Software Updates in the Cloud
Challenges and Opportunities

Iulian Neamtiu

University of California, Riverside
Cloud availability is key...

U.S. military takes cloud computing to Afghanistan
...but updates are essential

**Planned maintenance tomorrow, Monday**

We’re planning a significant database upgrade that will greatly increase system performance. To do this, we will require two planned maintenance windows: Friday, May 8th from 2p–3p Pacific and again on Monday, May 11th from Noon–1p Pacific.

We’re taking these maintenance windows earlier in the day because of the extent of the changes involved. We want to make sure that we are able to respond quickly and efficiently to any issues that may arise from the work being performed.

We will update here with any changes or additional information about this upgrade.

**Update (2:00p):** Today’s maintenance window is about to commence. During this time, and the subsequent window on Monday, all Twitter services will be unavailable.
Problem 1: unavailability

Planned maintenance tomorrow, Monday

year ago

We’re planning a significant database upgrade that will greatly increase system performance. To do this, we will require two planned maintenance windows: Friday, May 8th from 2p–3p Pacific and again on Monday, May 11th from Noon–1p Pacific.

We’re taking these maintenance windows earlier in the day because of the extent of the changes involved. We want to make sure that we are able to respond quickly and efficiently to any issues that may arise from the work being performed.

We will update here with any changes or additional information about this upgrade.

**Update (2:00p):** Today’s maintenance window is about to commence. During this time, and the subsequent window on Monday, all Twitter services will be unavailable.
Problem 2: unsafe updates

Twitter blames website upgrade for re-introducing XSS hole

By John Leyden • Get more from this author
Posted in Malware, 22nd September 2010 10:21 GMT

Twitter said it identified and fixed the cross site scripting flaw that led to meltdown on Tuesday a month ago, only to undo this fix with a later web site update.
Challenges

1. Availability

2. Safe updates
## Cloud update frequency

<table>
<thead>
<tr>
<th>Application</th>
<th>Average release interval (days)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facebook</td>
<td>&lt; 7</td>
</tr>
<tr>
<td>MediaWiki (Wikipedia)</td>
<td>21</td>
</tr>
<tr>
<td>Velvet</td>
<td>16</td>
</tr>
<tr>
<td>Sumo Paint (Photoshop in the Cloud)</td>
<td>13</td>
</tr>
</tbody>
</table>

*Extremely short update cycles!*
Rolling Cloud updates

Active domain

Inactive domain

Clients

Software Updates in the Cloud: Challenges and Opportunities
Rolling Cloud updates

Inactive domain  Active domain

Clients
Rolling upgrades can lead to outages...

Gmail's web interface had a widespread outage earlier today, lasting about 100 minutes. We know how many people rely on Gmail for personal and professional communications, and we take it very seriously when there's a problem with the service. Thus, right up front, I'd like to apologize to all of you — today's outage was a Big Deal, and we're treating it as such. We've already thoroughly investigated what happened, and we're currently compiling a list of things we intend to fix or improve as a result of the investigation.

Here's what happened: This morning (Pacific Time) we took a small fraction of Gmail's servers offline to perform routine upgrades. This isn't in itself a problem — we do this all the time, and Gmail's web interface runs in many locations and just sends traffic to other locations when one is offline.
Challenges and Opportunities

1. Availability
   - *On-the-fly Cloud updates*

2. Safe updates
   - *End-to-end update safety*
Why do updates go wrong?

Windows Azure: “... different users will access different versions of the application in the middle of the update”
Why do updates go wrong?

Storage $V_1$

Compute $V_2$

Storage $V_2$

Compute $V_1$

Storage $V_2$

Compute $V_2$

Client $V_1$
End-to-end update safety

- Consistent semantics
  - Impedance mismatch

\[ \sigma(\text{Storage } V_i) \approx \sigma(\text{Compute } V_i) \approx \sigma(\text{Client } V_i) \]

- Specification languages?
  - Swift, Fable, Links

- “Thick” clients?
Dynamic (on-the-fly) Cloud updates

Inactive domain — Active domain

Rolling Update

Dynamic Update

- Mechanism?
- Safety?
On-the-fly Storage Updates

- Update transparency?
  - DBMS “dynamic reorganization”
- Parallel databases?
  - e.g., ASTERIX, Teradata
- NoSQL storage?
  - e.g., Amazon SimpleDB, Windows Azure blobs/stores/queues
On-the-fly **Compute** Updates

- Dynamic Software Updating
  - Works for C, Java (Ginseng, Jvolve)
- Other languages/paradigms?
  - e.g., Web apps, MapReduce
- Language support for evolution?
On-the-fly **Client** Updates

- Thick clients?
- Mash-up applications?
System-wide Challenges

- End-to-end safety?
- Synchronization?
- HW/SW failures?
Conclusions

• Challenges
  – Extremely frequent updates
  – Downtime, update failures undesirable
  – New models for consistency, programming

• What are we doing for the Cloud?
  – “Elastifying” legacy applications
  – Formal end-to-end safety spec/enforcement
  – Dynamic Update mechanisms/safety

http://www.cs.ucr.edu/~neamtiu/index.html#cloud