Tom - Welcome to our talk, about the cloud as you can see it’s a pretty nice picture

Carlos - But it’s still a dirt road, we have a long way to go
- Tom Hughes-Croucher
- Work for Yahoo! Developer Network
- Get to work with a lot of startups
- Met Carlos in an internet cafe in San Francisco
Hi, I'm Carlos Bueno, an Engineer with Yahoo Mail.

Perviously I've founded a startup which was what I was doing when I met Tom.

My startup head trouble using two clouds together.

Before we start, lemme take a poll: how many of you work for a startup? Less than 50 people? Ok. And how many enterprise folks do we have? Right. How many are technical? You check in code? And business? How many people use the cloud? Planning to? Ok. Good mixed bag. There's a lot to say for all of you.
The title of this talk is the "Cloud's Hidden Lockin", which sounds kind of sinister. And it is, in the sense of being hidden and dangerous. It's not sinister in the sense of being planned.

There's no guy in the shadows twirling his moustache. The traps we're talking about are an artefact of how this platform evolves.

Whether or not it's planned it's still dangerous. What we want to do in this talk is to tell you how it functions, what you can do about it in the near term, and how you can help solve it in the medium term.
The lock-in of tomorrow is going to be about where your data lives and how fast and how expensive it will be to move it to where you need it. You can have all the open formats you want, but if it costs more money to physically move your data than to maintain it where it is, or if it's too slow to feasibly use Service A on Cloud B, you're effectively locked-in to one vendor.
Lock-in isn’t about APIs and Data Formats

Tom - One take away: lock-in not just about APIs and data formats

Carlos - open-source doesn’t solve lock-in. some kinds of problems you can't code your way out of.

Tom - Thinking like this is fighting the last interoperability battle

If there's one thing to take away from this talk, it's that lock-in is not just about APIs and data formats.

It's a BIG mistake to think that lock-in has gone away because we have open-source software. There are some kinds of problems that you can't program your way out of.
Latency, always sucks

Tom - So what is latency?
Carlos - time it takes to send a bit of data from point A to point B.
Tom - Punchline
Carlos - Hosting companies spend time and money to reduce the network latency within and between the buildings they own.
Tom - as a customer intra-colo traffic is fast and free. Low latency makes it feasible to use Amazon S3 with Amazon EC2.
Carlos - But between colos is kind of a no-man's land. You are left on your own.

---

Latency is roughly the time it takes to send a bit of data from point A to point B. Everyone in the hosting game: cloud vendors, managed hosting vendors, colo providers, etc, spend enormous amounts of time and money to reduce the network latency within and between the buildings they own.

The benefit to you as a customer is that intra-colo traffic is fast and free. Low latency is why it's feasible to use Amazon S3 with Amazon EC2.

Why does latency suck?
Well,
because
it makes everything slower.

But between colos is kind of a no-man's land. You are left on your own.
Carlos - Even if you were ok with high latency between cloud A to cloud B, the way things are going you'll pay twice for each byte sent between them.

Tom - Trend started with AWS. simple model, data in-colo free / all data out-colo fixed price per byte.

Carlos - There are lots of business reasons for this pricing model.

Tom - no distinction between data over the open 'net and freepeer cuts off many interesting uses of the cloud.

Carlos - And paying twice is no fun

---

Bandwidth metering is the other jaw of the trap.
Even if you were ok with high latency between cloud A to cloud B, the way things are going you'll pay twice for each byte sent between them.

This trend started with Amazon Web Services.
They introduced a simple model, where all data in-colo was free and all data out-colo had a fixed price per byte.

There are lots of business reasons for this pricing model.
But by making no distinction between data that goes out over the open 'net and data that goes over a freepeer, they cut off many interesting uses of the cloud.
Carlos - the proposition is pretty stark. Stay inside one cloud and everything is fast and cheap. Stray outside and suddenly everything gets slow and expensive.

Tom - It’s not so much a cloud as a bunch of bubbles

---

Put it that way and the proposition is pretty stark. Stay inside one cloud and everything is fast and cheap. Stray outside and suddenly everything gets slow and expensive.
Lock-in is whatever makes it hard to switch vendors

**Tom** - lockin is in whatever makes it expensive to switch between or interoperate with different vendors

**Carlos** - There are some problems you can’t code your way out of. Today if you ask an engineer how to use Cloud Server A with Cloud Database B, she’ll tell you it’s a) impossible and b) stupid.

**Tom** - Of course copying a gigabyte from Amazon to Rackspace is slow: it goes over the open internet. Of course you should be charged twice for the same byte: how else can they make money?

**Carlos** - really? is this the internet we know and love? is this the 21st century or the 19th?

---

The essence of lockin is in whatever makes it expensive to switch between or interoperate with different vendors. Cloud computing in particular brings up some new problems you can't code your way out of.

In the cloud world today it's taken as a given that you can't mix and match services from different cloud vendors. Of course copying a gigabyte from Amazon to Rackspace is slow: it goes over the open internet. Of course you should be charged twice for the same byte: how else can they make money?

You know what that sounds like? It sounds like the postal service circa 1840.
Not a new problem

Tom - Back in the day every country had their own rules and infrastructure. You had to think very hard about the best routes and accounting for all of the delays and expenses along the way.

Carlos - Countries slowly figured out that simple rules and flat postal rates inside their borders encouraged commerce by reducing friction and overhead.

Tom - But it took a while for that idea to extend worldwide. A lot of infrastructure --and, frankly, a lot of trust-- had not yet been built up.
carlos – a letter from manila, philippines to boston, usa took a long, sad route.
Manila → Panama
Panama → Other side Panama
Every link was slow. Different portage fees and taxes at each point.
Tom - Early 1870's two things. US and British Empire signed a flat rate treaty: 16 cents/oz anywhere in British Empire to anywhere US, no questions asked.
Tom - This was not necessarily cheaper, but it as much simpler and faster. So crafty folk in Manila started sending letters to the US through Hong Kong to take advantage of the special rates.
Tom - The United States completed its transcontinental railroad.
Carlos - Fun fact: an 8oz letter along this route cost about the same, inflation-adjusted, as a FedEx letter in 2009: 16c/oz is equivalent to $4/oz of today's money.
Carlos - This worked so well that it only took another generation before a big chunk of the world agreed to a single treaty, the Bern Treaty. Which allowed global mail at a flat fee.
Manila → San Francisco
San Francisco → Boston
Carlos - The Bern Treaty, with three simple rules:

1. Uniform letter rate between countries
2. Equal treatment for foreign and domestic mail
3. Origin country keeps the money

That third bit is the keystone: instead of everyone scrambling for their little cut, the treaty worked on the principle that every letter begat a reply so things pretty much came out even anyway. So the sending country keeps the money and makes it simpler for everybody.
There were four separate things needed to bring about what we think of as the modern international mail system.

First you needed better infrastructure inside and between countries.

You needed literally more portable objects, envelopes with stamps on them, instead of loose sheets, scrolls, wax seals, etc.

You also needed standardization of the rates and address formats so one letter can travel anywhere.

Last was a uniform rate, no-questions-asked promise to deliver via optimized routes, what we would now call a "peering agreement". Two ends of the agreement would treat each other as peers, and honor each others' communications as they would their own.
Cloud Interoperability

- Peering
  - reciprocal agreements, flat rates
- Compatibility
  - open APIs and formats
- Portability
  - standard VMs and images
- Better Infrastructure
  - reduce inter-cloud latency

It's that kind of system we don't have, but should, in the cloud.

We need better infrastructure in the form of optimized routes between clouds.

We need to be able to move our virtual machines and configurations around without special help.

We need to make sure we don't get locked-in by screwball APIs or data formats.

Most of all, we need the various cloud and web services vendors to commit to honoring each other's traffic without clobbering us, their customers, with metered billing.
Tom - Or to look at it another way, true interoperability is built on top of these foundations.
Where we are today?

Carlos - The state of the cloud is pretty close the state of the postal system circa 1840. We have better infrastructure between our colos than they had between countries but it's still too expensive to move large amounts of data around. It's also too slow to send important, time-sensitive data around at volume.

Tom - Meaning you can't feasibly use database A with cloud B.
Carlos - The most visible work people are doing seems to be around compatibility and portability. Not much is being done about infrastructure and peering.

Tom - It's nice that some day you'll be able to migrate machines from one cloud to another via Microsoft Simple Virtual Machine Object Protocol 2.0, but if moving the data is slow and expensive, who cares?
Carlos - Many startups are helping to make it easier to move between clouds.
Rightscale

Carlos – Rightscale

manage
backup
scaling
performance
Not:
- Open
- About Cloud Customers
- A Manifesto
- Holding to anything
Carlos – Protocol for shuttling virtual machines around
Yahoo! OpenCap

Tom – OpenCap (proposed IETF standard)

language for specifying how to manage clouds automatically
deal with bloated OS stack
incompatible apis
automation
Carlos - But Tom and I want to hit the full stack, the things we just mentioned only cover the middle two pieces.
Making it happen

Tom - So how do we make it happen?
Tom - As we all know the internet is a series of tubes.

Carlos - But seriously, it’s not such a bad analogy. We need better connections between colos.
Carlos - The interesting but unsurprising fact about datacenters is that they tend to cluster in the same areas. Anywhere there is cheap land, electricity, tax breaks and good backbone, there will usually be some aggressively anonymous buildings containing millions of dollars of hardware. So when the big internet players of the world decide to hook up, they will find that it’s relatively cheap. Often their counterparts are in the next building or just across town.

Tom - We are compiling a map of known datacenters from Yahoo, Google, Amazon, Microsoft, etc.
Carlos - The last component is the peering agreements around these new fast inter-cloud routes.

A peering agreement is simple in principle though the details differ. Remember the flat-rate treaty between the US and UK? That was an agreement to see each other as peers, and to treat the other’s mail just as you would treat your own.

Tom - So far the only instance of explicit cloud peering, ie, two parties did the work specifically to make it easier for their common customers to interoperate, was between Joyent and Facebook in late 2007. Joyent and Yahoo (2009).
Concrete example

I used to work for a search advertising startup called Terespondon. We were based in Miami. We brokered user web searches at very high volume between a certain large software company in Seattle with partners in Brasil.

Needless to say, our business depended on how fast and how cheap we could do this.

We discovered that one of our partners had some servers in the same datacenter as we did in miami.

So we did up an agreement over email and connected our servers with a cheap network cable, and we were off to the races.
Peering Agreement: $0
Firewall rules: 5 mins ($0)
Cross-over cable: $15.99
If anyone from Mastercard is here, apologies for parodying your excellent ads. Please don’t sue us.

There was a very direct effect on our top line, because we could service more searches faster, and bottom line, because we were spending less on bandwidth.
Recommendations

We're not in the promised land yet, but there are things you can do to avoid lock-in and also help push the evolution along.
If you are a CTO

Tom - If you are a CTO at a large company you are in a good position. If the vendors start talking about VPN, ask them why it's so slow and expensive to use it.

Right now, before you hand over your data, you have the most leverage you ever will.

You all know more about high-level negotiation than I. All I'm saying is that you should really push them on latency and cost of the bandwidth. If you are a Fortune 1000 you are probably big enough to swing a peering agreement on your own.

Pick “best of breed”
If you are a startup

if you are  smaller company with major operations in the cloud, I recommend using two separate vendors -- call them Cloud A and B. Host your site as normal in Cloud A. There's no reason to try to run them in parallel. Host your development / staging systems in Cloud B. Also keep a mirror of all your data backups in Cloud B. This forces good habits. You'll never get stuck writing cloud-specific code because you have to deal with both all the time as a normal part of your deployment and testing. Second you'll have off-site backups, always a good thing. Third, you'll have a complete copy of production data to test on. Once you have this warm spare running on cloud B, you're in a good position. In the event of a disaster you can provision servers and spin up at the new location in less time than it takes for the DNS change to propagate. All that is good hygine, but it's even better as leverage.

Make sure that both Cloud A and B know what you are up to. Call up the salespeople and say, "Hey, I love you guys!. But I'm doing backups to B. It says here on cloud-peering.com that you and B have datacenters next to each other. Why am I paying you so much to send data across the street? Why don't you two set up a peering agreement? If you do that I'll be able to spend more money on servers instead of backups."

Make it a very straightforward, win-win, top and bottom-line issue.
If you run a cloud
Tom - Now, why would the cloud vendors, the ones who apparently benefit most from lock-in, want to do peering?

image: http://www.flickr.com/photos/wheatfields/2587147000/
Carlos - Instead of fighting for a piece of the little pie, they can make the pie much much much bigger.

image: http://www.flickr.com/photos/diamondjoe/122553453/sizes/o/
Network effects. Throughout history, the **minute** competing networks have interconnected, total usage went off the gauge. 8 months after SMS interop went live in the UK, Portugal and Australia in 1999, network usage jumped as much as 900%. Holy Shit, Batman!

**Image credit:** [http://www.flickr.com/photos/kiwanja/3170279816/sizes/o/](http://www.flickr.com/photos/kiwanja/3170279816/sizes/o/)

Turn data storage into a commodity that can be traded, sold, swapped, reserved for future delivery, etc.

Spot Market for Storage

Tom - Widespread peering opens up things were not possible before. For example, why not selling data storage on the open market? Companies could trade in units of terabyte/seconds, swap storage in remote locations, etc. Just like they do with shipping containers and physical storage.

Carlos - (Note: this will leave current privacy, crimes investigation, and censorship laws in the dust)

Image Credit: http://www.flickr.com/photos/thewalkingirony/3051500551/sizes/o/
An extreme example of peering for data transfer is “Internet2”, a private network universities use to send large blobs of research around.

For example, the University of San Diego Supercomputing Center has projects that swap data with sites in Tokyo and New Jersey. A big reason they don’t put their data or compute nodes in the cloud is the cost of transit. So instead they use their own systems and Internet2.

Average throughput: 9.08 gigabits per second roundtrip Tokyo -> Seattle -> Amsterdam -> Tokyo
What is Y! doing about it?

Tom - Are not currently offering open cloud. Providing services for our internal customers.

Tom - But we are opening sourcing our cloud infrastructure.

Tom - and Providing aggregator for Web Services. YQL is Hong Kong.
Apache Traffic Server

ats used to be called yts (yaho! traffic server)

open source project for a fast caching proxy

part of the infrastructure needed between cloud vendors is caching. e.g. storing repeated requests between each other for their customers
blind peering on behalf of smaller services
Carlos - just to recap, true interoperability not just about software/data formats. It’s about flat rates and the ability to use more than one vendor

Tom - It’s about building an economy where everyone benefits, not just the cloud vendors, or the customers.

Carlos - bringing the cloud into the 21st c, at least as sophisticated as the mail system circa 1894
Questions

tom
@sh1mmer
kid666.com

Carlos
carlos.
bueno.org
express000
minds-eye
good directly down
catsegovia
darwinbell
deadhorse
fersayyge
gbaku
guitrento
jdlasica
julishannon
lightcliff
livenature
lwr
manchesterunitedman1
martindew
midorisyu
mike Baird
mzwazzard
mwichary
neenahhistory
Nicholaslaughlin
normanbleventhalmap
center
oceanviews
ogil
postaltrice
russelldavies
simongurr
stansich
tueksta
usnationalarchives

CC Flickr Photo Credits