Bolt: I Know What You Did Last Summer... In the Cloud

Christina Delimitrou\textsuperscript{1} and Christos Kozyrakis\textsuperscript{2}

\textsuperscript{1}Cornell University, \textsuperscript{2}Stanford University

ASPLOS – April 12\textsuperscript{th} 2017
Problem: cloud resource sharing hides security vulnerabilities
- Interference from co-scheduled apps $\rightarrow$ leaks app characteristics
- Enables severe performance attacks

Bolt: adversarial runtime in public clouds
- Transparent app detection (5-10sec)
- Leverages practical machine learning techniques
- DoS $\rightarrow$ 140x increase in latency
- User study: 88% correctly identified applications
- Resource partitioning is helpful but insufficient
Motivation

App1

App2

Amazon Web Services

Windows Azure

Google Cloud Platform
Motivation
Motivation

[Diagram showing containers and memory capacity for App1 and App2.]

- Containers
- Memory capacity

[Logos of Amazon Web Services, Windows Azure, and Google Cloud Platform.]
Motivation

containers

memory capacity

storage capacity/bw

App1

App2

Amazon Web Services

Windows Azure

Google Cloud Platform
Motivation

- App1
- App2

- Containers
- Memory capacity
- Network bw
- Storage capacity/bw

Amazon Web Services

Windows Azure

Google Cloud Platform
Motivation
Motivation
Motivation

Not all isolation techniques available
Not all used/configured correctly
Not all scale well
Mem bw/core resources not isolated
Bolt

- **Key idea**: Leverage lack of isolation in public clouds to infer application characteristics
  - Programming framework, algorithm, load characteristics

- **Exploit**: enable practical, effective, and hard-to-detect performance attacks
  - DoS, RFA, VM pinpointing
  - Use app characteristics (sensitive resource) against it
  - Avoid CPU saturation → hard to detect
Threat Model

- Impartial, neutral cloud provider
- Active adversary but no control over VM placement
Bolt

1. Contention injection
2. Interference measurement
3. App inference
4. Custom contention kernel
5. Performance attack
1. Contention Measurement

- Set of contentious kernels (iBench)
  - Compute
  - L1/L2/L3
  - Memory bw
  - Storage bw
  - Network bw
  - (Memory/Storage capacity)

- Sample 2-3 kernels, run in adversarial VM
- Measure impact on performance of kernels vs. isolation
2. Practical App Inference

- Infer resource pressure in non-profiled resources
  - Sparse → dense information
  - SGD (Collaborative filtering)

- Classify unknown victim based on previously-seen applications
  - Label & determine resource sensitivity
  - Content-based recommendation

Hybrid recommender
Big Data to the Rescue

1. Infer pressure in non-profiled resources
   - Reconstruct sparse information
   - Stochastic Gradient Descent (SGD), $O(mp_k)$

\[
\begin{align*}
\begin{bmatrix}
    r_1 & r_2 & r_3 & \cdots & r_N \\
    a_{11} & 0 & 0 & \cdots & a_{1N} \\
    0 & a_{22} & 0 & \cdots & 0 \\
    \vdots & \vdots & \vdots & \ddots & \vdots \\
    a_{M1} & 0 & a_{M3} & \cdots & 0
\end{bmatrix} & \rightarrow \\
\begin{bmatrix}
    r_1 & r_2 & r_3 & \cdots & r_N \\
    a_{11} & a_{12} & a_{13} & \cdots & a_{1N} \\
    a_{21} & a_{22} & a_{23} & \cdots & a_{2N} \\
    \vdots & \vdots & \vdots & \ddots & \vdots \\
    a_{M1} & a_{M2} & a_{M3} & \cdots & a_{MN}
\end{bmatrix}
\end{align*}
\]
Big Data to the Rescue

2. Classify and label victims
   - Weighted Pearson Correlation Coefficients
   - Output: distribution of similarity scores to app classes

\[
\begin{align*}
\begin{bmatrix}
1 & a_{12} & a_{13} & \cdots & a_{1N} \\
a_{21} & a_{22} & a_{23} & \cdots & a_{2N} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
a_{M1} & a_{M2} & a_{M3} & \cdots & a_{MN}
\end{bmatrix}
\end{align*}
\]

Bolt

Data

Hadoop SVM: 65%
Spark ALS: 21%
memcached: 11%
Inference Accuracy

- 40 machine cluster (420 cores)
- **Training apps:** 120 jobs (analytics, databases, webservers, in-memory caching, scientific, js) → high coverage of resource space
- **Testing apps:** 108 latency-critical webapps, analytics
- No overlap in algorithms/datasets between training and testing sets

<table>
<thead>
<tr>
<th>Application class</th>
<th>Detection accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>In-memory caching (memcached)</td>
<td>80%</td>
</tr>
<tr>
<td>Persistent databases (Cassandra, MongoDB)</td>
<td>89%</td>
</tr>
<tr>
<td>Hadoop jobs</td>
<td>92%</td>
</tr>
<tr>
<td>Spark jobs</td>
<td>86%</td>
</tr>
<tr>
<td>Webservers</td>
<td>91%</td>
</tr>
<tr>
<td><strong>Aggregate</strong></td>
<td><strong>89%</strong></td>
</tr>
</tbody>
</table>
3. Practical Performance Attacks

1. Determine the resource bottleneck of the victim
2. Create custom contentious kernel that targets critical resource(s)
3. Inject kernel in Bolt

- Several performance attacks (DoS, RFAs, VM pinpointing)
- Target specific, critical resource → low CPU pressure
3. Practical DoS Attacks

- Launched against same 108 applications as before
- On average 2.2x higher execution time and up to 9.8x
- For interactive services, on average 42x increase in tail latency and up to 140x

- Bolt does not saturate CPU
- Naïve attacker gets migrated
User Study

- 20 independent users from Stanford and Cornell

- Cluster
  - 200 EC2 servers, c3.8xlarge (32vCPUs, 60GB memory)

- Rules:
  - 4vCPUs per machine for Bolt
  - All users have equal priority
  - Users use thread pinning
  - Users can select specific instances
  - Training set: 120 apps incl. analytics, webapps, scientific, etc.
Accuracy of App Labeling

53 app classes (analytics, webapps, FS/OS, HLS/sim, other...)

Correct app labels 63%
Accuracy of App Characterization

Performance attack results in the paper

Correct app characteristics 88%
The Value of Isolation

- Need more scalable, fine-grain, and complete isolation techniques
Conclusions

- **Bolt**: highlight the security vulnerabilities from lack of isolation
  - Fast detection using online data mining techniques
  - Practical, hard-to-detect performance attacks
  - Current isolation helpful but insufficient

- **In the paper:**
  - Sensitivity to Bolt parameters
  - Sensitivity to applications and platform parameters
  - User study details
  - More performance attacks (resource freeing, VM pinpointing)
Questions?

- Bolt: highlight the security vulnerabilities from lack of isolation
  - Fast detection using online data mining techniques
  - Practical, hard-to-detect performance attacks
  - Current isolation helpful but insufficient

- In the paper:
  - Sensitivity to Bolt parameters
  - Sensitivity to applications and platform parameters
  - User study details
  - More performance attacks (resource freeing, VM pinpointing)
Evolving Applications

- Cloud applications change behavior
- Users use the same cloud resources for several apps over time
- Bolt periodically wakes up, checks if app profile has changed; if so, reprofile & reclassify
Within a framework, dataset and choice of algorithm affect resource requirements.

Bolt matches a new unknown application to apps in a framework by distinguishing their resource needs.