature. Time to generate signatures for attestation reports as we vary the number of requesters, with a measurement list of constant size (512 KB).

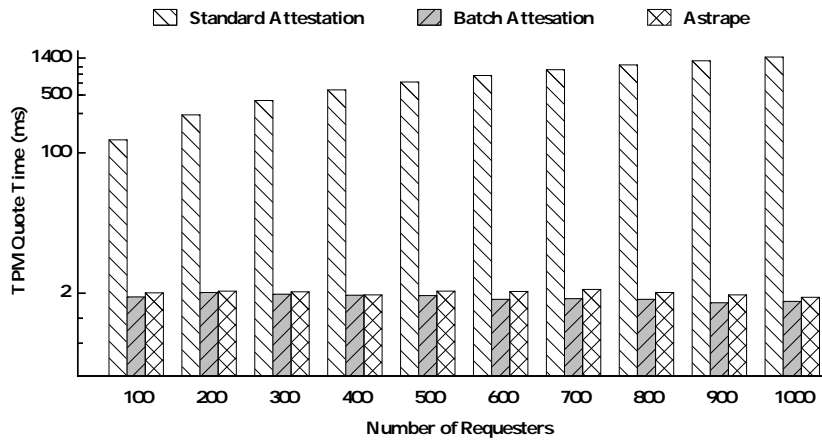


Figure 9. Performance overhead of the attestation signature. Time to sign attestation reports as we vary the number of requesters, with a measurement list of constant size (2048 KB).

As a result, the aggregated signature technique is crucial to improving the attestation performance in Astrape although introducing additional cost. The cost results from the nonces’ aggregation operation, but is acceptable to prove the comparisons above.

5.2.3. Symmetric Encryption

To study the effect of symmetric key encryption for the attestation content, we evaluate the performance overhead in two aspects as follows. First, we varied the size of the measurement list *ML* with two constant numbers of requesters (100 and 500) to gain the time to generate encrypted reports. Second, we measured the time to complete the encryption operations under various numbers of requesters conditions, in which the size of the measurement list is 512 KB and 2048 KB, respectively. Figures 10–13 show the overhead due to symmetric encryption under four settings. From the four figures, we make the following observations.

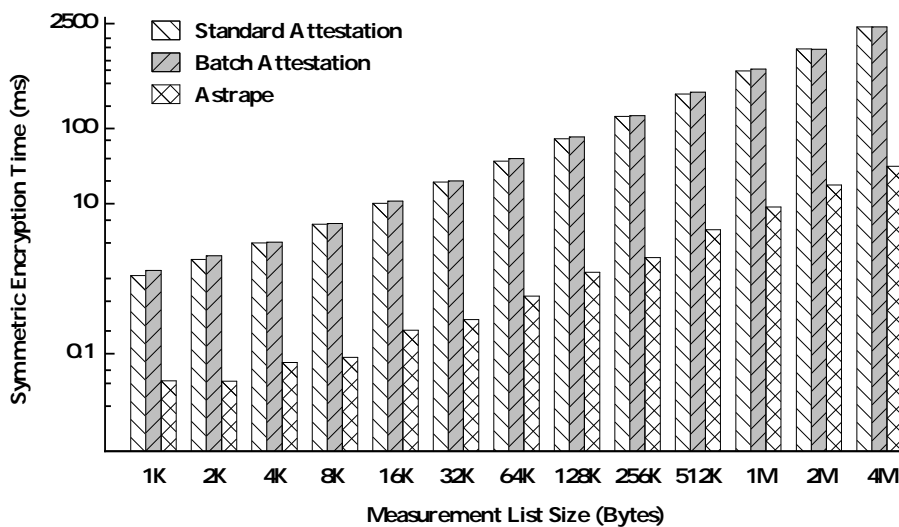


Figure 10. Performance overhead of encrypting reports with symmetric keys, which varied the size of the content with the constant number of requesters (100 requesters).

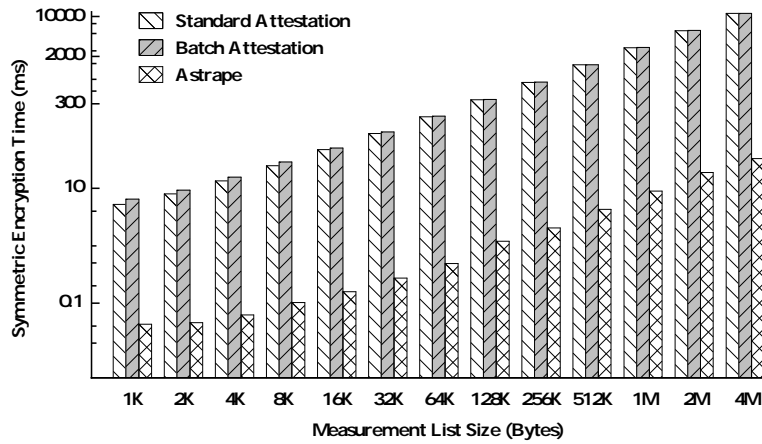


Figure 11. Performance overhead of encrypting reports with symmetric keys, which varied the size of the content with the constant number of requesters (500 requesters).

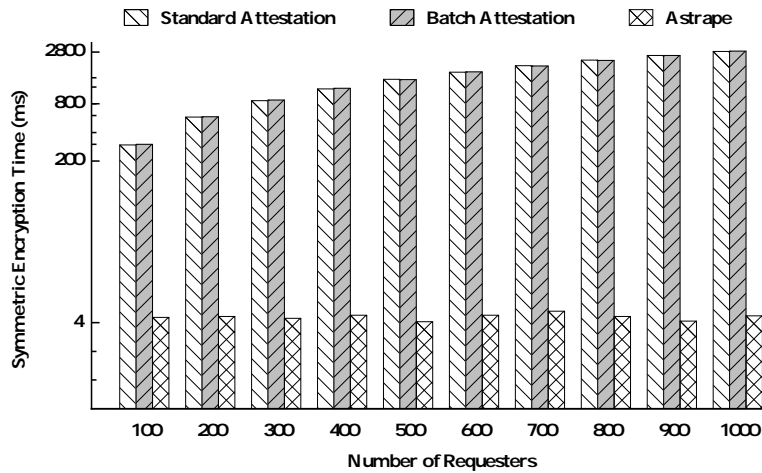


Figure 12. Performance overhead of encrypting reports with symmetric keys, which varied the number of requesters with a measurement list of constant size (512 KB).

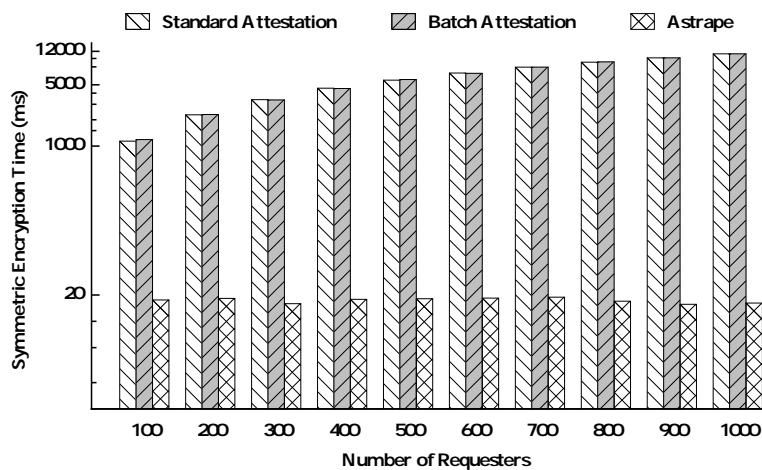


Figure 13. Performance overhead of encrypting reports with symmetric keys, which varied the number of requesters with a measurement list of constant size (2048 KB).

(1) The performance overhead of encryption in standard attestation, batch attestation and Astrape grows with the size of the input evidence (see Figures 10 and 11). When the size of the attestation

content becomes large, the encryption overhead of standard attestation and batch attestation grows sharply under the 500 requester scenario. For example, with the 4-MB measurement list, symmetric encryption operations took more than 10 s in standard attestation and batch attestation systems due to encrypting many times, which is the same as the number of requesters.

(2) As shown in Figures 12 and 13, the overhead of Astrape's encryption operations is fixed due to only one operation in the attestation procedure regardless of how many requesters raise trustworthy challenges. However, the overhead of standard attestation and batch attestation grows mightily under the 2048 KB-content condition. For instance, the time to encrypt the attestation content for 1000 requesters is over 11 s.

In summary, our evaluation of symmetric encryption operations shows that Astrape has higher performance advantages over the other two systems. The performance overhead remains almost constant when the number of requesters increases; the effect of large attestation content is still acceptable. For example, it took about 33 milliseconds to encrypt 4 MB of content with 500 requesters in Astrape.

5.2.4. Private Key Generation

The overhead of a CP-ABE private key relies on the number of attributes that are present in this key. We measured the time to generate a private key stemming from the same master key. Figure 14 shows the performance of the evaluated private key generation by varying the number of attributes from 1–50. The range seemed reasonable to characterize various requesters. From the figure, we make two observations, which confirm the findings of the authors of the original CP-ABE scheme [15].

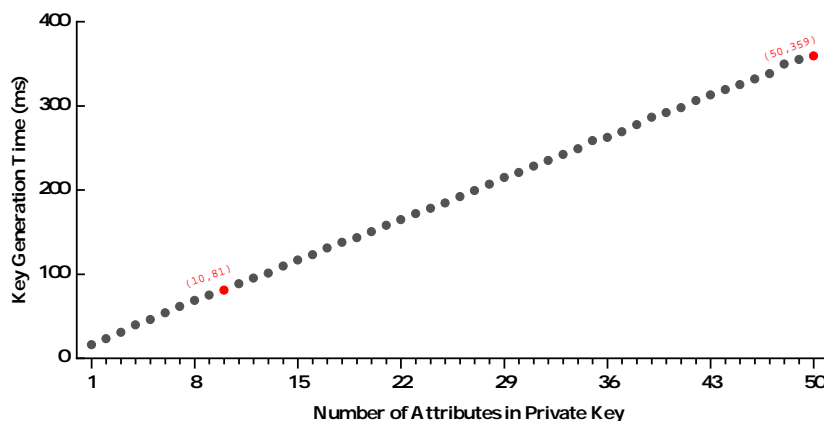


Figure 14. Performance of CP-ABE private key generation. Time to generate a private key as we vary the number of attributes.

(1) The overhead of generating a private key grows linearly with the number of attributes embedded in this key. The more attributes a private key contains, the longer time this key takes to create.

(2) Generating a CP-ABE private key is expensive. For instance, a key with 10 attributes took 0.081 s to create, and a key with 50 attributes took 0.359 s, which corresponds to a maximum rate of 12.35 and 2.79 keys/s, respectively.

The observations above show that private-key generation in CP-ABE is inherently inefficient. However, we still consider that its performance is acceptable when throughput pressure on the attester is relatively low because large groups of requesters are likely to have the same set of attributes. The latency to generate a key is experienced only at the first request time. Since the key is cached, it is reused in some future requests without additional costs, in which the requesters have identical attributes.

5.2.5. CP-ABE Encryption and Decryption

The use of CP-ABE in Astrape is to encrypt a 256-bit symmetric key, which is used to encrypt the attestation evidence. To understand the performance overhead of CP-ABE, we varied the number of leaf nodes in an access policy for encrypting and the number of attributes used by a policy for decrypting with the input data of constant size (256 bits). Figures 15 and 16 shows the performance overhead of encrypting and decrypting 256 bits of data as a function of policy complexity. From the two figures, we make the following observations:

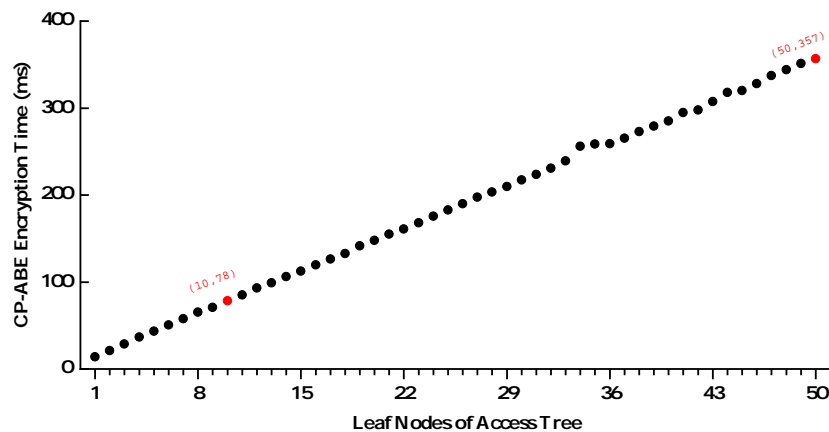


Figure 15. Performance overhead of CP-ABE encrypting data as a function of the complexity of the policy, with input data of constant size (256 bits, the length of an encryption key for attestation reports).

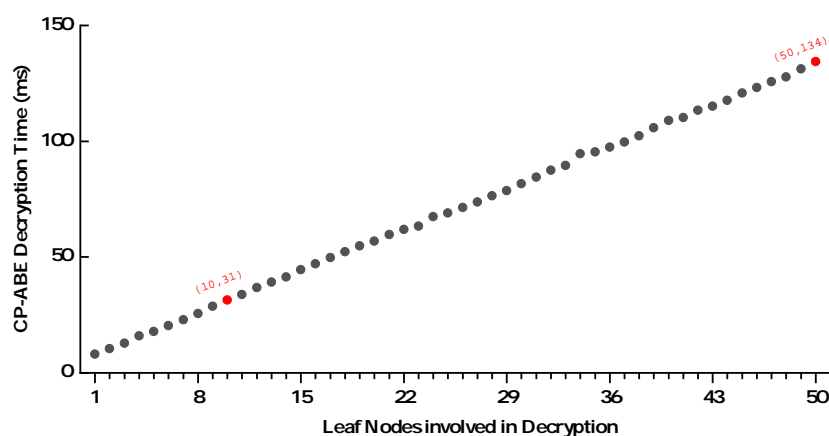


Figure 16. Performance overhead of CP-ABE decrypting data as a function of the complexity of the policy, with input data of constant size (256 bits, the length of an encryption key for attestation reports).

(1) Figure 15 shows the encryption cost, which confirms the relevant findings of the original authors of CP-ABE. For instance, the cost of encryption for a policy with 10 attributes was about 78 milliseconds, and the other with 50 attributes was about 357 milliseconds. Our observations show that the encryption cost grows linearly with the number of leaf nodes of the access tree.

(2) Figure 16 shows the decryption cost, which confirms the relevant findings of the original authors of CP-ABE. Unlike CP-ABE encryption, CP-ABE decryption depends on the number of attributes in the private key to satisfy the specified policy; for example, considering a private key with attributes (A, B, C) and two policies " $P_1 : A$ ", " $P_2 : A \text{ and } B \text{ and } C$ ". Policy P_1 uses one attribute, whereas P_2 uses three. From Figure 16, the time to decrypt with a policy containing 10 attributes was 31 milliseconds, and the other with 50 attributes was 134 milliseconds.

Our evaluation of the CP-ABE aspect in Astrape gives the following conclusions: (1) the cost of CP-ABE encryption on the attester-side was reasonable since fewer attributes can describe enough requesters in a concurrent attestation procedure, for example 10 attributes could be used to differentiate up to 1024 types of requesters; (2) the cost of CP-ABE decryption on the requester-side was also reasonable when taking into account how infrequently they are required. Note that the decryption time may depend significantly on the set of attributes involved since an access policy can be satisfied by different private keys. To avoid discrepancy caused by the various keys and ensure uniformity of the decryption tests, we built an access policy tree only using the AND gate.

5.2.6. Full System Performance

In this section, we measured the full system performance to evaluate the overhead of generating an unforgeable and encrypted report in Astrape. We use a combination of nonces' aggregation, attestation signature, symmetric encryption and CP-ABE encryption operations as the workload, in which we measured the (report generation) time by varying the number of requesters with two various sizes (512 KB and 2048 KB) of measurement list *ML*. For brevity, we used 10 and 50 attributes to describe different types of requesters in Figures 17 and 18, respectively.

Figures 17 and 18 show the report generation time for 512 KB- and 2048 KB-sized *ML* (evidence) by varying different numbers of requesters. From the two figures, we make the observation that the performance overhead of our Astrape is far lower than the two other systems (standard attestation and batch attestation) and independent of the number of requesters. For the 512-KB-*ML* content evaluation, the generation time is about 85 milliseconds under the 10-attribute scenario and 363 milliseconds under the 50-attribute scenario. For the 2048-KB *ML* evaluation, the generation time with 10 attributes is about 98 milliseconds, and the other with 50 attributes is 376 millisecond. The difference is attributed to one factor: CP-ABE encryption. The standard attestation and batch attestation systems took about 13 s and 11 s to generate reports for 1000 requesters, respectively, indicating that our Astrape system delivered a throughput speedup of approximately $30\times$. As such, Astrape has more advantages over standard attestation and batch attestation in the aspect of communication workloads. Figure 19 demonstrates our observation that the size of the attestation report generated by Astrape is about 2 MB regardless of how many challenging requesters. In contrast, the size of the reports generated by standard attestation and batch attestation yet grows with the number of requesters, from 200 MB to approximately 2 GB. The improvement is contributed to by the one-time signing and encrypting for the collected evidence when attested.

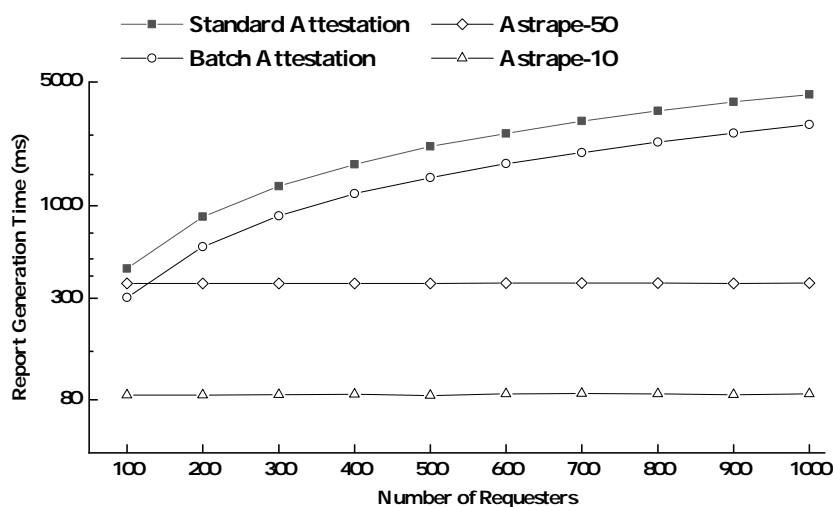


Figure 17. Performance overhead of attestation report generation. Time to generate encrypted reports as we vary the number of requesters, with a measurement list of constant size (512 KB). Ten and 50 stand for the number of attributes used in the policy of Astrape.

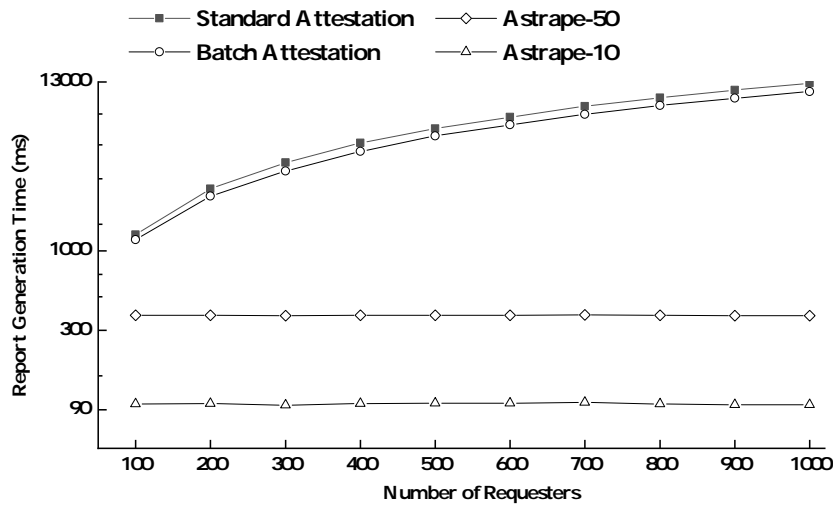


Figure 18. Performance overhead of attestation report generation. Time to generate encrypted reports as we vary the number of requesters, with a measurement list of constant size (2048 KB). Ten and 50 stand for the number of attributes used in the policy of Astrape.

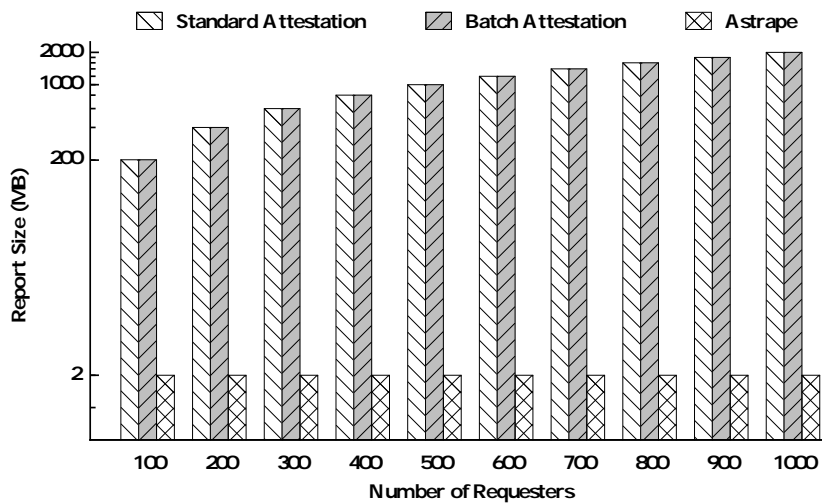


Figure 19. Generated attestation report sizes. The size of generated reports as we vary the number of requesters, with a measurement list of constant size (2048 KB).

In conclusion, Astrape is more efficient in improving report generation performance and reducing communication overhead due to the requesters’ nonces being decoupled from the generated evidence. Despite introducing inefficient CP-ABE, Astrape can efficiently limit its overhead, which leverages CP-ABE to encrypt a symmetric key and caches the generated private keys in the key store securely so that they can be resent to requesters with a set of the same attributes.

5.3. Security Evaluation

The trusted entities in the Astrape system include the Attester and the Requester. It is significant that these machines have a trusted boot and are secured for runtime protection. Traditional approaches can be leveraged to defend these entities, e.g., enabling firewalls, enforcing SELinux, data hashing and encryption in the database, etc. Note that we put the attestation proxy in a public cloud without any additional protection and require the proxy to do its job well with its reputation at stake.

6. Related Work

Different techniques have been proposed for trustworthy attestation. We describe some past work in attribute-based encryption and remote attestation.

Attribute-based encryption: The concept of Attribute-Based Encryption (ABE) was introduced by [39] as a step towards developing encryption systems with high expressiveness. Two different and complementary notions of ABE were defined: Key-Policy Attribute-Based Encryption (KP-ABE) [40] and CP-ABE [15]. In a KP-ABE system, a ciphertext is associated with a set of attributes and a private key is associated with an access structure; however, in a CP-ABE system, a ciphertext is associated with an access structure and a private key is associated with a set of attributes. Goyal et al. [41] further presented a generic construction to transform a KP-ABE scheme into a CP-ABE scheme.

Recent years have seen that CP-ABE can be more useful than KP-ABE in practical usage scenarios. The first CP-ABE scheme, proposed by [15], is an expressive scheme that enables fine-grained access control over ciphertext. The CP-ABE scheme and improved variants are widely used for trustworthy online data sharing in a cloud environment, such as HABE [20,23], HASBE [24], DAC-MACS [25,26] and RBKD [27]. In addition, Qiao [42] retrofitted the original CP-ABE scheme against privilege abuse in fog computing, and Fan [43] combined CP-ABE and the Trusted Execution Environment (TEE) to enforce a fine-grained access control so as to prevent the leakage of sensitive information in mobile platforms.

Astrape continues the line of research in leveraging CP-ABE for security control in cloud computing, which adds to the literature by showing that the CP-ABE-based access control on the encrypted report can be leveraged for trustworthy concurrent attestation against collusion attacks and privacy leakage launched by the untrustworthy attestation proxy.

Remote attestation: The first feasible implementation scheme of the attestation protocol, proposed in [6], is a simple scheme that only enables the attestation challenge between a single requester and a single attester. Then, various attestation schemes [9,28] were proposed. BIND [28] was designed to support concurrent attestations; however, the type of attestation was for multiple attested targets, meaning that it was under multi-attester scenarios and the attestations may have been raised by a single requester or multiple requesters. The work in [9] enhanced the attestation protocol with a technique that can batch a large number of requests into a single TPM quote using a Merkle tree. This scheme was able to reduce overhead and improve the throughput of the attester in the aspect of integrity guarantee; however, it still faces another efficiency challenge in the confidentiality, meaning that it needs the same number of symmetric encryption operations of the entire report content as the requesters.

Astrape increases efficiency in the guarantee of integrity and confidentiality by leveraging Merkle trees and ciphertext-policy attribute-based encryption, which can not only batch multiple requests into a single TPM quote, but also support only symmetric encryption of the attestation report under multi-requester scenarios.

7. Discussion and Limitation

Honest attestation proxy: In this paper, the Astrape system leverages CP-ABE to encrypt the attestation report and deliver it to the attestation proxy for the distribution of the report. The private key generation of a requester and encryption of the report are done on the attester, so our attestation remains secure even in the presence of an attacker that acts as the proxy. However, if we verify that the proxy is honest and trustworthy through some techniques, such as Intel's Software Guard Extensions (SGX) [44] and AMD's Secure Encrypted Virtualization (SEV) [45], we enable the attester to delegate the most laborious tasks of CP-ABE. In this case, it places minimal load on the attester upon CP-ABE operations to avoid being the performance bottleneck in the attesting servers.

More measurements to be attested: Our Astrape only attests code integrity measurement to various requesters in cloud computing, which is able to determine whether the remote attested system works as expected. However, our approach can be used to leverage bytecode runtime

measurement [14,29,46] to provide more dynamic protection for some high-level applications, such as Java-based cloud services. In addition, Astrape is also retrofitted to support other types of security measurements, such as the cornerstone measurements of confidentiality and availability [10,11] and virtual machine introspection-based [47,48] behavior measurements [49,50]. With these added measurements into our Astrape, the cloud party is more capable of defending against malicious attacks, which would give cloud customers more confidence to migrate their applications to the cloud. In the business context, these measurements can also contribute to the selection of credible organizations in cloud computing.

Shorter CP-ABE ciphertext: Our Astrape architecture relies on CP-ABE [15,51] to encrypt the report with an access structure associated with the user's attributes. One of the main efficiency drawbacks of the ABE systems is that the size of the ciphertext grows at least linearly with the number of attributes involved in the access structure. The size of the ciphertext in this paper is $s + \mathcal{O}(1)$ where s is the number of attributes involved in the access structure. Given the existing issue, we can leverage other efficient attribute-based encryption systems [52–54] whose ciphertext has a constant size.

8. Conclusions

Although remote attestation has been proven to boost the confidence of tenants in the cloud, the inefficiency of the current attestation schemes under multi-requester scenarios becomes an obstacle for their further use. To such an end, we propose an efficient and trustworthy concurrent attestation architecture for multiple requester challenges, Astrape, to support integrity and confidentiality protection for the attestation with minimal overhead. Astrape achieves its goal by two key techniques: the first technique aggregates all requesters' nonces through a Merkle tree to batch a large number of request into a single quote operation, thus providing efficient attestation integrity; the second key technique leverages the attribute-based encryption to produce an encrypted controlled evidence report and delegates a proxy to respond to a qualified requester with this report. A proof-of-concept system has been developed to demonstrate the effectiveness and efficiency of Astrape. Evaluations show that Astrape can achieve a throughput speedup of approximately $30\times$ for 1000 requesters compared to the existing standard attestation [6] and batch attestation [9].

We plan to extend our work in several directions. First, we plan to extend Astrape to work with a trustworthy third-party proxy in an SGX-enable enclave so that the proxy can do the most laborious tasks of attribute-based encryption to have more efficient attestation, but with lower overhead on the attester-side. Second, as there is a trend to perform computing tasks on mobile devices with the emergence of mobile edge computing [55,56], we intend to deploy our Astrape on the most widely-used Android platform to improve security and see the performance implications.

Author Contributions: H.B. and H.Z. contributed equally to the design of the ideas, the analysis of results and the writing of the paper. S.M, H.Q., T.H., Z.W. and J.R. proofread the paper.

Funding: This research is supported by the National Natural Science Foundation of China (Grant No. 61303191 and No. 61402508) and the National High Technology Research and Development Program of China (Grant No. 2015AA016010).

Acknowledgments: We would like to acknowledge our families, friends and colleagues for their support, suggestions and reviews while conducting this study. We are also thankful to them for the successful completion of this part of the project.

Conflicts of Interest: The authors declare that there is no conflict of interest regarding the publication of this manuscript.

Abbreviations

The following abbreviations are used in this manuscript:

ABE	Attribute-Based Encryption
AIK	Attestation Identity Key
CA	Certificate Authority

CP-ABE	Ciphertext-Policy Attribute-Based Encryption
FIFO	First-In-First-Out
IMA	Integrity Measurement Architecture
KP-ABE	Key-Policy Attribute-Based Encryption
PCR	Platform Configuration Register
SELinux	Security-Enhanced Linux
SEV	Secure Encrypted Virtualization
SGX	Software Guard Extensions
TCB	Trusted Computing Base
TCG	Trusted Computing Group
TEE	Trusted Execution Environment
TPM	Trusted Platform Module
vTPM	virtual TPM
VM	Virtual Machine

References

- Palos-Sanchez, P.R. Drivers and Barriers of the Cloud Computing in SMEs: The Position of the European Union. *Harv. Deusto Bus. Res.* **2017**, *6*, 116–132. [[CrossRef](#)]
- Armbrust, M.; Fox, A.; Griffith, R.; Joseph, A.D.; Katz, R.; Konwinski, A.; Lee, G.; Patterson, D.; Rabkin, A.; Stoica, I.; Zaharia, M. A View of Cloud Computing. *Commun. ACM* **2010**, *53*, 50–58. [[CrossRef](#)]
- Takabi, H.; Joshi, J.B.; Ahn, G.J. Security and Privacy Challenges in Cloud Computing Environments. *IEEE Secur. Priv.* **2010**, *8*, 24–31. [[CrossRef](#)]
- Zissis, D.; Lekkas, D. Addressing Cloud Computing Security Issues. *Future Gener. Comput. Syst.* **2012**, *28*, 583–592. [[CrossRef](#)]
- Palos-Sanchez, P.R.; Arenas-Marquez, F.J.; Aguayo-Camacho, M. Cloud Computing (SaaS) Adoption as a Strategic Technology: Results of an Empirical Study. *Mob. Inf. Syst.* **2017**, *2017*, 2536040. [[CrossRef](#)]
- Sailer, R.; Zhang, X.; Jaeger, T.; van Doorn, L. Design and Implementation of a TCG-based Integrity Measurement Architecture. In Proceedings of the 13th USENIX Security Symposium, USENIX Association, San Diego, CA, USA, 9–13 August 2004; pp. 223–238.
- TCG. TPM Main Specification. Available online: <https://trustedcomputinggroup.org/resource/tpm-main-specification/> (accessed on 8 March 2018).
- TCG. TPM Library Specification. Available online: <https://trustedcomputinggroup.org/resource/tpm-library-specification/> (accessed on 8 March 2018).
- Santos, N.; Rodrigues, R.; Gummadi, K.P.; Saroiu, S. Policy-sealed Data: A New Abstraction for Building Trusted Cloud Services. In Proceedings of the 21st USENIX Security Symposium, USENIX Association, Bellevue, WA, USA, 8–10 August 2012; pp. 175–188.
- Zhang, T.; Lee, R.B. CloudMonatt: An Architecture for Security Health Monitoring and Attestation of Virtual Machines in Cloud Computing. In Proceedings of the 42nd Annual International Symposium on Computer Architecture, Portland, OR, USA, 13–17 June 2015; pp. 362–374.
- Zhang, T.; Lee, R.B. Design, Implementation and Verification of Cloud Architecture for Monitoring a Virtual Machine's Security Health. *IEEE Trans. Comput.* **2018**, *67*, 799–815. [[CrossRef](#)]
- Dragoni, N.; Giallorenzo, S.; Lafuente, A.L.; Mazzara, M.; Montesi, F.; Mustafin, R.; Safina, L. Microservices: Yesterday, Today, and Tomorrow. In *Present and Ulterior Software Engineering*; Springer: Berlin, Germany, 2017; pp. 195–216.
- Mei, S.; Wang, Z.; Cheng, Y.; Ren, J.; Wu, J.; Zhou, J. Trusted Bytecode Virtual Machine Module: A Novel Method for Dynamic Remote Attestation in Cloud Computing. *Int. J. Comput. Intell. Syst.* **2012**, *5*, 924–932. [[CrossRef](#)]
- Ba, H.; Zhou, H.; Qiao, H.; Wang, Z.; Ren, J. RIM4J: An Architecture for Language-Supported Runtime Measurement against Malicious Bytecode in Cloud Computing. *Symmetry* **2018**, *10*, 253. [[CrossRef](#)]
- Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-Policy Attribute-Based Encryption. In Proceedings of the 2007 IEEE Symposium on Security and Privacy, San Jose, CA, USA, 20–23 May 2007; pp. 321–334.
- Jula, A.; Sundararajan, E.; Othman, Z. Cloud Computing Service Composition: A Systematic Literature Review. *Expert Syst. Appl.* **2014**, *41*, 3809–3824. [[CrossRef](#)]

17. Stieninger, M.; Nedbal, D. Characteristics of Cloud Computing in the Business Context: A Systematic Literature Review. *Glob. J. Flex. Syst. Manag.* **2014**, *15*, 59–68. [[CrossRef](#)]
18. Radu, L.D. Green Cloud Computing: A Literature Survey. *Symmetry* **2017**, *9*, 295. [[CrossRef](#)]
19. Mell, P.; Grance, T. The NIST Definition of Cloud Computing, Technical Report 2011. Available online: <https://csrc.nist.gov/publications/detail/sp/800-145/final> (accessed on 5 March 2018).
20. Wang, G.; Liu, Q.; Wu, J. Hierarchical Attribute-based Encryption for Fine-grained Access Control in Cloud Storage Services. In Proceedings of the 17th ACM Conference on Computer and Communications Security, Chicago, IL, USA, 4–8 October 2010; pp. 735–737.
21. Li, J.; Yao, W.; Zhang, Y.; Qian, H.; Han, J. Flexible and Fine-Grained Attribute-Based Data Storage in Cloud Computing. *IEEE Trans. Serv. Comput.* **2017**, *10*, 785–796. [[CrossRef](#)]
22. Zuo, C.; Shao, J.; Liu, J.K.; Wei, G.; Ling, Y. Fine-Grained Two-Factor Protection Mechanism for Data Sharing in Cloud Storage. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 186–196. [[CrossRef](#)]
23. Wang, G.; Liu, Q.; Wu, J.; Guo, M. Hierarchical Attribute-based Encryption and Scalable User Revocation for Sharing Data in Cloud Servers. *Comput. Secur.* **2011**, *30*, 320–331. [[CrossRef](#)]
24. Wan, Z.; Liu, J.; Deng, R. HASBE: A Hierarchical Attribute-Based Solution for Flexible and Scalable Access Control in Cloud Computing. *IEEE Trans. Inf. Forensics Secur.* **2012**, *7*, 743–754. [[CrossRef](#)]
25. Yang, K.; Jia, X. Attributed-Based Access Control for Multi-authority Systems in Cloud Storage. In Proceedings of the 2012 IEEE 32nd International Conference on Distributed Computing Systems, Macau, China, 18–21 June 2012; pp. 536–545.
26. Yang, K.; Jia, X.; Ren, K.; Zhang, B.; Xie, R. DAC-MACS: Effective Data Access Control for Multiauthority Cloud Storage Systems. *IEEE Trans. Inf. Forensics Secur.* **2013**, *8*, 1790–1801. [[CrossRef](#)]
27. Cheng, Y.; Ren, J.; Wang, Z.; Mei, S.; Zhou, J. Keys Distributing Optimization of CP-ABE Based Access Control in Cryptographic Cloud Storage. *IEICE Trans. Inf. Syst.* **2012**, *95*, 3088–3091. [[CrossRef](#)]
28. Shi, E.; Perrig, A.; Doorn, L.V. BIND: A Fine-Grained Attestation Service for Secure Distributed Systems. In Proceedings of the 2005 IEEE Symposium on Security and Privacy, Oakland, CA, USA, 8–11 May 2005; pp. 154–168.
29. Ba, H.; Ren, J.; Wang, Z.; Zhou, H.; Li, Y.; Hong, T. User-policy-based dynamic remote attestation in cloud computing. *Int. J. Embed. Syst.* **2016**, *8*, 39–45. [[CrossRef](#)]
30. Berger, S.; Cáceres, R.; Goldman, K.A.; Perez, R.; Sailer, R.; van Doorn, L. vTPM: Virtualizing the Trusted Platform Module. In Proceedings of the 15th Conference on USENIX Security Symposium, Vancouver, BC, Canada, 31 July–4 August 2006.
31. Strasser, M. A Software-Based TPM Emulator for Linux. Semester Thesis, Eidgenössische Technische Hochschule Zürich (ETH Zürich), Zürich, Switzerland, 2004.
32. Strasser, M.; Stamer, H. A Software-Based Trusted Platform Module Emulator. In Proceedings of the 1st International Conference on Trusted Computing, Oslo, Norway, 23–25 June 2008; Springer: Berlin, Germany, 2008; pp. 33–47.
33. Dolev, D.; Yao, A.C. On the Security of Public Key Protocols. In Proceedings of the 22nd Annual Symposium on Foundations of Computer Science, Nashville, TN, USA, 28–30 October 1981; pp. 350–357.
34. Dolev, D.; Yao, A.C. On the Security of Public Key Protocols. *IEEE Trans. Inf. Theory* **1983**, *29*, 198–208. [[CrossRef](#)]
35. Ries, S. Extending Bayesian Trust Models Regarding Context-dependence and User Friendly Representation. In Proceedings of the 2009 ACM Symposium on Applied Computing, Chicago, IL, USA, 9–13 November 2009; pp. 1294–1301.
36. TrouSerS: The Open-Source TCG Software Stack. Available online: <http://trousers.sourceforge.net/> (accessed on 5 March 2018).
37. OpenSSL: Cryptography and SSL/TLS Toolkit. Available online: <https://www.openssl.org/> (accessed on 5 March 2018).
38. Advanced Crypto Software Collection. Available online: <http://acsc.cs.utexas.edu/cpabe/> (accessed on 5 March 2018).
39. Sahai, A.; Waters, B. Fuzzy Identity-based Encryption. In Proceedings of the 24th Annual International Conference on Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, 22–26 May 2005; Springer: Berlin, Germany, 2005; pp. 457–473.

40. Goyal, V.; Pandey, O.; Sahai, A.; Waters, B. Attribute-based Encryption for Fine-grained Access Control of Encrypted Data. In Proceedings of the 13th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, 30 October–3 November 2006; pp. 89–98.
41. Goyal, V.; Jain, A.; Pandey, O.; Sahai, A. Bounded Ciphertext Policy Attribute Based Encryption. In Proceedings of the 35th International Colloquium on Automata, Languages and Programming, Part II, Reykjavik, Iceland, 7–11 July 2008; Springer: Berlin, Germany, 2008; pp. 579–591.
42. Qiao, H.; Ren, J.; Wang, Z.; Ba, H.; Zhou, H. Compulsory Traceable Ciphertext-policy Attribute-based Encryption against Privilege Abuse in Fog Computing. *Future Gener. Comput. Syst.* **2018**, *88*, 107–116. [[CrossRef](#)]
43. Fan, Y.; Liu, S.; Tan, G.; Qiao, F. Fine-Grained Access Control Based on Trusted Execution Environment. *Future Gener. Comput. Syst.* **2018**. [[CrossRef](#)]
44. Anati, I.; Gueron, S.; Johnson, S.; Scarlata, V. Innovative Technology for CPU Based Attestation and Sealing. In Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy, Tel-Aviv, Israel, 23–24 June 2013; Volume 13.
45. Kaplan, D.; Powell, J.; Woller, T. AMD Memory Encryption. Available online: https://developer.amd.com/wordpress/media/2013/12/AMD_Memory_Encryption_Whitepaper_v7-Public.pdf (accessed on 5 March 2018).
46. Ba, H.; Zhou, H.; Ren, J.; Wang, Z. Runtime Measurement Architecture for Bytecode Integrity in JVM-Based Cloud. In Proceedings of the 36th IEEE Symposium on Reliable Distributed Systems, Hong Kong, China, 26–29 September 2017; pp. 262–263.
47. Garfinkel, T.; Rosenblum, M. A Virtual Machine Introspection Based Architecture for Intrusion Detection. In Proceedings of the 10th Annual Network and Distributed System Security Symposium, San Diego, CA, USA, 6–7 February 2003; Internet Society: Reston, VA, USA, 2003; pp. 191–206.
48. Shi, J.; Yang, Y.; Tang, C. Hardware Assisted Hypervisor Introspection. *SpringerPlus* **2016**, *5*, 647–669. [[CrossRef](#)] [[PubMed](#)]
49. Ren, J.; Liu, L.; Zhang, D.; Zhang, Q.; Ba, H. Tenants Attested Trusted Cloud Service. In Proceedings of the 9th IEEE International Conference on Cloud Computing, San Francisco, CA, USA, 27 June–2 July 2016; pp. 600–607.
50. Zhou, H.; Ba, H.; Ren, J.; Wang, Y.; Wang, Z.; Li, Y. Decoupling Security Services from IaaS Cloud Through Remote Virtual Machine Introspection. In Proceedings of the 10th International Conference on Security, Privacy, and Anonymity in Computation, Communication, and Storage, Guangzhou, China, 12–15 December 2017; Springer: Berlin, Germany, 2017; pp. 516–529.
51. Waters, B. Ciphertext-policy Attribute-based Encryption: An Expressive, Efficient, and Provably Secure Realization. In Proceedings of the 14th International Conference on Practice and Theory in Public Key Cryptography Conference on Public Key Cryptography, Taormina, Italy, 6–9 March 2011; Springer: Berlin, Germany, 2011; pp. 53–70.
52. Herranz, J.; Laguillaumie, F.; Ràfols, C. Constant Size Ciphertexts in Threshold Attribute-based Encryption. In Proceedings of the 13th International Conference on Practice and Theory in Public Key Cryptography, Paris, France, 26–28 May 2010; Springer: Berlin, Germany, 2010; pp. 19–34.
53. Chen, C.; Zhang, Z.; Feng, D. Efficient Ciphertext Policy Attribute-based Encryption with Constant-size Ciphertext and Constant Computation-cost. In Proceedings of the 5th International Conference on Provable Security, Xi'an, China, 16–18 October 2011; Springer: Berlin, Germany, 2011; pp. 84–101.
54. Attrapadung, N.; Herranz, J.; Laguillaumie, F.; Libert, B.; de Panafieu, E.; Ràfols, C. Attribute-Based Encryption Schemes with Constant-Size Ciphertexts. *Theor. Comput. Sci.* **2012**, *422*, 15–38. [[CrossRef](#)]
55. Orsini, G.; Bade, D.; Lamersdorf, W. Computing at the Mobile Edge: Designing Elastic Android Applications for Computation Offloading. In Proceedings of the 8th IFIP Wireless and Mobile Networking Conference, Munich, Germany, 5–7 October 2015; pp. 112–119.
56. Satyanarayanan, M. The Emergence of Edge Computing. *Computer* **2017**, *50*, 30–39. [[CrossRef](#)]

