The API Platform

Marcin Wichary

QCon San Francisco

November 17, 2011

Greg Brail @gbrail

Apigee @apigee

Overview

What is an API?

What is an API Platform?

Typical API Call Flow

API Platform Challenges

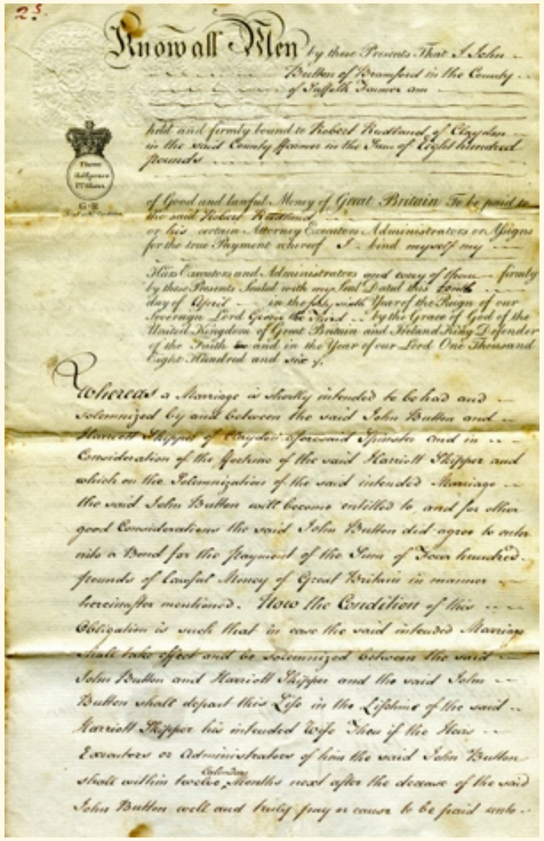
What Makes an API an API?

APIs have a contract

- WADL
- WSDL
- Web site
- Phone call…

APIs are used

- by other programmers



25

Know all Men by these Presents That I John Button of Bramford in the County of Suffolk Farmer am held and firmly bound to Robert Rudland of Claydon in the said County farmer in the Sum of Eight hundred pounds of Good and lawful Money of Great Britain To be paid to the said Robert Rudland or his certain Attorney Executors Administrators or Assigns for the true Payment whereof I bind myself my Heirs Executors and Administrators and every of them firmly by these Presents Sealed with my Seal Dated this tenth day of April in the fifty sixth Year of the Reign of our Sovereign Lord George the Third by the Grace of God of the United Kingdom of Great Britain and Ireland King Defender of the Faith and in the Year of our Lord One Thousand Eight Hundred and six [illegible]

Whereas a Marriage is shortly intended to be had and solemnized by and between the said John Button and Harriott Shipper of Claydon aforesaid Spinster and in Consideration of the fortune of the said Harriott Shipper and which on the Solemnization of the said intended Marriage the said John Button will become entitled to and for other good Considerations the said John Button did agree to enter into a Bond for the payment of the Sum of Four hundred pounds of lawful Money of Great Britain in manner hereinafter mentioned. Now the Condition of this Obligation is such that in case the said intended Marriage shall take effect and be solemnized between the said John Button and Harriott Shipper and the said John Button shall depart this Life in the Lifetime of the said Harriott Shipper his intended Wife Then if the Heirs Executors or Administrators of him the said John Button shall within twelve Calendar Months next after the decease of the said John Button well and truly pay or cause to be paid unto

Michael Jefferies

What Else?

APIs in 2011 use a consistent technology stack

- HTTP(s)
- OAuth
- JSON and/or XML
- Attributes of the
  - REST Architectural Style*

(*Not getting into that can of worms today, sorry)

Alan Berning

What’s Not an API?

- A bunch of XML web services you wrote for your mobile app
- A SOAP service hidden inside the corporate network

Does it have a contract so that others can use it?

If not, it’s not an API

Why a Platform for APIs?

APIs aren't web apps

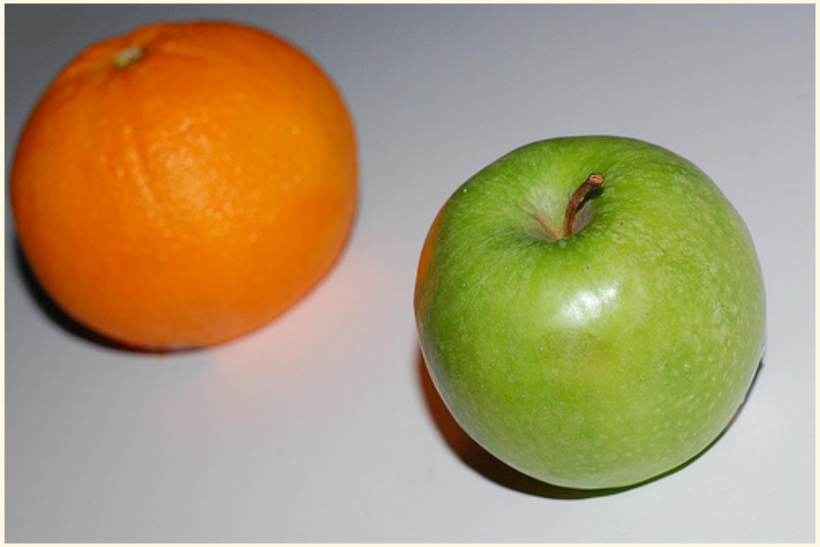


Image by lindzstrom

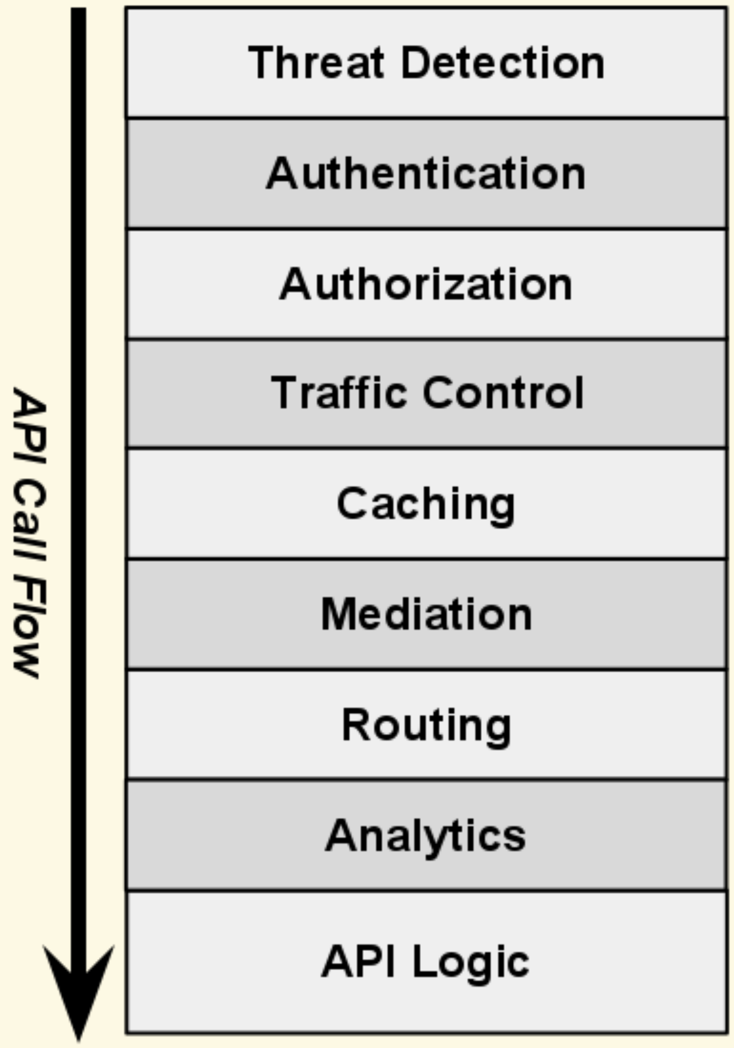
Web Apps vs APIs

Web App	**API**
For normal humans (great pattern-matchers)	For programs (lousy pattern-matchers)
Change is good	Incompatible change is bad
Password security	OAuth security
A few browser versions	Many different client devices
Client will run your scripts	Client won’t

What Goes in the API Platform?

- Authentication
- Authorization
- Audit
- Format mediation
- Transformation
- Caching
- Rate Limiting
- Threat Detection
- Content Customization
- Logging
- Analytics
- Developer “On-boarding”

Typical API Call Flow



Not all APIs use these exact steps…

but many do,

and usually in this order

Threat Detection

Is the API request an obvious security threat?

- Correct input format (JSON or XML validation)
- Obvious injection attacks
- Suspect IP address

HACKERS CAN TURN YOUR HOME COMPUTER INTO A BOMB

. . . & blow your family to smithereeens!

By RANDY JEFFRIES / *Weekly World News*

WASHINGTON — Right now, computer hackers have the ability to turn your home computer into a bomb and blow you to Kingdom Come — and they can do it anonymously from thousands of miles away!

Experts say the recent "break-ins" that paralyzed the Amazon.com, Buy.com and eBAY websites are tame compared to what will happen in the near future.

Computer expert Arnold Yabenson, president of the Washington-based consumer group National CyberCrime Prevention Foundation (NCPF), says that as far as computer crime is concerned, we've only seen the tip of the iceberg.

"The criminals who knocked out those three major online businesses are the least of our worries," Yabenson told *Weekly World News*.

"There are brilliant but unscrupulous hackers out there who have developed technologies that the average person can't even dream of. Even people who are familiar with how computers work have trouble getting their minds around the terrible things that can be done.

"It is already possible for an assassin to send someone an e-mail with an innocent-looking attachment connected to it. When the receiver downloads the attachment, the electrical current and molecular structure of the central processing unit is altered, causing it to blast apart like a large hand grenade.

"As shocking as this is, it shouldn't surprise anyone. It's just the next step in an ever-escalating progression of horrors conceived and instituted by hackers."

Yabenson points out that these dangerous sociopaths have already:

- Vandalized FBI and U. S. Army websites.
- Broken into Chinese military networks.
- Come within two digits of cracking an 87-digit Russian security code that would have sent deadly missiles hurtling toward five of America's major cities.

"As dangerous as this technology is right now, it's going to get much scarier," Yabenson said.

"Soon it will be sold to terrorists cults and fanatical religious-fringe groups.

"Instead of blowing up a single plane, these groups will be able to patch into the central computer of a large airline and blow up hundreds of planes at once.

"And worse, this e-mail bomb program will eventually find its way into the hands of anyone who wants it.

"That means anyone who has a quarrel with you, holds a grudge against you or just plain doesn't like your looks, can kill you and never be found out."

Sickos can wreak death and destruction from thousands of miles away!

Arnold Yabenson.

KABOOM! It might not look like it, but an innocent home computer like this one can be turned into a deadly weapon.

Authentication

- Are the security credentials valid?
  - Valid OAuth token
    - Both end user and application are authenticated
  - Correct password
  - Valid SSL cert
- Basis of the next processing steps…

Authorization

- Is the user allowed to make this API call?
    - Often based on OAuth "scope"
    - Lots of other ways to do this

Tom Hensel

Traffic Control

- Is the user or app allowed to make the call *now?*
  - Does it have quota?
    - Business-oriented limit on traffic
    - Based on user, application, or both
  - Is there an additional rate limit?
    - Operations-based limits based on IP, etc.
  - Do we need to restrict traffic anyway?
    - Prioritize to help an overloaded back end

Caching

- Did we cache the API response?
  - API response caching in the “last mile” can be very effective,
  - in addition to all the other caching layers:
    - App server
    - Database
    - Disk
    - CPU
    - Etc.

Mediation

- Did the server return what the client wants?
  - May need to convert otherwise
    - XML to JSON
    - SOAP to REST
  - A big issue when building APIs from legacy systems
- Is the client making many calls?
  - Aggregate on the server

Tom Natt

Routing

- Where should the API call go now?
  - Multiple tiers of servers
    - Sandbox
    - Production
    - Different databases, geographies

Analytics

- What are the API usage patterns?
  - Usage
  - Response Time
  - Error Rate
    - By application
    - By developer
    - By end user
    - By IP address

Business Logic

- Where does the actual API code run?
- *Now*, web apps and APIs *both* need:
  - Code deployment
  - Thread scheduling
  - Database connection management, etc.
- Sounds like an app server to me…
  - There are nice API-specific toolkits that run there
    - JAX-RS for Java, Sinatra for Ruby, etc.

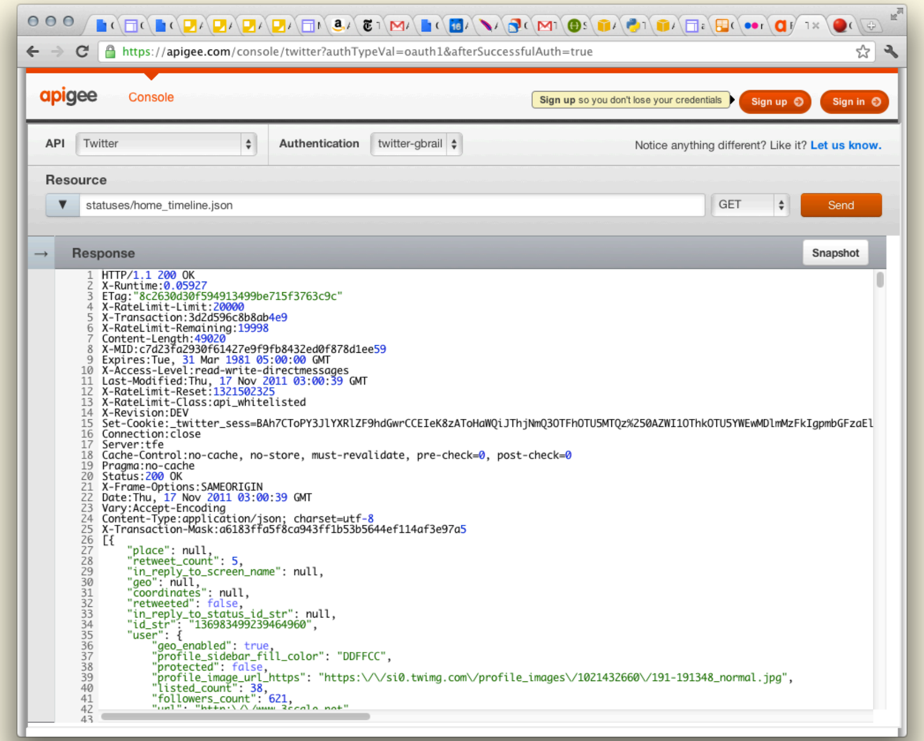
Don’t Forget

Docs and Test Console

- Remember that APIs are contracts
- and developers would rather test than read

Developer Sign-Up

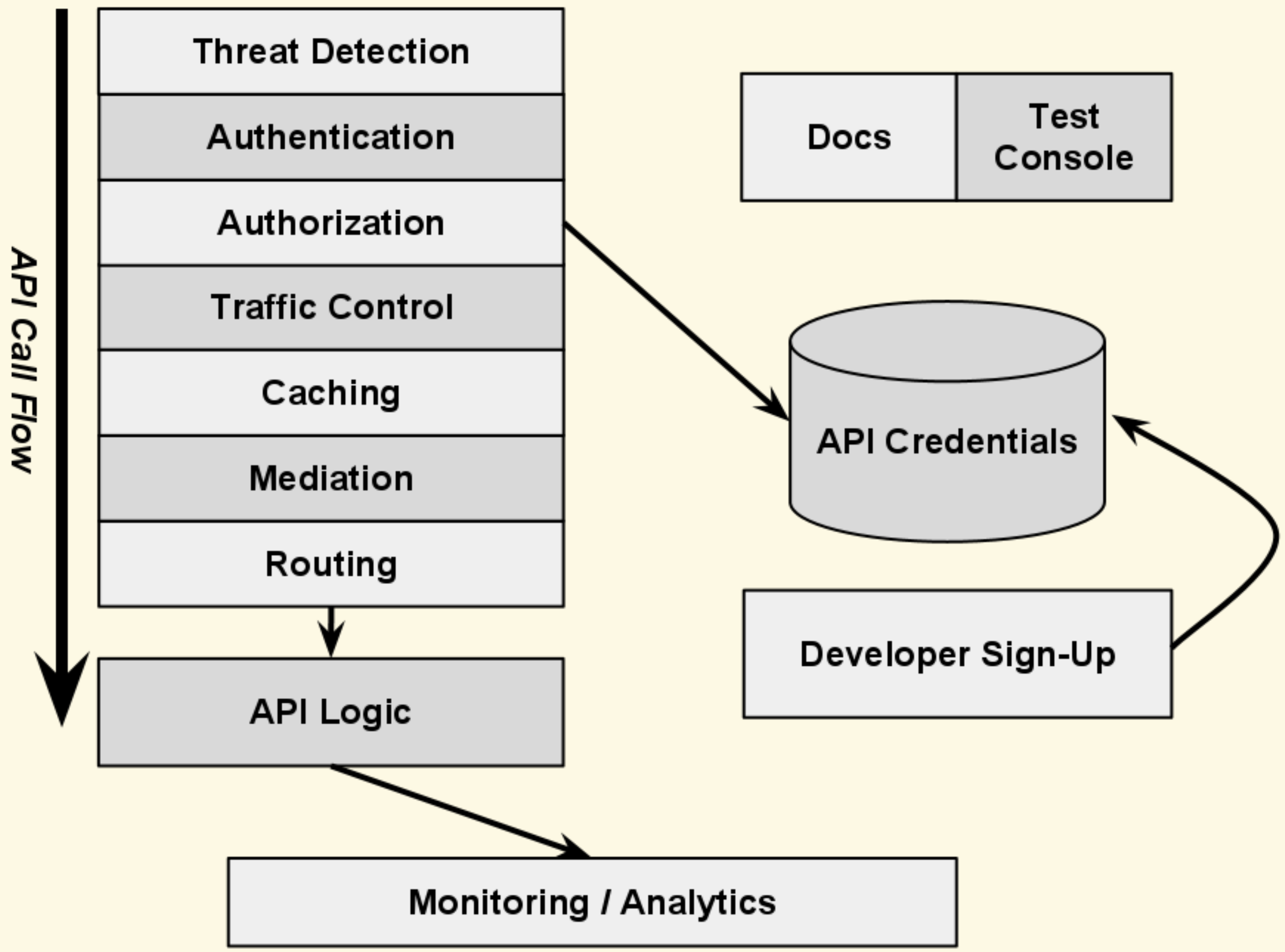
- They need credentials to build an app


https://apigee.com/console/twitter?authTypeVal=oauth1&afterSuccessfulAuth=true
apigee
Console
Sign up so you don't lose your credentials
Sign up
Sign in
Notice anything different? Like it? Let us know.
Resource
Send
Response
Snapshot

API: Twitter
Authentication: twitter-gbrail
Resource: statuses/home_timeline.json
GET

```
HTTP/1.1 200 OK
X-Runtime:0.05927
ETag:"8c2630d30f594913499be715f3763c9c"
X-RateLimit-Limit:20000
X-Transaction:3d2d596c8b8ab4e9
X-RateLimit-Remaining:19998
Content-Length:49020
X-MID:c7d23fa2930f61427e9f9fb8432ed0f878d1ee59
Expires:Tue, 31 Mar 1981 05:00:00 GMT
X-Access-Level:read-write-directmessages
Last-Modified:Thu, 17 Nov 2011 03:00:39 GMT
X-RateLimit-Reset:1321502325
X-RateLimit-Class:api_whitelisted
X-Revision:DEV
Set-Cookie:_twitter_sess=BAh7CToPY3JlYXRlZF9hdGwrCCEIeK8zAToHaWQiJThjNmQ3OTFhOTU5MTQz%250AZWI1OThkOTU5YWEwMDlmMzFkIgpmbGFzaEl
Connection:close
Server:tfe
Cache-Control:no-cache, no-store, must-revalidate, pre-check=0, post-check=0
Pragma:no-cache
Status:200 OK
X-Frame-Options:SAMEORIGIN
Date:Thu, 17 Nov 2011 03:00:39 GMT
Vary:Accept-Encoding
Content-Type:application/json; charset=utf-8
X-Transaction-Mask:a6183ffa5f8ca943ff1b53b5644ef114af3e97a5
[{
    "place": null,
    "retweet_count": 5,
    "in_reply_to_screen_name": null,
    "geo": null,
    "coordinates": null,
    "retweeted": false,
    "in_reply_to_status_id_str": null,
    "id_str": "136983499239464960",
    "user": {
        "geo_enabled": true,
        "profile_sidebar_fill_color": "DDFFCC",
        "protected": false,
        "profile_image_url_https": "https:\/\/si0.twimg.com\/profile_images\/1021432660\/191-191348_normal.jpg",
        "listed_count": 38,
        "followers_count": 621,
        "url": "http:\/\/www.3scale.net"
```

Abstract API Platform



Where is the API Platform?

- Built in to the application server
  - Code libraries
  - Less complex, no latency
- Or, an additional network tier
  - Can combine many APIs into one
  - Offloads rate limits, security, etc.
  - Additional latency
    - But not as much as you think

For Example

A company may start with a simple API

- Manual developer sign-up, simple authentication

They may want to open to a larger audience

- Developer portal for sign up
- OAuth
- Quotas / rate limits
- Caching

An API platform enables this change

Another Example

- A company may have many web services
  - But they are not suitable for use by API developers
  - Bad design, incompatible security, etc.
- An API platform can virtualize these services
  - Route to the right service
  - Transform requests and responses
  - Enforce consistent security and rate limits

Interesting Challenges

OAuth

Rate Limiting

Performance

OAuth >= LDAP

Authentication should be easy to scale

1. Deploy LDAP
2. Replicate it all over the place
3. Presto!

This Used to be Fine

- Runs at human speed
- Expectations were low
  - Change my password every once in a while
  - It usually works within a minute or two
  - Sometimes it doesn’t and that’s OK

- Works fine with a single master
  - Read-only replication for scalability

OAuth is Different

- Runs at device speed
- Lots of API calls, little latency
  - Create “unauthenticated request token” and *immediately* use it
  - Validate password and expect it to work right away

- Today’s apps demand uptime
  - It’s not OK for “OAuth to be down right now”

Big OAuth

Lots of data centers

Thousands of requests / second

Hundreds of new OAuth tokens / second

No time for downtime

Enter CAP

Consistency

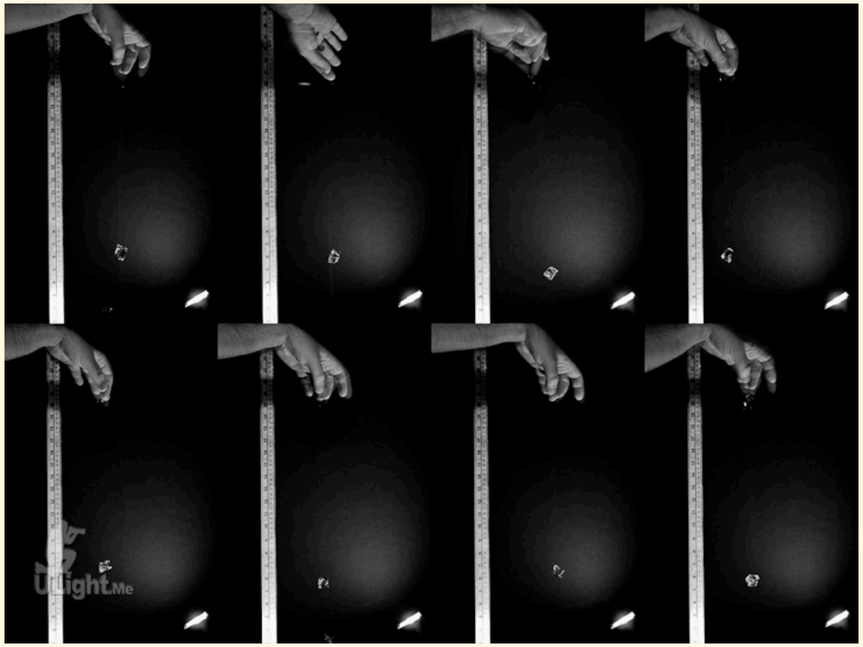
Availability

Partition Tolerance

Pick Two

See Eric Brewer, 2000:

http://en.wikipedia.org/wiki/CAP_theorem



Rafal Kiermacz

Naïve Solution

- OAuth tokens go in a database
- Read-only replicas for scalability

If the master goes down,

No new tokens are issued

Many new API clients fail

Another Solution

Use a data store that supports "AP"

… an "eventually consistent key-value store"

Cassandra

Riak

Voldemort

“AP” Solution

- OAuth tokens are simple
  - A unique, random number
  - Very simple lifecycle
    - Create
    - Invalidate
    - Refresh (Sometimes)
  - In other words, inconsistency can be handled

“AP” Database Example

Three or more database servers

Three copies of every OAuth token *(N = 3)*

Try to write new tokens to all *(W >= 1)*

Read from one *(R = 1)*

Retry if token is not found *(R = 3)*

Controlling API Traffic

Frequent customer request:

API calls == money

Customers buy “buckets” of calls

Want 100 percent consistency

Want 99.99 percent availability

This is a Hard Problem

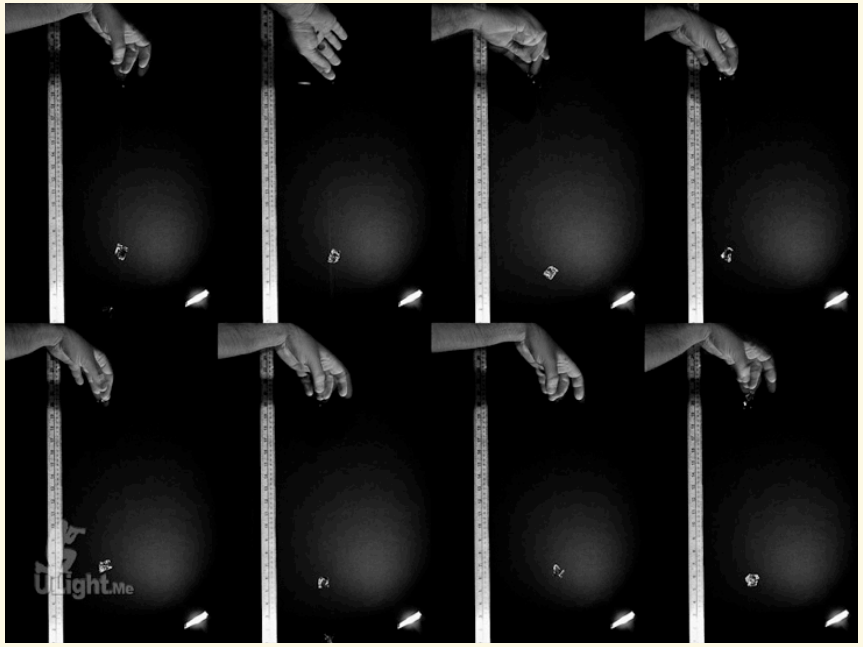
Back to the CAP Theorem

Consistency

Availability

Partition Tolerance

Pick Two



Rafal Kiermacz

Synchronous Server Solution

- One copy of each customer’s API quota
  - One big database, cache, with sharding
  - Fail over if it breaks
- 100 percent consistent
- Not partition-tolerant
- Performance limited by database / cache

Asynchronous Server Solution

Central "quota server"

Each node calculates local quota

- Communicates with the quota server out of band

Less consistent

- Because we sync quotas asynchronously

Partition-tolerant

Faster too

“AP” Quota Solution

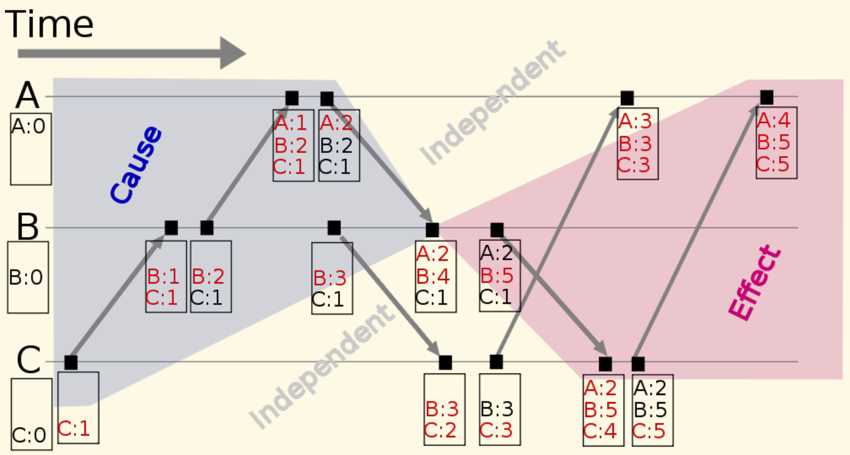
Take advantage of vector clocks

Available and partition-tolerant

More consistent than a fully asynchronous one

Good enough

for *most* tasks



http://wiki.apache.org/cassandra/Counters

Performance: The API Frontier

- Our API platform includes a proxy server
  - Lots in common with an app server
    - Threads, concurrency, HTTP processing
  - But also not
    - A proxy passes data *through* and may not change it
    - An app server returns a totally different response
- Pop Quiz: Which one will be faster?
  - C + Java + custom FPGA
  - Java

Making APIs Perform

I love solving API performance problems

- Lots of concurrency
- High throughput
- Low latency

Making APIs Perform

- I hate solving API performance problems
  - Lousy test tools
  - Lousy test methodology
  - Lousy locations for test clients
  - Slow back end servers
  - Blaming the new guy

What I’ve Learned

- Start with a fast test client
  - JMeter is too slow
  - “ab” is great but limited
  - Others are complex
  - My entry: http://code.google.com/p/apib
- Start with a fast back end
  - At least for testing
  - Apache httpd can reply 75,000+ times / second…

What Else I’ve Learned

- Most people do it wrong
  - Test run is too short (less then 5 seconds!)
  - Test is returning 404 25% of the time
  - No one measured CPU usage
  - No one calculated network bandwidth
  - etc.

And also

- Isolate, baby, isolate
  - Turn everything *off*
  - Get a baseline
  - Then turn things back on one at a time…
- Measure, baby, measure
  - The thing you think is awful isn’t that bad
  - The thing you don’t know about is awful

Conclusion

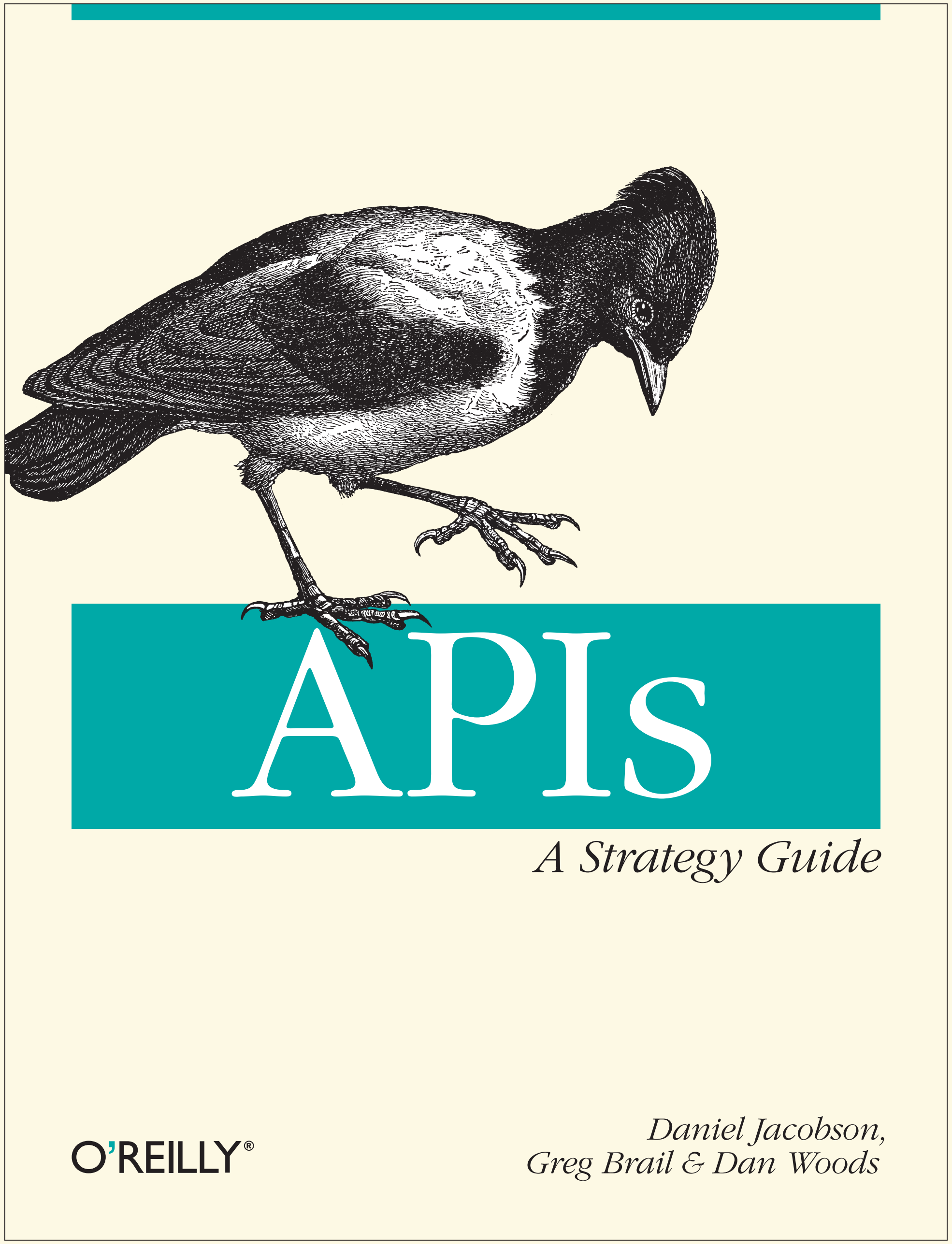
APIs have distinct needs

A dedicated “API Platform” eases the burden

There are lots of ways to do this

And lots of challenges along the way

Thanks!



APIs

A Strategy Guide

O'REILLY®

Daniel Jacobson, Greg Brail & Dan Woods

THANK YOU

Questions and ideas to:

@gbrail