Secure Serverless Computing using Dynamic Information Flow Control

Kalev Alpernas, Cormac Flanagan, Sadjad Fouladi, Leonid Ryzhyk, Mooly Sagiv, Thomas Schmitz, and Keith Winstein

@kalevalp  kalevalp

TEL AVIV UNIVERSITY  vmware  UC SANTA CRUZ  Stanford
The Cloud is Growing Faster than Ever

Stellar Intel Earnings Underscore Tech Sector Strength, Enormous Growth Potential In The Cloud

Dave Altavilla  Contributor  
Consumer Tech  
I cover break-out tech in mobile, on desktop and in the data center.
The Cloud is Growing Faster than Ever

Strong Cloud Growth Continues to Drive Microsoft Higher

Growth was strong across all segments, but Azure was still the highlight.

Danny Vena (TMFLifeIsGood)
Oct 28, 2018 at 10:37AM

Over the past several years, Microsoft (NASDAQ:MSFT) has accomplished an amazing feat, transforming from a shrink-wrapped business software provider to one of the biggest names in cloud computing. The company has been successful challenging the cloud leader, Amazon Web Services, and making meaningful headway, currently ranking in a strong second place.

As the global digital transformation gains steam, investors were watching closely for signs that Microsoft’s cloud momentum would continue. When the company reported the financial results of its fiscal 2019 first quarter, which ended Sept. 30, 2018, Microsoft showed that it has much more growth in the tank, producing a record first quarter.

AUTHOR
Danny Vena
(TMFLifeIsGood)

Daniel W. Vena, CPA, CGMA is long-term investor searching for intangibles that provide explosive growth opportunities in his investments. He served on active duty with the US Army and has a Bachelors degree in accounting.

ARTICLE INFO
Oct 28, 2018 at 10:37AM
Technology and Telecom

STOCKS
18,495 views | Jan 7, 2018, 07:36pm

83% Of Enterprise Workloads Will Be In The Cloud By 2020

Louis Columbus Contributor ☺

TWEET THIS

Digitally transforming enterprises (63%) is the leading factor driving greater public cloud engagement or adoption today.

66% of IT professionals say security is their most significant concern in adopting an enterprise cloud computing strategy.
The Cloud is Growing Faster than Ever

Is security now a driver of cloud adoption?
The Internet is a Leaky Place

Facebook just had its worst hack ever — and it could get worse

By Donie O’Sullivan, CNN Business
Updated 9:22 AM ET, Thu October 4, 2018

New York (CNN) — On Sunday, September 16, engineers at Facebook detected some unusual activity on the social media platform’s networks. It was an attack, the biggest security breach in Facebook’s history. And it would take the company 11 more days to stop it.

Now, almost a week since the public was first told of the attack, we still barely know anything about what happened.

We don’t know who the hackers were, or what they were looking for. We don’t know whether they were targeting particular people in certain countries. We don’t know how long they had access to users’ information. And we don’t know what, if anything, they took.
Cathay Pacific got hacked, compromising the data of millions of passengers

By Jethro Mullen, CNN Business
Updated 12:22 AM ET, Thu October 25, 2018

Hong Kong (CNN Business) — One of Asia's top airlines has discovered a data breach in which the personal information of more than 9 million passengers may have been stolen.

Cathay Pacific (CPCAY) said late Wednesday that a wide range of data — including passengers' names, dates of birth, phone numbers, email addresses and passport numbers — was exposed in a hack of its information systems earlier this year.

"We are very sorry for any concern this data security event may cause our passengers," CEO Rupert Hogg said in a statement. The Hong Kong-based carrier is in the process of contacting affected people, he added.
Amazon Web Services Customers Can Hack AWS Cloud And Steal Data, Says Oracle CTO Larry Ellison

(Note: After an award-winning career in the media business covering the tech industry, Bob Evans was VP of Strategic Communications at SAP in 2011, and Chief Communications Officer at Oracle from 2012 to 2016. He now runs his own firm, Evans Strategic Communications LLC.)

CLOUD WARS -- Oracle founder Larry
The real victim in health data breaches? Patients' medical identities

By Jessica Davis | October 29, 2018 | 05:03 PM

Hackers can perform synthetic identify theft long after a cybersecurity event by piecing together data to conduct medical and insurance fraud.
The Internet is a Leaky Place

We’re all Equif*cked

Josh Constine  @joshconstine  /  1 year ago
Cloud Applications

Traditional (monolithic) three tiered application
Cloud Applications

Goal: keep data secure – Information Flow Control (IFC)

Non-Interference
Cloud Applications

Servers are vulnerable
Cloud Applications

Servers are vulnerable
Cloud Applications

What happens when we replace them with serverless functions?
Cloud Applications

Functions are vulnerable
Cloud Applications

Polyglot, distributed environment makes things even harder
Monolithic Applications

Long running processes, serving multiple users, managing complex local state
Serverless Applications

Every execution serves a single request, no local state

Alice

Bob

AWS Cloud

Amazon API Gateway

DB Instance
Serverless Applications

Every execution serves a single request, no local state
Serverless Applications

Every execution serves a single request, no local state
Serverless Applications

Every execution serves a single request, no local state
Serverless Applications

Every execution serves a single request, no local state
Underneath Everything, there is Hope

Serverless computing is our opportunity to implement efficient IFC

Perform IFC tracking at function boundaries
Serverless+IFC

How do we exploit Serverless to design an efficient IFC system?
Serverless+IFC

Alice makes a request

Alice

Bob

Amazon API Gateway

AWS Cloud

DB Instance
Serverless+IFC

The Serverless framework spawns a function to serve the request
Serverless+IFC

Functions may spawn additional functions
Serverless+IFC

Functions interact with DB
Serverless+IFC

Function reads **labeled** data and is marked with the **label**
Serverless+IFC

Label is propagated across function calls
Serverless+IFC

Other requests may occur in parallel
Serverless+IFC

Parallel requests spawn new functions, that are independent from one another

Alice

Bob
Serverless+IFC

Alice

Bob

Amazon API Gateway

AWS Cloud

DB Instance
Serverless+IFC

Function reads **labeled** data and is marked with the **label**
Serverless+IFC

Function returns result

Alice

Bob

Amazon API Gateway

AWS Cloud

DB Instance
Serverless+IFC

Before the result is sent to the user, we can check that the security policy is not violated.
Serverless+IFC

And send the response if it does not violate the security policy
Serverless+IFC
Serverless+IFC

Function returns result
Serverless+IFC

Before the result is sent to the user, we can check that the security policy is not violated
Serverless+IFC

And send the response if it does not violate the security policy
Serverless+IFC

Security policy might have a hierarchy – making shared dbs unavoidable
Serverless+IFC

Security policy might have a hierarchy – making shared dbs unavoidable
Serverless+IFC

Security policy might have a hierarchy – making shared dbs unavoidable
Serverless+IFC

What happens when a hacker tries to steal data?
Serverless+IFC

The hacker might exploit some vulnerability in the application (e.g. code injection)
Serverless+IFC

Trapeze guarantees security even if the underlying function code is completely compromised
Serverless+IFC

The malicious code may read a secret, and is marked with a label.
Serverless+IFC

The malicious function then tries to send the secret in a response
Serverless+IFC

At which point we can check if the hacker is allowed to see the data
Serverless+IFC

And block the request from being sent
Termination Channel

But – the attacker learned of the existence of secret data
Termination Channel

Which is bad if the existence of a secret is itself a secret
Termination Channel

Termination Insensitive Non-Interference is insufficient!
Termination Channel

Attacker can encode a secret value by making certain values ‘inaccessible’
Termination Channel

Attacker reads a secret value
Termination Channel

Attacker reads a secret value ('1001')
Termination Channel

Write something to predefined locations
Termination Channel

Encode secret as a memory storage pattern
Termination Channel

Attacker can leak every bit by accessing each cell separately
Termination Channel

Attacker can leak every bit by accessing each cell separately
Termination Channel

Attacker can leak every bit by accessing each cell separately
Closing the Termination Channel

How do we get Termination *Sensitive* Non-Interference?

- Static labeling instead of floating labels
  - Label assigned at function invocation
  - Output behavior depends on invocation, not secret

- Faceted data store
Closing the Termination Channel

Modified read and write semantics

AWS Cloud

Harry
Closing the Termination Channel

Writes add a facet to the location, instead of overwriting
Closing the Termination Channel

An execution only sees the facets that is allowed to see
Closing the Termination Channel

An execution only sees the facets that is allowed to see
Closing the Termination Channel

An execution without the correct label can’t distinguish between an empty cell, and a faceted cell.
Trapeze

• Runtime enforcement of *Termination Sensitive Non-Interference*
  • Attacker cannot leak information by affecting execution termination

• Function executions get a security label when invoked
  • Label matches the security privilege of the invoker

• Faceted data store semantics
  • Items in the DB may have multiple values
  • Depending on the security label of the function that wrote the data
Trapeze – Goals

• Practical IFC system for serverless applications

• Low overhead in runtime

• Allow running existing application securely with minimal modification

• Transparency – secure applications have the same semantics as when running without Trapeze
Trapeze Architecture

Shim layer wraps functions, controlling communication and enforcing faceted store semantics
Evaluation

Test cases:
• Nordstrom Hello, Retail!
• gg
• Image feature extraction

Questions:
• Performance
• Ease of deployment
• Transparency
Evaluation – Hello, Retail!
Evaluation – Hello, Retail!

Security policy:
# Evaluation Results

<table>
<thead>
<tr>
<th>Scenario</th>
<th>#(\lambda)</th>
<th>(\lambda) runtime (ms)</th>
<th>(\Delta(%))</th>
<th>App. code</th>
<th>Modified</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>baseline</td>
<td>Trapeze</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hello, Retail!</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Update &amp; Purchase</td>
<td>435</td>
<td>51,246</td>
<td>98,563</td>
<td>92.33</td>
<td>1300</td>
</tr>
<tr>
<td>Build &amp; Browse</td>
<td>955</td>
<td>257,799</td>
<td>391,900</td>
<td>52.03</td>
<td></td>
</tr>
<tr>
<td>gg</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mosh</td>
<td>111</td>
<td>681,173</td>
<td>654,448</td>
<td>-3.92</td>
<td>8800</td>
</tr>
<tr>
<td>git</td>
<td>917</td>
<td>2,602,500</td>
<td>2,660,721</td>
<td>2.24</td>
<td></td>
</tr>
<tr>
<td>vim</td>
<td>234</td>
<td>1,242,873</td>
<td>1,338,128</td>
<td>7.66</td>
<td></td>
</tr>
<tr>
<td>openssh</td>
<td>654</td>
<td>1,626,233</td>
<td>1,649,139</td>
<td>1.41</td>
<td></td>
</tr>
<tr>
<td>Image Feature Extraction</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Upload</td>
<td>1</td>
<td>475</td>
<td>525</td>
<td>10.4</td>
<td>95</td>
</tr>
<tr>
<td>Extract</td>
<td>1</td>
<td>1,882</td>
<td>2,114</td>
<td>12.3</td>
<td></td>
</tr>
</tbody>
</table>
# Evaluation Results

<table>
<thead>
<tr>
<th>Scenario</th>
<th>#λ</th>
<th>λ runtime (ms)</th>
<th>Δ(%)</th>
<th>App. code</th>
<th>Modified</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>baseline</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Update &amp; Purchase</td>
<td>435</td>
<td>51,246</td>
<td>98,563</td>
<td>92.33</td>
<td>1300</td>
</tr>
<tr>
<td>Build &amp; Browse</td>
<td>955</td>
<td>257,799</td>
<td>391,900</td>
<td>52.03</td>
<td></td>
</tr>
<tr>
<td><strong>Hello, Retail!</strong></td>
<td></td>
<td><strong>Δ(%)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mosh</td>
<td>111</td>
<td>681,173</td>
<td>654,448</td>
<td>-3.92</td>
<td></td>
</tr>
<tr>
<td>git</td>
<td>917</td>
<td>2,602,500</td>
<td>2,660,721</td>
<td>2.24</td>
<td>8800</td>
</tr>
<tr>
<td>vim</td>
<td>234</td>
<td>1,242,873</td>
<td>1,338,128</td>
<td>7.66</td>
<td></td>
</tr>
<tr>
<td>openssh</td>
<td>654</td>
<td>1,626,233</td>
<td>1,649,139</td>
<td>1.41</td>
<td></td>
</tr>
<tr>
<td><strong>gg</strong></td>
<td></td>
<td><strong>Δ(%)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Upload</td>
<td>1</td>
<td>475</td>
<td>525</td>
<td>10.4</td>
<td>95</td>
</tr>
<tr>
<td>Extract</td>
<td>1</td>
<td>1,882</td>
<td>2,114</td>
<td>12.3</td>
<td></td>
</tr>
</tbody>
</table>
# Evaluation Results

<table>
<thead>
<tr>
<th>Scenario</th>
<th>#(\lambda)</th>
<th>(\lambda) runtime (ms)</th>
<th>(\Delta(%))</th>
<th>App. code</th>
<th>Modified</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>baseline</td>
<td>Trapeze</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hello, Retail!</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Update &amp; Purchase</td>
<td>435</td>
<td>51,246</td>
<td>98,563</td>
<td>92.33</td>
<td>1300</td>
</tr>
<tr>
<td>Build &amp; Browse</td>
<td>955</td>
<td>257,799</td>
<td>391,900</td>
<td>52.03</td>
<td></td>
</tr>
<tr>
<td>gg</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mosh</td>
<td>111</td>
<td>681,173</td>
<td>654,448</td>
<td>-3.92</td>
<td>8800</td>
</tr>
<tr>
<td>git</td>
<td>917</td>
<td>2,602,500</td>
<td>2,660,721</td>
<td>2.24</td>
<td></td>
</tr>
<tr>
<td>vim</td>
<td>234</td>
<td>1,242,873</td>
<td>1,338,128</td>
<td>7.66</td>
<td></td>
</tr>
<tr>
<td>openssh</td>
<td>654</td>
<td>1,626,233</td>
<td>1,649,139</td>
<td>1.41</td>
<td></td>
</tr>
<tr>
<td>Image Feature Extraction</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Upload</td>
<td>1</td>
<td>475</td>
<td>525</td>
<td>10.4</td>
<td>95</td>
</tr>
<tr>
<td>Extract</td>
<td>1</td>
<td>1,882</td>
<td>2,114</td>
<td>12.3</td>
<td></td>
</tr>
</tbody>
</table>
Summary

• Serverless computing
  • Security Risks
  • Security opportunities

• Trapeze
  • A practical IFC system for serverless applications
    • Guarantees TSNI
  • Low overhead (mostly) thanks to function granularity
  • High transparency

GitHub https://github.com/kalevalp/trapeze