Towards Secure Provenance-Based Access Control in Cloud Environments

Adam Bates, Ben Mood, Masoud Valafar, and Kevin Butler
Department of Computer and Information Science
University of Oregon, Eugene, OR
{amb,bmood,valafar,butler}@cs.uoregon.edu

ABSTRACT

As organizations become increasingly reliant on cloud computing for servicing their data storage requirements, the need to govern access control at finer granularities becomes particularly important. This challenge is increased by the lack of policy supporting data migration across geographic boundaries and through organizations with divergent regulatory policies. In this paper, we present an architecture for secure and distributed management of provenance, enabling its use in security-critical applications. *Provenance*, a metadata history detailing the derivation of an object, contains information that allows for expressive, policy-independent access control decisions. We consider how to manage and validate the metadata of a provenance-aware cloud system, and introduce protocols that allow for secure transfer of provenance metadata between end hosts and cloud authorities. Using these protocols, we develop a provenance-based access control mechanism for Cumulus cloud storage, capable of processing thousands of operations per second on a single deployment. Through the introduction of replicated components, we achieve overhead costs of just 14%, demonstrating that provenance-based access control is a practical and scalable solution for the cloud.

Categories and Subject Descriptors

C.2.0. [**Computer-Communication Networks**]: General — *Security and protection*; H.3.4. [**Information Systems**]: Information Storage and Retrieval — *Systems and Software*

General Terms

Security, Storage

Keywords

Provenance, Access Control, Secure Storage

CODASPY'13, February 18–20, 2013, San Antonio, Texas, USA.

1. INTRODUCTION

Cloud computing has become a fundamental part of the modern computing landscape. Organizations are moving computing to cloud environments for a number of reasons, from easy management of resources to leveraging the elasticity implicit in the cloud model. Along with this comes an increased reliance on outsourced cloud services, and thus concern over data security and management. These concerns are well-founded based on widely-publicized instances where cloud services have not provided stated security guarantees, or have been used as the launching point for attacks on other services [11].

For the cloud customer, a necessary step for managing data is access control. Today, cloud storage services provide explicit access control lists (ACL) for sharing or protecting data. Managing access control in this manner is time-consuming even on a single machine, and scales poorly in the cloud due to the presence of data migration. When crossing organizational boundaries, data may be subject to a new set of access policies that require the explicit creation of new access rules. To avoid the need for constant policy editing, it would be desirable if data carried with it the necessary information to make accurate access control decisions.

Another largely unaddressed concern is the attribution of data as it enters the cloud and moves around within this environment. This is an issue that many organizations have not faced to this point, and it addresses a characteristic inherent to the cloud: data can be replicated transparently and stored in multiple locations. Amazon, for example, has data centers not only in the eastern and western United States, but also in Europe, all representing the same cloud abstraction. This can be useful since data is now closer to more users and widespread redundancy is assured. However, issues such as compliance can be in conflict with this migration. For example, US regulations governing the protection of data differ from those laid out by the European Union's Directive 95/46/EC [27]. As a result, data created in the US may risk being out of compliance if it is silently transferred to European data centers. As another example, companies subject to US International Traffic in Arms Regulations (ITAR) must follow strict export restrictions such as preventing data from physically leaving the United States. While Amazon's GovCloud service is designed with these compliance issues in mind [2] customers not only pay more for the service but do not benefit from geographic redundancy within the US, since GovCloud is deployed separately from the rest of the Amazon cloud service and only within one data center. Because of these and associated con-

cerns regarding the jurisdiction of cloud data [19, 12], users of general-purpose clouds must be careful about uploading data since support for data migration, as well as other forms of controlling access and movement, is not provided, an issue that Peterseon et al. call data sovereignty [24].

This paper proposes that *data provenance* provides the necessary support to address these challenges. With provenance, information is kept about the origin and pedigree of data through documenting the initial conditions and variables under which data was created, and the conditions under which any modifications were made. It thus allows for more robust access decisions: by using the lineage of data, rather than its ephemeral present condition, we can change the *stateless* access control decision into a *stateful* decision.

While proposals exist to support provenance in the cloud [21], the issue of securing provenance in such environments has been left unaddressed. Providing this security is challenging for a variety of reasons. Securely generating and collecting provenance is difficult, as is transmitting it in a secure and trustworthy fashion. We propose additions to the cloud infrastructure that support the secure management of provenance for use in access decisions. *Policy enforcement points* (PEPs) collect provenance from hosts and act as the arbiters of whether to allow reading or writing to cloud storage. *Policy decision points* (PDPs) are system components responsible for ensuring provenance validity and policy compliance of data prior to allowing it into cloud storage. Finally, *provenance databases* store provenance for querying by PDP. Our implementation of the system is deployed in a real cloud environment. Our focus is the management and mediation of data and its provenance as it travels to and beyond organizational boundaries.

Our contributions can thus be summarized as follows:

- We design a system and associated protocols for securing and managing provenance in distributed cloud environments. This is the first system we are aware of that contains this functionality.
- We present an access control scheme based on the use of provenance attributes for policy decisions, and mechanisms for evaluating and optimizing these decisions in cloud environments.
- We evaluate our prototype implementation using the Cumulus cloud storage environment on the University of Oregon's ACISS science cloud, and show that provenance-based access controls can support thousands of operations per second on a single deployment with an imposed overhead cost as low as 14%.

Section 2 provides further background about data provenance and the Nimbus and Cumulus cloud systems. Section 3 describes the design of our system, including details on operation. Section 4 describes our prototype implementation, while in Section 5 we evaluate our design through intensive testing of overhead and scalability. Section 6 describes related work, and we conclude in Section 7.

2. BACKGROUND

2.1 Cloud Storage

Organizations use cloud storage systems to outsource their storage needs. Once the storage is leased, organizations can delegate read and write access to users. Based on granted permissions, users download files, process them on some host, and then upload the data back to the cloud. Nimbus is an open-source project for converting clusters of computers into an infrastructure-as-a-service cloud. Nimbus provides a storage system, Cumulus, which includes an authentication module that provides discretionary access control and a back-end that allows cloud providers to connect Cumulus to various storage devices. Cumulus is compatible with Amazon S3 REST API [1], exposing a set of commands for uploading, reading and updating files on the storage system.

Discretionary access control schemes in existing cloud storage systems suffer from many challenges. Once permissions are granted, cloud storage operations happen anonymously. Thus, organizations cannot keep track of who accesses their data. Moreover, organizations cannot track what happens to data after it exits the cloud boundary. They cannot answer questions such as: *Which application processed the data? What other datasets were used to generate current data? What were the host system conditions, such as security level and geographic location, when this data was processed?*

This lack of monitoring complicates the enforcement of data access. Organizations may wish to prevent access by hosts outside of their institutional boundaries, block hosts in certain geographical regions due to legal restrictions, or ensure that hosts accessing their data can provide certain integrity assurances.

2.2 Provenance

Provenance provides information on the actions taken on data from its genesis onwards. More specifically, provenance information describes the sources and the processing the data has undergone to get to its current state. Provenance information can be used to answer questions such as: *Which programs, applications, and datasets helped produce the data? In what environment was the data produced?*

Provenance information has a wide range of critical applications. In some cases, the value of data depends on its sources. Data provenance can identify the trusted or knowledgeable sources, making it useful as a mechanism for ensuring regulatory purposes. However, provenance information cannot be trusted unless its integrity is assured. Moreover, provenance must be protected differently than regular data. Recent work has explored scenarios where provenance requires different security than the data it describes [6]. Knowledge of a file's provenance may reveal secret details about company processes and environments, and may sometimes be more sensitive than the data itself. Unlike typical data, which can be created, modified or deleted, provenance metadata is immutable and append-only, requiring an efficient management infrastructure as it grows over time.

To address the challenge of collecting provenance from the host, we leverage the PASS system [20]. PASS overlays its own file system on top of raw storage and intercepts system calls in the kernel to generate provenance information. Whereas PASS provides a mechanism for collecting provenance on host systems, our purpose is to develop an infrastructure for securely sharing and applying provenance in distributed environments.

3. SYSTEM DESIGN

This work sets out to address the challenge of securely managing provenance metadata in the cloud. To this end,

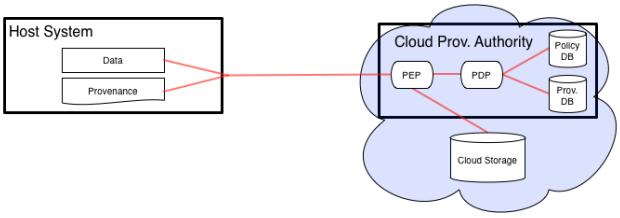


Figure 1: Cloud Provenance Authority system overview.

we assume that end hosts reliably generate complete and accurate provenance records, employing a trusted mechanism that collects and transmits provenance to the cloud. This challenge is orthogonal to our work and has previously been explored elsewhere. Most notably, the Hi-Fi system introduces a provenance reference monitor by leveraging the Linux security module framework [17, 25].

Our work considers an actively malicious adversary, in contrast to the benign environment considered in previous cloud provenance work [22, 21]. This attacker will attempt to manipulate provenance inputs to a security-critical cloud function such as access control or integrity verification. The attacker may attempt to alter the secrecy or integrity levels of an object by modifying its provenance chain, or discover hidden processing information by inspecting an object's ancestry. The attack surface considered in this work is the components comprising our Cloud Provenance Authority.

A further consideration in our work is protecting existing data from being tainted by untrusted hosts or organizations. This is a special case of integrity enforcement that is of major concern in the cloud environment. Shifts in organizational trust may occur often, causing frequent changes to our access control policy. For this reason, we wish for our system to be able to adapt to changes in policy quickly. We will soon show that provenance can be used in policy generation to quickly infer access control labels.

With this in mind, our system sets out to achieve the following goals:

- *Secure Distributed Provenance.* Previous work on secure provenance has focused on individual hosts [14, 13], while literature on managing provenance in distributed environments has left security unaddressed [21].
- *Fast Evaluation.* Many provenance use cases focus on infrequent, offline costs [22] or on data objects with limited ancestry [9]. Our architecture must provide efficient evaluation of remote data objects, even when they possess extensive ancestries.
- *Provenance-based Access Control.* We pursue an access control mechanism that accepts provenance as input in addition to subjects, objects and actions.

Due to space concerns, we have omitted a complete threat model and security analysis. Our design draws its security from previous work on signed provenance chains [14]. In this work, we present a scalable and efficient methodology for extending these constructions to distributed environments.

3.1 Design Overview

Our *Cloud Provenance Authority* architecture, shown in Figure 1, is a set of distributed components responsible for mediating access to cloud storage while collecting and managing provenance metadata. This architecture forms an overlay for existing cloud services, and thus does not require any modifications to cloud infrastructure. This system is data agnostic, allowing it to be applied to any level of cloud abstraction with only minor modification.

The core components of the Cloud Provenance Authority are the *Policy Enforcement Point (PEP)*, *Policy Decision Point (PDP)*, *Provenance Database*, and *Policy Database*. The PEP checks integrity of user requests and enforces access control on data according to an organization's policy. The PDP keeps organization policies and evaluates requests against the policies. The Provenance and Policy Databases store provenance information and security policies, respectively. The system operations are as follows:

- A user requests to store data on the cloud storage. The PEP then receives the request, approves it through the PDP, and stores the data on cloud storage.
- A user requests data stored on the cloud. The PEP receives the request, approves it through the PDP, and sends data to user.
- A user requests provenance from the Cloud Provenance Authority. The PEP receives the request and approves it through the PDP, which shares the provenance information for the PEP to return to the user.

1. $C \rightarrow PA:$ n_c, $Guid_c$, oid
2. $PA \rightarrow C:$ n_{pa}, $PC_{oid,t-1}$, $Sign[K^-_{pa}, PC_{oid,t-1}||n_c]$
3. $C \rightarrow PA:$ $Sign[K^-_c, PC_{oid,t}||n_{pa}]$, $PC_{oid,t}$, $Object$
4. $PA \rightarrow C:$ $Sign[K^-_{pa}, PC_{oid,t}]$

Table 1: Commitment protocol for write operation between Clients (C) and the Cloud Provenance Authority (PA).

n_i :	nonce issued by i
K^+_i, K^-_i :	i's private key and public key
$Hash(m)$:	digest of message m
$Sign[k, m]$:	signature of message m using key k
$Guid_i$:	global user identifier for user i
oid :	object identifier
$a\|\|b$:	concatenation of two strings, a and b
$Object$:	data object
PC_t :	t-th chain in provenance chain
PI_t :	provenance information of t-th version of the file

Figure 2: Summary of notations.

3.2 System Operation

3.2.1 Writing Data to Cloud Storage

When a user tries to write a file to cloud storage, a query consisting of the data and its provenance is sent to the Cloud

Provenance Authority. The PEP component receives the query and acts as a transparent proxy for all cloud storage operations. It checks the integrity of the request and sends it to the PDP, which evaluates it against a set of pre-defined policies. The PDP then informs the PEP of the outcome. If the operation is allowed, the PEP updates data in cloud storage on behalf of the user, transmits the provenance of the data to the PDP's provenance database, and sends an acknowledgement to the user.

To transmit the new chain to the cloud authority and provide mutual non-repudiability, the client follows a provenance commitment protocol. The protocol is described in Table 1 and uses the list of notations explained in Figure 2. In the first two steps of this protocol, the client and the cloud authority exchange initialization information. At step 3, the client sends provenance information with the write request to the Cloud Provenance Authority. The cloud sends back the signed operation to the client at step 4. This ensures mutual non-repudiability. The dialog between the cloud provenance authority and the client is mediated through the PEP.

Provenance integrity assurance is a critical component of our system design. Each time a write is committed to cloud storage, a new data version must be accounted for in the associated provenance records. Since we want to make sure that we can track the provenance information at each version of the data, we use a secure provenance chain introduced by Hasan et al. [14]. Equation 1 summarizes how the provenance chain is created at each step:

$$PC_t = Sign[K_u^-, hash(PI_t||PC_{t-1})] \quad (1)$$

Provenance chains prevent a variety of tampering attacks by tracking writes and securing the associated provenance. The provenance chain, signed by the user, connects the provenance information of a newly modified file to previous versions. Creating a chain of signed provenance commitments ensures that links are not removed, added, or altered.

3.2.2 Reading Data from Cloud Storage

The read operation from cloud storage is very similar to the write operation. When the PEP receives the query, it queries the PDP for an access control decision. The PDP retrieves the latest provenance information from the database and checks the information against the security policy. If the query is compatible with the policies, the PEP sends the file to the user.

In order to avoid losing track of the ancestry of a given file, each file has a reference back to the cloud. This reference consists of the cloud identification, file identification (used by the cloud storage) and the version of the file in the cloud. This information can inform an auditor where to find a file's provenance. The main advantage of such a reference is to address the problem of provenance growth [5].

3.2.3 Provenance Management and Retrieval

In order to create a scalable architecture for cloud provenance, a necessary first step is detaching provenance from the its associated data. There are two disadvantages for a system that stores the data and provenance in tandem. The first disadvantage of such a system is it makes provenance information visible to everyone who has access to data. On occasion, provenance can require different security than the data it describes [6].The second disadvantage is that provenance information can grow arbitrarily large as a result of the processing that occurs to data on a single host, hampering performance and scalability. Thus, in our architecture we do not transmit provenance with every file access. Instead, we entrust provenance storage to PDP components.

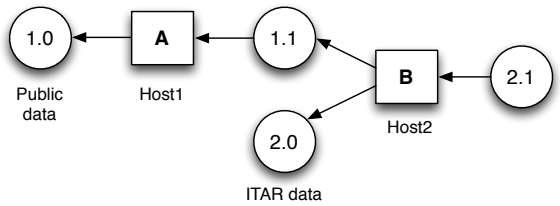


Figure 3: The provenance causality graph for artifact 2.1. Ancestors have been annotated with data labels.

In turn, this leads to another design question regarding entities with multiple storage servers: *If data retrieved from storage unit A is then stored on storage unit B, is the provenance information on A duplicated to B?* Our answer to this question is *no*. Instead, the provenance information remains in storage unit A, which permits B to access portions of provenance on demand.

This protocol works as follows: upon requesting provenance information, the user provides a chain of certificates. The first certificate is signed by the organization that owns the data, O_0. This certificate contains the delegation rights to another organization, O_1. The next chain contains the delegation rights to O_2, issued by O_1. The sequence of chains continues until one organization grants access to a user, U, as shown in Equation 2:

$$\begin{aligned} &Sign[K_{O_0}^-, Delegation, O_1] \\ &Sign[K_{O_1}^-, Delegation, O_2] \\ &Sign[K_{O_2}^-, read, U] \end{aligned} \quad (2)$$

In this manner, cloud provenance authorities can cooperate with each other to track data. For environments that require end user's to inspect provenance information, the query interface can be exposed to clients outside the cloud.

3.3 Provenance-Based Access Control

The Cloud Provenance Authority can be leveraged to perform security-critical functions, such as access control, in a manner that benefits from the rich contextual information that provenance provides. As described by Rosenthal et al. [26], attribute-based access control is superior to role-based access control for provenance due to improved scalability, flexibility, and the ability to modularize policy concerns (i.e., only one attribute needs to be inspected if a change in policy occurs). Attributes of data can be extracted and used to create labels for use in access control decisions. To extract all the attributes of the data, two properties should be assured: (1) the availability of complete data ancestry from original sources to its current form, and (2) the availability of all the necessary attributes of the sources. Our access control language is a simplified model of the provenance-based access control scheme presented by Cadenhead et al. [8].

Provenance-based labeling possesses two appealing characteristics: (1) the ability to dynamically adapt to sudden changes in policy, and (2) the ability to express access decisions based on a variety of attributes and levels of granularity. Consider the example shown in Figure 3. In this case, artifact 2.1 has been generated from the results of two

data sets, one of which is public data, and one of which is technical data related to defense and hence subject to ITAR regulations. In order to comply with ITAR, cloud nodes have the following simplified policy rule:

```
<Rule>
  <Operation>migrate-foreign</Operation>
   <Object>
    <Attribute Name="ITAR">
  </Object>
  <Result>deny</Result>
</Rule>
```

This rule states that migration of data to a cloud server outside of the US will be denied if the data contains the ITAR attribute. In Figure 3, we see that as host B combines artifacts 1.1 and 2.0, the corresponding provenance record will have the following entry:

```
<record>
  <record-type>INPUT</record-type>
  <record-data>
    <xref pnode="B" version="2.0" />
  </record-data>
  <record-type>INPUT</record-type>
  <record-data>
    <xref pnode="A" version="1.1" />
  </record-data>
</record>
```

and following the `XREF` to artifact 2.0 will uncover the `ITAR` attribute. As a result, the new record will not be allowed to migrate to foreign cloud centers. Host2 is now the provenance authority for artifact 2.1, but another party accessing the object can check the chain of records, which now includes artifact 2.1, artifact 2.0, and artifacts 1.1 and 1.0 hosted by Host1. This is a simplification of the operations, which would involve read and write restrictions based on location of the requesting party and rules defining the semantics of the `migrate-foreign` operation, but is meant to show that label inference is possible over a provenance attribute set. Policy can similarly be defined that conforms to Bell-LaPadula MLS confidentiality model [3], Biba integrity models [4], or supporting Chinese Wall policy [7] based on attributes of the provenance artifacts.

More formally, let us consider the provenance system as a state machine with subjects $S = \{s_1, s_2, \dots\}$, objects $O = \{o_1, o_2, \dots\}$, attributes $A = \{a_1, a_2, \dots\}$, actions $T = \{t_1, t_2, \dots\}$, values $V = \{v_1, v_2, \dots\}$, and states $\Sigma = \{\sigma_1, \sigma_2, \dots\}$. An access decision in the system will be based on the evaluation of $O \times A \times T \times \Sigma \rightarrow V$ where Σ represents the current state of the Cloud Provenance Authority as defined by the policy in place by the organization. Access control decisions are defined with the set $V = \{\text{allow, deny}\}$. Subjects and objects can be identified in terms of their attributes such that when subject s_1 with attribute set a_1 acts on object o_2 created by subject s_2 with attributes a_2, a new object o_3 is created with attributes $a_3 = a_1||a_2$. The access decision is contingent on the evaluation of the attributes associated with object o_3 so that when the access policy is evaluated, the result is $o_3 \times a_1||a_2 \times T \times \sigma_n \rightarrow \{\text{allow, deny}\}$. Note that the provenance record for o_3 will be a secure provenance chain as defined by Hasan and each object o_i and attribute set a_i is signed by its associated subject s_j (note that subjects can clearly sign arbitrary numbers of objects and attribute sets as they are created and modified). The attribute set a_3 is thus equivalent to $Sign[K^-_{s_1}, a_1]||Sign[K^-_{s_2}, a_2]$, ensuring that the attributes cannot be tampered with. Thus in our above example with the migration policy as dictated giving system state σ_i and action t_j = `migrate-foreign`, even if $o_1 \times a_1 \times t_j \times \sigma_i \rightarrow \{\text{allow}\}$, the newly created object will cause a policy decision $o_1 \times a_3 \times t_j \times \sigma_i \rightarrow \{\text{deny}\}$ due to a_3 receiving the `ITAR` tag from a_2.

Because we can reconstruct the complete ancestry of existing objects, we are free to make arbitrary modifications to our policy without loss of accuracy. A change in policy only requires the one-time cost of re-evaluating the provenance chains of existing artifacts. Unlike traditional labeling mechanisms, the history of modifications is kept as the provenance, making it clear by examining the metadata how the current label has been derived.

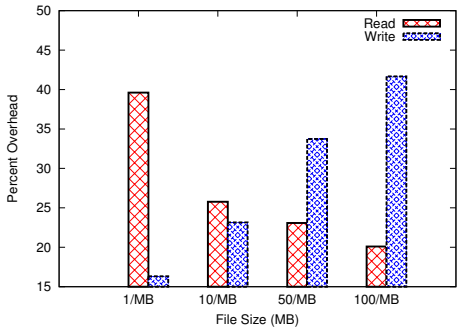
4. IMPLEMENTATION

We implemented the architecture described in Section 3 to run on separate virtual machines on an OpenStack KVM cloud service running on the University of Oregon's ACISS high performance cluster. Each component was provisioned with 2 vCPUs and 4GB memory. Each physical host was connected to a 1:1 provisioned Voltaire 8700 switch with fiber channel. The switch had 2 10 GbE trunks to a Cisco router that connected to the university network. Our client scripts connected to the Cloud Provenance Authority components over the campus network. They ran on a commodity PC containing an Intel 3.00GHz dual-core and running the Linux 2.6.38 kernel. The Cloud Provenance Authority was an overlay to a Cumulus storage server. The REST API of Cumulus is consistent with other cloud storage system APIs such as Amazon S3. Accordingly, our system can be implemented on any other compatible cloud storage systems.

The PEP, implemented in Python, acted as a transparent proxy to a Cumulus cloud store. The PEP was responsible for facilitating communication between with the client and PDP using the protocol outlined in Table 1. The PEP authenticated the client, verified provenance chain signatures, and mediated requests by checking policies with the PDP via remote procedure calls. The PEP was stateless, querying the PDP for user information and access decisions.

The PDP server, also implemented in Python, connected locally to a `sqlite3` database that stored policy, provenance, and client session information. The PDP reconstructed provenance chains by storing tuples of chain sequence numbers and file handle references for the provenance metadata. To evaluate our system we implemented three different access control mechanisms for the PDP: a basic whitelist-by-hostname policy, an MLS policy that evaluated provenance chains and labeled each object after the highest secrecy level in its history, and an optimizated MLS mechanism that cached decisions in order to avoid repetitive parsing of object ancestry.

A final component, a client host, submitted requests to the PEP to get and put data in the cloud store. These operations were either allowed or disallowed by the PDP based on the active policy. The client used PASS as the host provenance system. Provenance information was queried using Path Query Language (PQL) [15]. For the MLS mechanisms, attributes from the PASS provenance metadata were



(a) End-to-end system overhead.

Request	Step	1M	10M	50M	100M
Put Obj.	From User: Parse Req.	0.012	0.035	0.185	0.418
Put Obj.	To PDP: Get User Info	0.003	0.003	0.003	0.003
Put Obj.	To PDP: Check policy	0.004	0.004	0.004	0.005
Put Obj.	To Storage: Write Data	0.109	0.153	0.384	0.603
Put Obj.	To PDP: Append Prov.	0.004	0.004	0.004	0.005
Get Obj.	From User: Parse Req.	0.001	0.001	0.001	0.001
Get Obj.	To PDP: Check Policy	0.003	0.003	0.003	0.003
Get Obj.	To Storage: Read Data	0.015	0.089	0.385	0.757
Get Obj.	To User: Send Data	0.007	0.028	0.113	0.187

(b) Microbenchmarks for Policy Enforcement Point.

Figure 4: Performance cost of Cloud Provenance Authority architecture.

mapped to secrecy levels, and then submitted to the cloud authority in order to trigger access control decisions.

5. EVALUATION

5.1 System Overhead

Remote storage already has an associated delay, so our infrastructure needs to maintain a minimal performance footprint. We begin by investigating the overhead imposed by our system on write and read operations compared to basic cloud storage use. For this trial, a client host issued write and read requests of increasing size to a single PEP. The requests were checked against a simple whitelist policy at a single PDP, then processed. Figure 4a shows the percent overhead of our system relative to the raw cost of the Cumulus read and write operations. The overhead imposed decreases with the size of read operations, as low as 20%. Overhead increases with the size of write operations, with a low of 17.4% on small writes.

To better understand these costs, we recorded microbenchmarks for the various operations at the PEP. These results are displayed in Table 4b, with size-dependent fields highlighted. Because the PEP acts as a proxy to the storage, twice as much data transmission is required over raw storage access. The overhead of write operations increases with file size due to disproportionate growth in the time required to parse write requests. Without these transmission costs, the performance footprint is extremely small. Because the PDP is not involved in transmission of the actual object, it is underutilized for both the read and write operations.

Configuration	Avg. Speed (s)	Overhead
Direct Read	0.0928	0.0%
1 PEP Proxy	0.9512	90.2%
2 PEP Proxies	0.4901	81.1%
3 PEP Proxies	0.3551	73.0%
4 PEP Proxies	0.2643	64.8%
5 PEP Proxies	0.2205	14.0%

Table 2: Read speeds with additional PEP components.

5.2 Distributed Enforcement Points

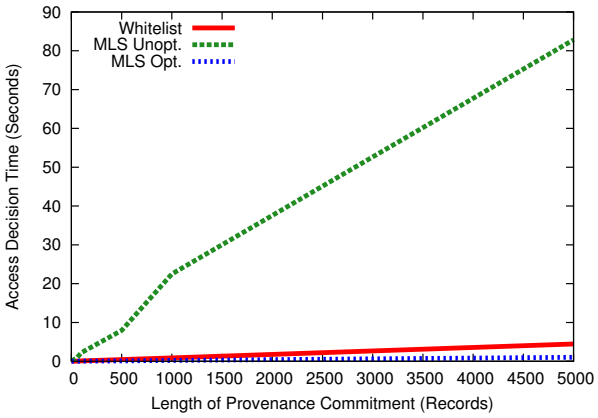
While the performance cost imposed by redundant transmission is considerable, it was necessary to our overlay design to avoid requiring special changes to PASS or the Amazon API. Fortunately, overhead cost can be easily decreased by using a distributed set of PEPs to process user requests. As noted in Section 5.1, a single PDP is underutilized in a deployment featuring a single PEP, allowing for the distributing of PEP tasks until either cloud storage or the PDP form a new system bottleneck. In a new trial, a single client executed batches of 100 reads of a 10/MB file with up to 5 PEPs running as separate instances within a university cloud service. Table 2 shows the results of this trial. As the system becomes more distributed, the per-request cost of the batch operations decreases by a factor of 4, resulting in an overhead as low as 14%.

5.3 Provenance Size

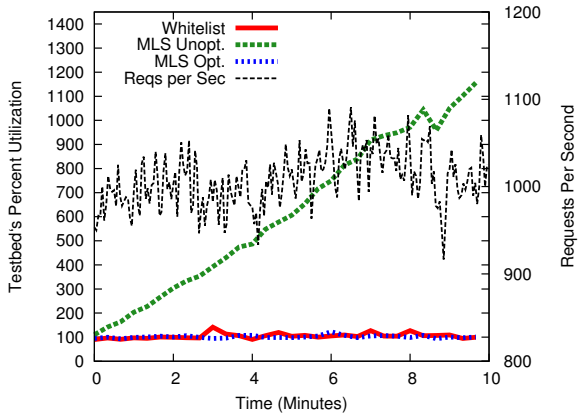
Another factor contributing to the duration of write transactions is the size of the provenance information that is attempting to be committed. A provenance chain's immutable nature can lead to unwieldy growth in the number of records, leading to considerable overhead on write transactions if large amounts of provenance has been stored locally or if new provenance is being migrated into storage. This cost is entirely separate from the size of the actual data. To analyze the impact of provenance growth on system performance, we repeated the previous trial for a single object with increasingly large provenance chains. Under three different access control mechanisms, we measured the speed with which the PDP arrived at an access decision. Figure 5a depicts the results. The unoptimized MLS policy, which re-parses the entire object ancestry with each request, became prohibitively expensive at large provenance sizes. In contrast, the whitelist and memoized MLS policies need only parse the newly appended provenance records.

5.4 Extreme Workload

To assess our infrastructure's ability to handle real-world workloads, we tested each of our access control mechanisms against the 1998 World Soccer Cup website trace. Envisioning a frequent-read-infrequent-write scenario, we mapped 99.8% of the HTTP requests in the trace to GET requests for our file, and the other remaining 0.2% to PUT requests. We selected a 10 minute window of 606,060 requests for analysis. Using the 5 PEP deployment from Section 5.2, we played the trace through one second at a time and observed the utilization level required for our infrastructure to handle the load in real time. To better isolate our system's performance, the PEPs transmitted empty acknowledgement messages to the user instead of communicating with Cumulus storage. All of



(a) Provenance chain size effect on PDP response times.



(b) Utilization during 1998 World Cup trace web requests.

Figure 5: Performance of provenance-based access control mechanisms.

the other system functions, including provenance migration and evaluation, were active for the experiment.

The performance of each mechanism is pictured in Figure 5b. The system load under the different access controls appears on the primary y-axis, while the trace workload in requests per second appear on the secondary y-axis. The performance of the unoptimized MLS mechanism, which is dependent on length of the object's ancestry, degrades over time. The optimized MLS mechanism was nearly able to handle the entire workload in real time, averaging 100.2% utilization over the course of the trial (i.e. 12 seconds of total delay were observed). Even on a modest deployment using a single centralized provenance database, our infrastructure handles thousands of access decisions per second.

6. RELATED WORK

Provenance systems have been well studied and surveyed in scientific contexts [28, 10, 18]. Provenance information can be recorded at various levels of abstraction. Application-level provenance systems collect provenance at the semantic level of the applications [30], while provenance systems for workflow engines record provenance at the level of workflow stages [29]. Provenance systems can also be implemented at the kernel level by intercepting system calls and recording them as provenance information. The provenance-aware storage system (PASS) [20] that we integrate into our system leverages the kernel-level approach. Muniswamy-Reddy et al. [22, 21] were the first to consider the challenges of adopting provenance to the cloud. They leveraged Amazon's S3 data storage to extend their host level provenance collector (PASS) [20] to the cloud environment. We extend this work by developing a distributed provenance management infrastructure under a malicious threat model.

During the past few years, several studies have recognized the importance of securing provenance [19, 13, 17]. Braun et al. [6] discuss that provenance requires its own security model that differs from regular data. Hasan et al. [14] consider a scheme for securely collecting and auditing provenance records with a focus on preventing undetected rewriting of provenance history. We use their approach to provide integrity for provenance information in the cloud. Rosenthal et al. [26] discuss an attribute-based access control scheme for provenance information. Their emphasis is on determining access based on provenance attributes. In our work, the permission to access provenance information is granted by the organization that owns the data. Another line of work by Ni et al. [23] modifies the provenance records to enable finer-grained policies for accessing provenance information. Given the significantly different features of cloud environment, our main target in this paper is scalability.

Ensuring that provenance metadata is properly collected and securely communicated makes certain demands of host systems. McDaniel et al. [17, 25] propose addressing the challenge of provenance mediation through *provenance monitors*. Their scheme exhibits reference monitor guarantees by preventing circumvention of provenance recording mechanisms, assuring the trustworthiness of collected information. Trusted computing, particularly trusted platform modules (TPMs) and attested boot mechanisms, provide a means of supporting this bottom-up trust model [16]. However, implementing such systems in the cloud requires support (e.g., remote attestation) from cloud providers. The secure provenance system that we propose in this paper can work on normal services offered by cloud providers.

7. CONCLUSION

This paper has introduced an architecture for managing provenance metadata in the cloud and provides a means for using provenance as the basis for an attribute-based access control system suitable for enforcing organizational security policies. Through the use of provenance to enforce access control, we effectively shift the access control paradigm from a stateless decision made strictly on the current state of data, to a stateful decision based on the data's origin and lineage. Our prototype implementation shows that such a system is also practical, supporting thousands of operation per second on a modestly specified deployment. These investigations are a preliminary exploration into the myriad challenges faced when securing provenance at scale across organizational boundaries and within the cloud.

Acknowledgemnts

We thank the members of the OSIRIS Lab at the University of Oregon for their helpful comments and suggestions, and Chris Hoge and the ACISS staff for their assistance and support. This work is supported by NSF grant CNS-1118046.

8. REFERENCES

[1] Amazon. Amazon Simple Storage Service (S3), 2011.

[2] Amazon. Amazon Web Services: Risk and Compliance. `http://media.amazonwebservices.com/AWS_Risk_and_Compliance_Whitepaper.pdf`, 2012.

[3] D. Bell and L. LaPadula. Secure Computer Systems: Mathematical Foundations and Model. Technical Report M74-244, MITRE Corporation, Bedford, MA, 1973.

[4] K. J. Biba. Integrity Considerations for Secure Computer Systems. *Proceedings of the 4th Annual Symposium on Computer Architecture*, 5(7):135–140, 1977.

[5] U. Braun, S. Garfinkel, D. A. Holland, K. Muniswamy-Reddy, and M. Seltzer. Issues in Automatic Provenance Collection. In *Proceedings of the 2006 International Provenance and Annotation Workshop*, Chicago, Illinois, May 2006.

[6] U. Braun, A. Shinnar, and M. Seltzer. Securing Provenance. In *Proceedings of the USENIX Workshop on Hot Topics in Security (HotSec)*, San Jose, CA, 2008.

[7] D. F. C. Brewer and M. J. Nash. The Chinese Wall Security Policy. In *Proceedings of the 10th IEEE Symposium on Security and Privacy*, Oakland, CA, USA, 1989.

[8] T. Cadenhead, V. Khadilkar, M. Kantarcioglu, and B. Thuraisingham. A Language for Provenance Access Control. In *CODASPY '11: Proceedings of the first ACM Conference on Data and Application Security and Privacy*, pages 133–144, San Antonio, TX, USA, 2011. ACM Press.

[9] M. Dietz, S. Shekhar, Y. Pisetsky, A. Shu, and D. S. Wallach. QUIRE: Lightweight Provenance for Smart Phone Operating Systems. In *Proceedings of the 20th USENIX Security Symposium*, 2011.

[10] J. Freire, D. Koop, E. Santos, and C. T. Silva. Provenance for Computational Tasks: A Survey. *Computing in Science and Engineering*, 10(3):11–21, 2008.

[11] J. Galante, O. Karif, and P. Alpeyav. Sony's Network Breach Shows Amazon Cloud's Appeal for Hackers. *Bloomberg News*, 16 May 2011.

[12] R. Gellman. Privacy in the Clouds: Risks to Privacy and Confidentiality from Cloud Computing. *World Privacy Forum*, pages 1–26, 2009.

[13] R. Hasan, R. Sion, and M. Winslett. Introducing Secure Provenance: Problems and Challenges. In *Proceedings of the 2007 ACM Workshop on Storage Security and Survivability*, StorageSS '07, pages 13–18, New York, NY, USA, 2007. ACM.

[14] R. Hasan, R. Sion, and M. Winslett. The Case of the Fake Picasso: Preventing History Forgery with Secure Provenance. In *FAST '09: Proceedings of the 7th USENIX Conference on File and Storage Technologies*, 2009.

[15] D. A. Holland and M. Seltzer. PQL - Path Query Language. http://www.eecs.harvard.edu/syrah/pql/, 2011.

[16] J. Lyle and A. Martin. Trusted Computing and Provenance: Better Together. In *TaPP '10: Proceedings of the 2nd USENIX Workshop on the Theory and Practice of Provenance*, Berkeley, CA, USA, 2010.

[17] P. McDaniel, K. Butler, S. McLaughlin, R. Sion, E. Zadok, and M. Winslett. Towards a Secure and Efficient System for End-to-End Provenance. In *TaPP '10: Proceedings of the 2nd USENIX Workshop on the Theory and Practice of Provenance*, 2010.

[18] L. Moreau, B. Ludäscher, I. Altintas, et al. Special Issue: The First Provenance Challenge. *Concurrency and Computation: Practice and Experience*, 20(5):409–418, 2008.

[19] M. Mowbray. The Fog over the Grimpen Mire: Cloud Computing and the Law. *Scripted Journal of Law, Technology and Society*, 6(1), Apr. 2009.

[20] K. Muniswamy-Reddy, D. A. Holland, U. Braun, and M. Seltzer. Provenance-Aware Storage Systems. In *Proceedings of the 2006 USENIX Annual Technical Conference*, 2006.

[21] K. Muniswamy-Reddy, P. Macko, and M. I. Seltzer. Provenance for the Cloud. In *FAST '10: Proceedings of the 8th USENIX Conference on File and Storage Technologies*, 2010.

[22] K. Muniswamy-Reddy and M. Seltzer. Provenance as First-Class Cloud Data. In *Proceedings of the ACM ACM SIGOPS International Workshop on Large Scale Distributed Systems and Middleware (LADIS)*, 2009.

[23] Q. Ni, S. Xu, E. Bertino, R. Sandhu, and W. Han. An Access Control Language for a General Provenance Model. In *Secure Data Management*, Aug. 2009.

[24] Z. N. J. Peterson, M. Gondree, and R. Beverly. A Position Paper on Data Sovereignty: The Importance of Geolocating Data in the Cloud. In *HotCloud'11: Proceedings of the 3rd USENIX Workshop on Hot Topics in Cloud Computing*, June 2011.

[25] D. Pohly, S. McLaughlin, P. McDaniel, and K. Butler. Hi-Fi: Collecting High-Fidelity Whole-System Provenance. In *Proceedings of the 2012 Annual Computer Security Applications Conference*, ACSAC '12, Orlando, FL, USA, 2012.

[26] A. Rosenthal, L. Seligman, A. Chapman, and B. Blaustein. Scalable Access Controls for Lineage. In *TaPP '09: Proceedings of the 1st USENIX Workshop on the Theory and Practice of Provenance*, San Francisco, CA, USA, 2009.

[27] G. Shaffer. Globalization and Social Protection: The Impact of EU and International Rules in the Ratcheting Up of US Data Privacy Standards. *Yale Journal of International Law*, 25:1–88, 2000.

[28] Y. Simmhan, B. Plale, and D. Gannon. A Survey of Data Provenance in e-Science. *ACM SIGMOD Record*, 34(3):31–36, 2005.

[29] M. Szomszor and L. Moreau. Recording and Reasoning over Data Provenance in Web and Grid Services. In *International Conference on Ontologies, Databases and Applications of SEmantics (ODBASE'03)*, volume 2888, pages 603–620, 2003.

[30] J. Widom. Trio: A System for Integrated Management of Data, Accuracy, and Lineage. Technical Report 2004-40, Stanford InfoLab, Aug. 2004.