Opportunities and Challenges in Mobile and Cloud Computing

Ben Y. Zhao
University of California, Santa Barbara
http://current.cs.ucsb.edu
(w/ Krishna Puttaswamy, Chris Kruegel)
Privacy and Confidentiality Challenges in Mobile and Cloud Computing

Ben Y. Zhao
University of California, Santa Barbara
http://current.cs.ucsb.edu
(w/ Krishna Puttaswamy, Chris Kruegel)
A Critical Mass of Mobile Devices

- Programmable mobile application platforms
  - Cell market dominated by smartphones
    - GPS, WiFi/3G, open application platforms
A LOOK AT THE SMARTPHONE INDUSTRY’S MARKET GROWTH

2Q09 Sales (units) 2Q08 Sales (units) 2Q09 Market Share (%) 2Q08 Market Share (%)

Nokia: 18.441M 15.297M 45.0% 47.4%
Research in Motion: 7.678M 5.594M 18.7% 17.3%
Apple: 5.434M 892M 13.3% 2.8%
HTC: 2.471M 1.330M 6.0% 4.1%
Fujitsu: 1.249M 1.071M 3.0% 3.3%
Others: 5.688M 8.085M 13.9% 25.1%

Share increased from previous year
Share decreased from previous year

Sources: AppleInsider.com and Gartner
A Critical Mass of Mobile Devices

Programmable mobile software platforms
- Cell market dominated by smartphones
  - GPS, WiFi/3G, open application platforms

Open mobile app platforms
- iOS/iPad/iPhone: 200K apps
- Android: 50K apps
- Application downloads to go from 7 billion in 2009 to ~50 billion in 2012
- Account for 17.5B in revenue in ‘12

300M units in 2011 (iSuppli)
Mobile Applications and the Cloud

- Killer apps in the near future?
  - No one “knows,” but some guesses…

- Today’s mobile devices
  - Still resource constrained
  - ~1Ghz CPU, 512MB RAM, 1300-1400mAh, 16GB storage
  - Offload computation to hosted servers

- Benefits of third-party clouds
  - On-demand use of resources (or elasticity)
  - Zero overhead for maintenance or management
  - Low costs due to economies of scale
  - Natural fit for mobile applications?

<table>
<thead>
<tr>
<th>Top Mobile Applications ’12 (Gartner)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Money Transfer ($ via SMS)</td>
</tr>
<tr>
<td>2. Location-based Services</td>
</tr>
<tr>
<td>(512M users in ’12 predicted)</td>
</tr>
<tr>
<td>3. Mobile Search</td>
</tr>
<tr>
<td>4. Mobile Browsing</td>
</tr>
<tr>
<td>5. Mobile Health Monitoring</td>
</tr>
<tr>
<td>6. Mobile Payment</td>
</tr>
<tr>
<td>7. Near-field Communication Services</td>
</tr>
<tr>
<td>8. Mobile Advertising</td>
</tr>
<tr>
<td>9. Mobile Instant Messaging</td>
</tr>
<tr>
<td>10. Mobile Music</td>
</tr>
</tbody>
</table>

Also notable: Augmented Reality
Location-based Applications

- Buddy Notification
- Social Recommendation for Dining or Shopping Nearby
- Location-Based Gaming with Friends
- Location-Based Reminders for Family and Friends

- Small companies, low resources, need scalability/growth
- Ideal fit for the cloud
So What is Missing?
Major Obstacle: Concerns over Security

- A business’ most valuable asset is data
  - Operational data
  - Intellectual property
  - Users’ personally identifiable information (PII)
    E.g.: FourSquare and pleaserobme.com

- No data confidentiality on the cloud
  - Hosted model requires storing data on cloud
  - Data outside owner organization’s control

- Multiple sources of data leaks
  - Operator errors, bugs in the cloud
  - Malicious applications leveraging multi-tenancy
Real-World Breaches and Leaks

Google Docs leaks private user data online
Flaw made documents available to unauthorised users
Iain Thomson in San Francisco
ymnet.com, 10 Mar 2009

Hackers find a home in Amazon's EC2 cloud
Security researchers discover the Zeus password-stealing botnet running on Amazon's EC2 cloud computing servers
By Robert McMillian | IDG News Service

Vulnerability Seen in Amazon's Cloud-Computing
New research reveals how to find would-be victims within cloud hardware.
By David Talbot

Oops! Private Facebook e-mails sent to wrong people
MIKE LUTTRELL | Thu 25th Feb 2010, 09:10 pm
#facebook #glitch

Ben Zhao, UC Santa Barbara 6/14/10
Implications

- 2009 Survey of 502 global C-level executives (17 countries)
  - 71% believe cloud is a real technology option
  - 80+% believe internal IT systems too expensive
  - 65% (worldwide) and 80% (US) believe cloud can cut costs
  - 5-1 ratio: trust internal systems over cloud-based
  - 80+% do not plan to integrate any cloud computing in near future

⇒ A difficult challenge for mobile services seeking the cloud
  ➞ Mobile services have most to gain from elasticity and lower costs
  ➞ But also most sensitive to data confidentiality risks
Potential Solutions

- **Full data encryption**
  - Store data on cloud only in encrypted form
  - Severely limits operations that can be run on data
- **Alternatives**
  - Homomorphic encryption, searchable encryption
  - Significant computation overhead, limited functionality

- **Our approach**
  - Protect legacy applications: *encrypt what you can*
  - Identify and encrypt “*functionally encryptable*” data
  - Automate process of moving applications to secure cloud
Outline

- Introduction

- Security and Privacy Challenges
  - Silverline: automated migration to a secure cloud
    - Data tracking and key labeling
    - Evaluation

- Conclusion
Goals

- Reconcile data confidentiality and elasticity in existing clouds
  - Develop mechanisms to enable applications to obtain both

- Help both new and deployed applications
  - Identify which features can and cannot be supported privately on the cloud
  - Automatically identify these features

- Minimize developer effort in securing their application
  - Developers can use the output of our techniques
  - And focus only on securing the “pain points”

- Contributions: A developer toolset called Silverline

Ben Zhao, UC Santa Barbara

6/14/10
## State of the Art

<table>
<thead>
<tr>
<th><strong>Use Advanced Encryption Schemes</strong></th>
<th><strong>Build a Secure Cloud from Scratch</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>□ Fully homomorphic encryption, searchable encryption, etc.</td>
<td>□ Augment clouds with TPMs, secure logs, more software</td>
</tr>
<tr>
<td>□ Pros: Can perform computation on encrypted data</td>
<td>□ Pros: Clients can verify the security properties of the cloud</td>
</tr>
<tr>
<td>□ Cons: Computationally expensive, and not composable</td>
<td>□ Cons: Delays deployment, expensive, and vulnerable to new attacks</td>
</tr>
</tbody>
</table>

Ben Zhao, UC Santa Barbara 6/14/10
Proposed Architecture

- Encrypt sensitive data before moving to cloud
- Store keys on organization's in-house hardware
- Only clients can decrypt the data they have access to
Scope of Our Approach

- Applications that involve limited computation
  - Where data sharing among users is prominent

- Example applications (e.g. LAMP architecture)
  - Social networks, message boards, database services
  - Location-based services, backup services

- After applying our techniques to these applications:
  - Functions on the public data still remain on cloud server
  - Server can still share (encrypted) data among users
  - Move data encryption and decryption to the clients
Outline

- Introduction
- Security and Privacy Challenges

  - Silverline: automated migration to a secure cloud
    - Key components: data tracking and key labeling
    - Evaluation

- Conclusion
Challenges in Migrating Applications

- **Identifying functionally encryptable data**
  - Sensitive data not *interpreted* by any computations
  - Thus encrypting data does not break functionality

- **Inferring encryption key assignment**
  - Identify the groups of users that have access to the data
  - Assign one key to each group, to provide strong privacy

- **Safe key management on the client**
  - Even if the cloud is malicious, the cloud should not be able to gain access to encryption keys
Approach: Dynamic Application Analysis

- **Encrypted data tracking**
  - Tag the sensitive data and track its usage

- **Database labeling**
  - Label the database cells with the users that access them
  - Learn the group of users that have access to a cell
  - Assign single key to each user group

- **Leverage recent work on safe browser architectures**
  - Verify the code from the cloud is signed by the organization
  - Run each iFrame in a separate process
Steps in Applying Silverline

- Analyze application during/after training set
  - Run the application; monitor database content’s usage
  - How is data used by application? Who accesses the data?

- Encrypted Data Tracking: Tag data from the database
  - Track flow/use of tagged data through application
  - Similar to information tainting approaches
  - Output tags to logs whenever data is “interpreted”

- Database Labeling: gather info about users that have access to different cells in the database
  - Partition the data based on usage, and assign keys
Part 1: Encrypted Data Tracking

- **Intuition:** Tag data read from the database, and identify data fields actually used in computation

- **Assume all data in the database is sensitive**
  - Each query in the system is statically assigned a number
  - Query results are tagged with a query number
  - These tags are propagated through the program
  - Warn when tagged data is used in computation
Data Tracking Policy

- Tag all data based on the MySQL query that accesses it

- Assignments propagate the tag from RHS to LHS
  - $a = b$ (tag\_a $\rightarrow$ tag\_b)

- Arithmetic and bitwise operations
  - Warn if any tagged data is used in the RHS
  - $a = b + c$ (warn if either $b$ or $c$ is already tagged)

- Relational operators
  - Warn if any operand is tagged
  - $a == b$ (warn if either $a$ or $b$ is tagged)

End result: If any query number appears in the log, its corresponding cells in the DB cannot be encrypted
Part 2: Automatic Key Inference

- Problem: What key should be used to encrypt a data item (cell), and who should get that key?

- Consider a database
  1) Four tables
  2) Many rows per table
  3) Many cells per row

- Goal: automate key assignment such that the privacy of each cell in the database is protected.
A Naïve Approach

- **Naïve approach**
  - Use one key to encrypt the whole database
  - Give this key to all users

- **Problem:** users can decrypt other users’ data
  - If the cloud joins the system as a user, privacy is compromised

- Ideal solution:
  - Give each user only the “power” he actually needs
  - Give access to only his own data (no more; no less)
Our Solution: Database Labeling

- **Intuition**
  - Learn who has access to the cells in the database during application runs (Database Labeling)
  - Use this to assign keys (Key Assignment)

- **Database Labeling**
  - When a query is run, tag returned cells with the name of the owner that ran the query
  - A cell’s label is a set of its owners
  - Over time, labels tell us owners of each cell in the database
# An Example

<table>
<thead>
<tr>
<th>Userld</th>
<th>Username</th>
<th>Gender</th>
<th>Location</th>
<th>Email</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Alice</td>
<td>M</td>
<td>Seattle</td>
<td><a href="mailto:a@gmail.com">a@gmail.com</a></td>
</tr>
<tr>
<td>2</td>
<td>Bob</td>
<td>F</td>
<td>London</td>
<td><a href="mailto:b@gmail.com">b@gmail.com</a></td>
</tr>
<tr>
<td>3</td>
<td>Carl</td>
<td>M</td>
<td>LA</td>
<td><a href="mailto:c@gmail.com">c@gmail.com</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MessageId</th>
<th>GroupId</th>
<th>Subject</th>
<th>Description</th>
<th>Timestamp</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>G1</td>
<td>Msg1</td>
<td>Desc1</td>
<td>2/2/2010</td>
</tr>
<tr>
<td>2</td>
<td>G2</td>
<td>Msg2</td>
<td>Desc2</td>
<td>3/3/2010</td>
</tr>
</tbody>
</table>

- Users login, and read their messages
- Build *labels* for each data object (list of authorized users)

Ben Zhao, UC Santa Barbara 6/14/10
Key Assignment Algorithm

- Use labels assigned to data cells
- Create **one key for every unique label**
  - Cells with the same label, even across tables, are accessed by the same set of users => can share the key
- Assign the key to **all the owners** in the label

<table>
<thead>
<tr>
<th>MessageId</th>
<th>Groupld</th>
<th>Subject</th>
<th>Description</th>
<th>Timestamp</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>G1</td>
<td>Msg1</td>
<td>Desc1</td>
<td>2/2/2010</td>
</tr>
<tr>
<td>2</td>
<td>G2</td>
<td>Msg2</td>
<td>Desc2</td>
<td>3/3/2010</td>
</tr>
</tbody>
</table>

Ben Zhao, UC Santa Barbara 6/14/10
Key Assignment Security Properties

- A user that doesn’t own a cell doesn’t get access to the cell
  - User won’t be in the label, won’t get the keys

- Cells with the same label can get the same key, without any loss in security
  - Has same set of owners => owners would get access to the keys, even if different keys were assigned

- Cells with different labels must get different keys
  - If same key is assigned; a user in one of the label gets access to cells he doesn’t own (hence more power) => privacy loss
Outline

- Introduction
- Security and Privacy Challenges

- Silverline: automated migration to a secure cloud
  - Data tracking and key labeling
  - Evaluation

- Conclusion
Data Tracking Implementation

- Modified **PHP runtime** to implement data tracking
  - Also modified the PHP-MySQL interface to tag data

- **_zval_struct** contains the main extension
  - The base structure from which all objects are derived
  - Added an unsigned int to store the query number tag
  - Tag used in warning about computations

- Run the application in this **modified runtime**
  - Logs all warnings in a log file

Ben Zhao, UC Santa Barbara  
6/14/10
Database Labeling Implementation

- Implemented on the client side
  - Applications perform labeling on a shadow copy of the database

- Labeling setup
  - Create a sample database of the application
  - Run the application under different user accounts
  - Label the shadow copy based on the returned results

- Analyze the labels in the end
  - Assign keys to the right set of users
## Evaluation

- Applied to four production applications (sourceforge.net)
  - A mix of different communication patterns

<table>
<thead>
<tr>
<th>Application</th>
<th>Purpose</th>
<th>Language</th>
<th>LOC</th>
<th># of Queries</th>
</tr>
</thead>
<tbody>
<tr>
<td>AstroSpaces</td>
<td>Social networking</td>
<td>PHP</td>
<td>14790</td>
<td>51</td>
</tr>
<tr>
<td>UseBB</td>
<td>Full-fledged message board</td>
<td>PHP</td>
<td>21264</td>
<td>114</td>
</tr>
<tr>
<td>Comendar</td>
<td>Community calendar</td>
<td>PHP</td>
<td>23627</td>
<td>42</td>
</tr>
<tr>
<td>EBMLCS</td>
<td>Library consulting</td>
<td>Ruby</td>
<td>7738</td>
<td>49</td>
</tr>
</tbody>
</table>

Ben Zhao, UC Santa Barbara  
6/14/10
## Encrypted Data Tracking Results

- A measurement of the encryptable database fields
  - Assume that auto-increment IDs as non-sensitive

<table>
<thead>
<tr>
<th>Application</th>
<th>Total</th>
<th>Sensitive</th>
<th>Encryptable</th>
<th>Non-Encryptable</th>
</tr>
</thead>
<tbody>
<tr>
<td>AstroSpaces</td>
<td>37</td>
<td>24</td>
<td>17</td>
<td>7</td>
</tr>
<tr>
<td>UseBB</td>
<td>106</td>
<td>81</td>
<td>67</td>
<td>14</td>
</tr>
<tr>
<td>Comendar</td>
<td>105</td>
<td>57</td>
<td>41</td>
<td>16</td>
</tr>
</tbody>
</table>

Ben Zhao, UC Santa Barbara 6/14/10
Take Away from Data Tracking

- **Most of the fields are not used in computation**
  - Just read from the database and returned to users
  - Number of non-encryptable fields increase with increase in the computation-oriented features

- **Several ways to handle the non-encryptable fields**
  - Move the computation to the client
    - For example, display-related fields (theme, icon, style, etc.)
  - Leave them in plain text on the cloud
    - Non-sensitive fields such as date, friend request status, etc.
Examples of Worst Case Features

- **Keywords searches**
  - On the content of events, posts
  - On usernames

- **Cannot move to the clients**
  - As it involved massive data transfer

- **Cannot leave on the cloud in plain text**
  - Violates privacy

- **Good news:** only one or two such features exist in the applications used in evaluation
Database Labeling Results

- There were different types of data in the applications:
  - Data accessible to the entire world (public)
  - Data accessible to all users in the system (community)
  - Data accessible to a group of users (group)
    - Or a pair of users (or friends)
  - Data accessible only to a user (personal)

- Correctly partitioned all the different types of data:
  - Identified all the users that have access to the data items
  - And assigned the keys to the right set of users
Example: Community Calendar

- Had all four types of data
- Created a sample database
  - With 50 users; 10 groups
  - Each user created four events at four access levels
  - Assigned group events to random groups

- Out tool assigned 61 keys
  - 50 user-specific keys, 1 for public, 1 for the community, and 9 for the groups
Summary and Implications

- A new approach to obtain both privacy and elasticity from today’s compute clouds
- Helps both “native” and deployed applications
- Many more companies can leverage the cloud
  - Including companies with sensitive data
  - Get both cost savings and privacy from clouds

Ben Zhao, UC Santa Barbara
Ongoing / Future Work

- **Automatic application partitioning**
  - Supports all features in an application

- **Improving the coverage of dynamic analysis via “rule mining”**
  - Leverage work on associative rule mining to learn data access policies, and then improve key assignment algorithm

- **Using specialized encryption selectively to retain computation on the cloud**
  - Works well in combination with automatic partitioning
Thank You, Questions?

For more information, check out:

http://current.cs.ucsb.edu