Policy-Sealed Data: A New Abstraction for Building Trusted Cloud Services

Nuno Santos¹, Rodrigo Rodrigues², Krishna P. Gummadi¹, Stefan Saroiu³
MPI-SWS¹, CITI / Universidade Nova Lisboa², Microsoft Research³
Managing the Cloud is Complex & Error-Prone

Is my data properly managed?

Cloud software admins. can compromise customers’ data
1. Newer hypervisors can offer protection from SW admins.
   
   - e.g., nested virtualization: CloudVisor [SOSP ‘11], Credo [MSR-TR]

2. Trusted computing can attest cloud node runs “correct” hypervisor
   
   - Trusted Platform Module (TPM)

But, TPMs alone ill-suited for the cloud
TPMs Alone Are Ill-Suited for the Cloud

1. **Stifle VM and data migration across cloud nodes**
   - TPMs root-of-trust not transferable from one node to another

2. **Cloud providers hesitant to reveal low-level cloud details**
   - TPMs abstractions can reveal node’s identity and details of the node’s entire software stack

3. **Commodity TPMs can hinder the cloud’s ability to scale**
   - TPMs’ poor performance may introduce bottlenecks
Our Contributions

1. **Policy-sealed data abstraction**
   - Data is handled only by nodes satisfying customer-chosen policy
   - Examples:
     - Handle data only by nodes running CloudVisor
     - Handle data only by nodes located in the EU

2. **Use attribute-based encryption (CP-ABE) to implement abstraction efficiently**
   - Binds policies and node attributes to node configurations
   - Ciphertext-Policy Attribute-Based Encryption [Bethencourt07]

Excalibur incorporates both contributions
Excalibur Addresses TPM Limitations in Cloud

Policy-sealed data
- Enables flexible data migration across cloud nodes
  - Customer data accessible to any node that satisfies the customer policy

Attribute-based encryption
- Hides node’s identities and low-level details of the software
  - Only high-level attributes are revealed
- Masks TPMs’ poor performance
  - Enforcing policies does not require direct calls to TPMs
Outline

- Introduction
- Threat model
- Policy-sealed data
- Design
  - Monitor
  - CP-ABE
- Evaluation
## Threat Model

<table>
<thead>
<tr>
<th>The attacker can...</th>
<th>The attacker cannot...</th>
</tr>
</thead>
<tbody>
<tr>
<td>configure nodes remotely</td>
<td>perform physical attacks</td>
</tr>
<tr>
<td>reboot nodes</td>
<td>e.g., scrape TPMs to learn its secrets</td>
</tr>
<tr>
<td>install software platform</td>
<td>compromise system’s TCB</td>
</tr>
<tr>
<td>access disk</td>
<td>monitor</td>
</tr>
<tr>
<td>eavesdrop network</td>
<td>secure hypervisor</td>
</tr>
<tr>
<td></td>
<td>compromise CP-ABE</td>
</tr>
</tbody>
</table>
Outline

- Introduction
- Threat model
- Policy-sealed data
- Design
  - Monitor
  - CP-ABE
- Evaluation
Policy-Sealed Data

**Seal**
encrypt and bind data to policy

**Unseal**
decrypt data iff node meets policy

Seal to:
visor = “secure visor”

Customer

Provider

Hypervisors
- Secure
- Commodity
Policy-Sealed Data: Attributes & Policies

- **Node configurations expressed as set of attributes**
  - Attributes mapped to nodes’ identities and software config
    - node id $\rightarrow$ hardware attributes
    - software config $\rightarrow$ software attributes

- **Customers select trusted node configurations in policies**
  - Logic expressions over attributes

```python
Node Attributes

service = "EC2"
hypervz = "CloudVisor"
version = "1"
country = "Germany"
zone = "z1"
```

```python
Data Policy

service = "EC2"
and
hypervz = "CloudVisor"
and
version >= "1"
and
(country = "Germany"
or
country = "UK")
```
Outline

- Introduction
- Threat model
- Policy-sealed data
- Design
  - Monitor
  - CP-ABE
- Evaluation
Excalibur Architecture

- Check node configurations
  - Monitor attests nodes in background

- Scalable policy enforcement
  - CP-ABE operations at client-side lib
Monitor goals:

- Track node ids + TPM-based attestations
  - Hides low-level details from users

- Track nodes’ attributes that cannot be attested via today’s TPMs
  - e.g., nodes’ locations (EU vs. US)

- Form the cloud’s root of trust
  - Customers only need to attest the monitor’s software configuration
Attribute-based Encryption Is Key to Scalability

Customers seal data to a policy with a CP-ABE encryption key.
Once each node attests its configuration, monitor hands CP-ABE decryption key.

- Ciphertext-Policy Attribute-Based Encryption [Bethencourt07]
Outline

- Introduction
- Threat model
- Policy-sealed data
- Design
  - Monitor
  - CP-ABE
- Evaluation
Methodology

- Two questions:
  - What is the overhead of policy-sealed data?
  - Is the monitor a scalability bottleneck?

- Implemented cloud service akin to EC2
  - Based on Eucalyptus / Xen cloud platform
  - Supports location attribute
  - Interposed seal / unseal in VM management operations

- Testbed: single monitor and five nodes
  - Intel Xeon, 2.83Ghz 8-core CPU, 1.6 GB RAM, TPM v1.2
What Is the Overhead of Seal / Unseal?

Overhead of CP-ABE in Eucalyptus / Xen platform

CP-ABE’s overhead could be significant
However, VM operations are infrequent
Is the Monitor a Scalability Bottleneck?

- **Monitor can attest a large number of nodes**
  - Max throughput: 630 attestation-verifications/sec
  - E.g., 10K node cluster attests in ~15 seconds

- **Monitor can serve many attestation requests from customers**
  - Max throughput: 4800 attestation-requests/sec
  - Increases throughput of standard TPM attestation
    - Batches multiple attestation requests into single TPM call
  - Speedup orders of magnitude over standard TPM attestation
Conclusions

- **Excalibur** overcomes TPM’s limitations in the cloud

- **Policy-sealed data**: new trusted computing primitive
  - Flexible sealed storage
  - Reduce overexposure

- **CP-ABE** makes Excalibur scale
  - Masks low performance of TPMs

- Evaluation indicates that the system is practical