Serverless Content Delivery

@johnchapin | symphonia.io

github.com/symphoniacloud/oscon-2018-static-content

John Chapin

- Currently Partner, Symphonia
- Former VP Engineering, Technical Lead
  - Data Engineering and Data Science teams
- 20+ yrs experience in govt, healthcare, travel, and ad-tech
- Intent Media, RoomKey, Meddius, SAIC, Booz Allen

Symphonia resources

- **github.com/symphoniacloud/oscon-2018-static-content**
- **github.com/symphoniacloud/lambda-monitoring** - Open source logging/monitoring library for Lambda
- **What is Serverless?** Our 2017 report, published by O'Reilly
- **Programming AWS Lambda** - Our upcoming full-length book with O'Reilly.
- **Serverless Architectures** - Mike's de facto industry primer on Serverless.
- **Learning Lambda** - A 9-part blog series to help new Lambda devs get started.
- **Serverless Insights** - Our email newsletter covering Serverless news, event, etc.
- **blog.symphonia.io** - The Symphonium (our blog), featuring technical content and analysis.

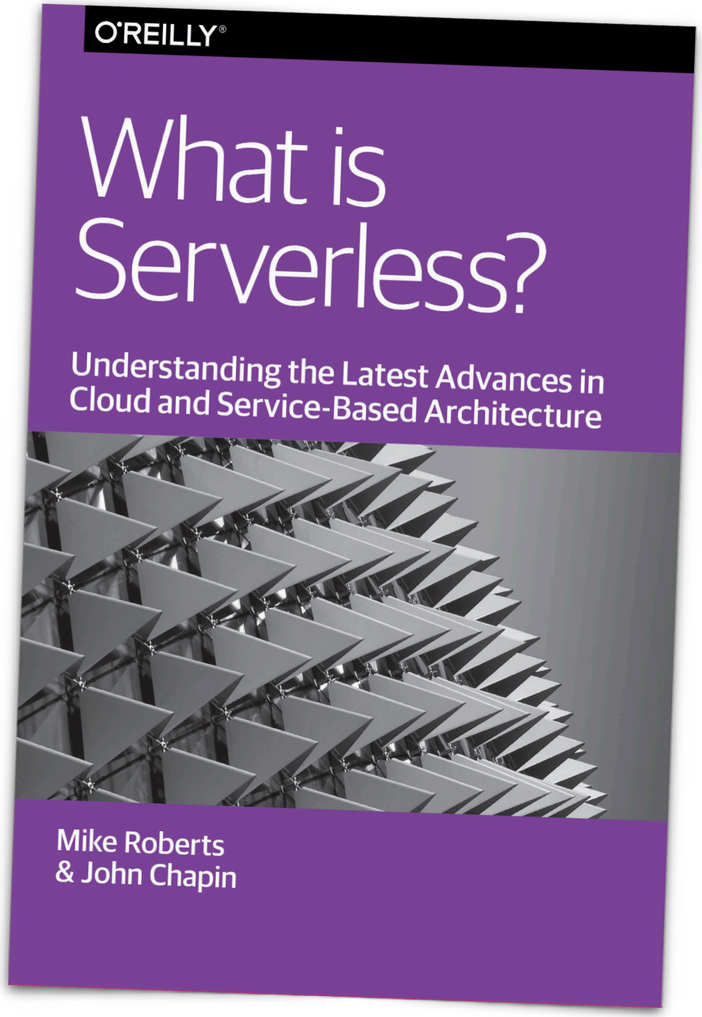
Agenda

- What is Serverless?
- Static content using S3 + CloudFront
- Custom domains using Route 53
- SSL using AWS Certificate Manager
- Logic on the edge using Lambda@Edge
- Discussion

What is Serverless?

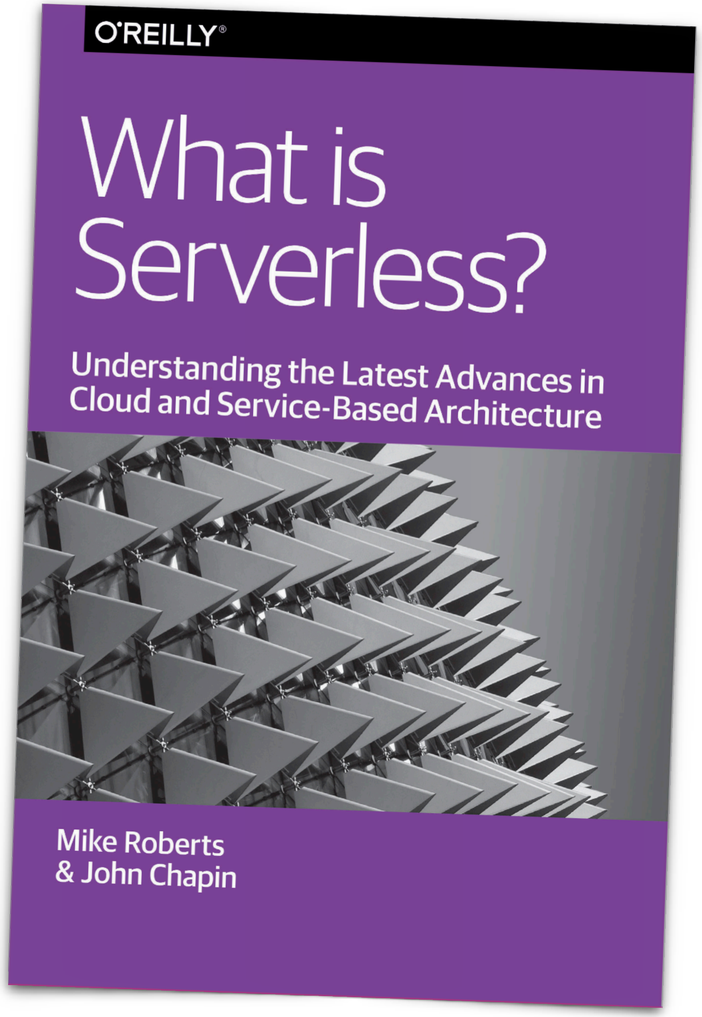
Serverless benefits

- Free O'Reilly report!
- Cloud benefits ++
  - Reduced cost
  - Scaling flexibility
  - Shorter lead time



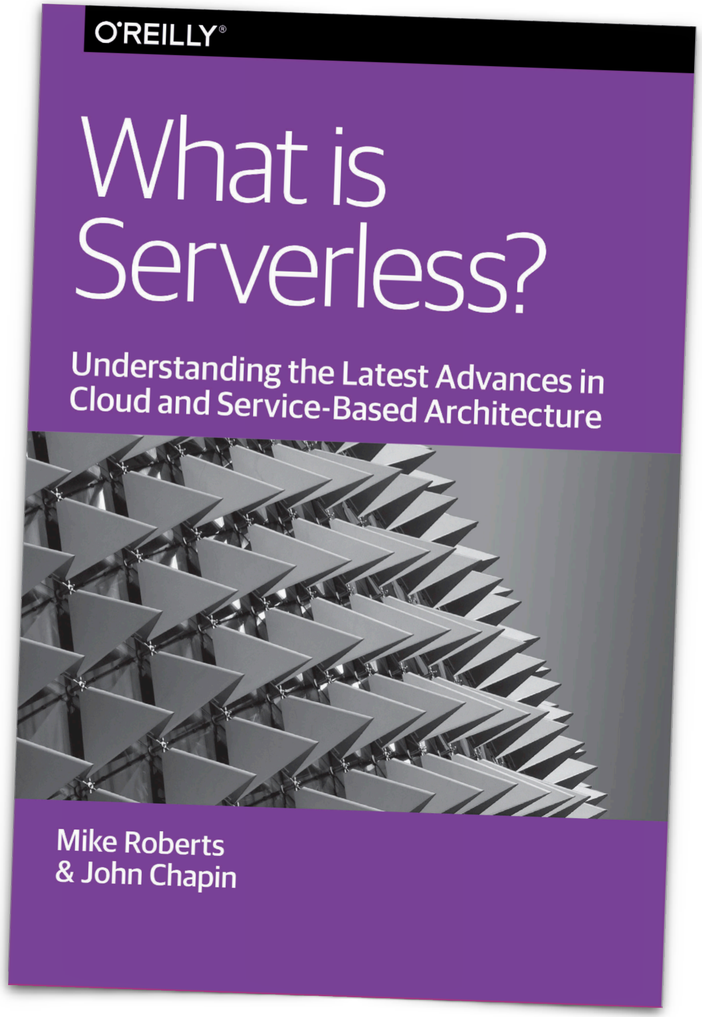
Serverless attributes

- No managing of hosts or processes
- Self auto-scaling and provisioning
- Costs based on precise usage
- Performance specified in terms other than host size/count
- Implicit high availability



Serverless = FaaS + BaaS!

- Same benefits and attributes!
- FaaS = **F**unctions **a**s **a** **S**ervice
  - AWS Lambda, Auth0 Webtask, Azure Functions, Google Cloud Functions, etc...
- BaaS = **B**ackend **a**s **a** **S**ervice
  - Auth0, Google Firebase, Parse, Amazon **CloudFront**, DynamoDB, **S3**, etc...



The Original Serverless Service!

S3 overview

- Simple Storage Service
- Launched in March 2006
- Key/value store, optimized for large amounts of data
- 99.999999999% durability (given 10k objects, you'll lose one every 10M years)
- 99.99% availability (4.38 minutes of downtime per month)
- Resource-level access control via ACLs, bucket policies

Hosting a website on S3

- Create an S3 bucket
- Upload content via CLI (or API)
- Set bucket policy allowing public read
- *CORS - Allow Javascript cross-origin requests*
- *CNAME - Use a custom domain with your S3 bucket (names must match)*

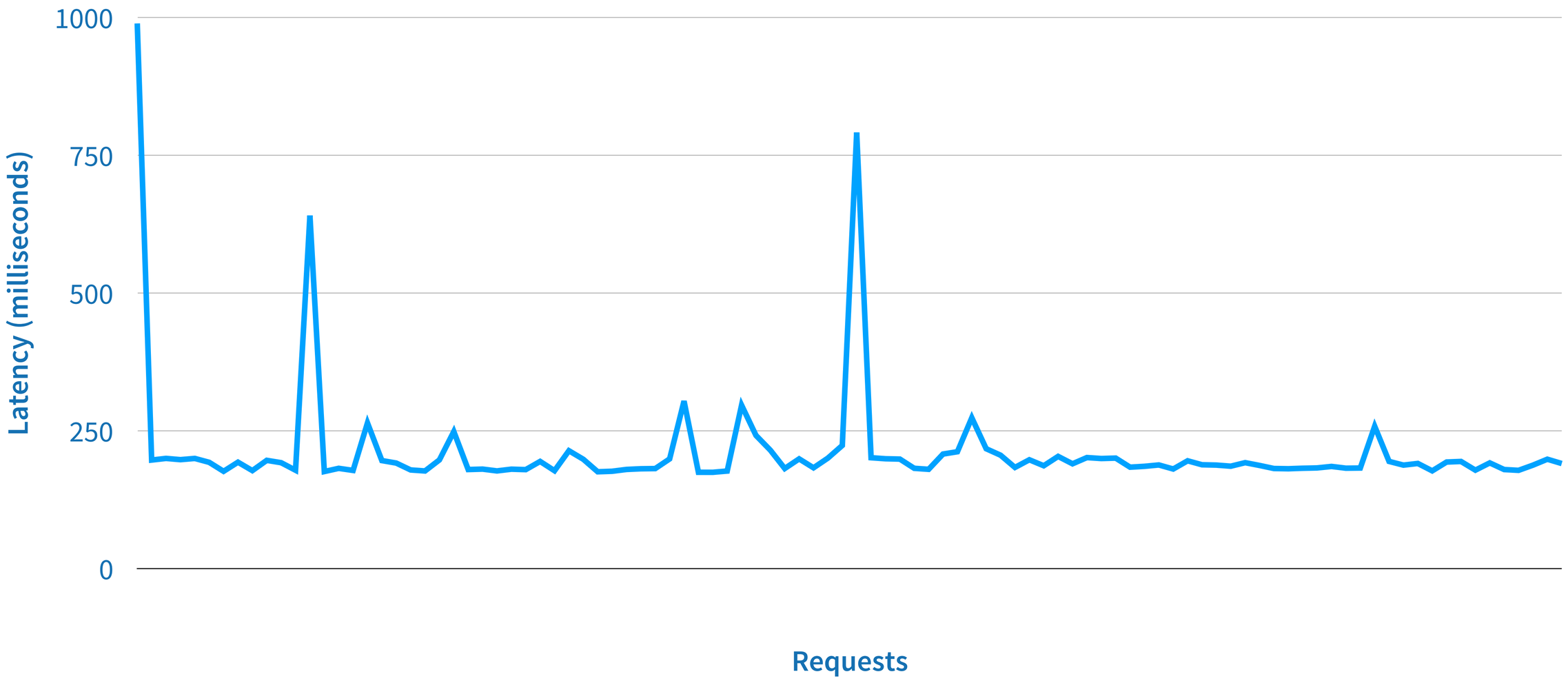
S3 website demo

http://oscon-static-bucket-v6lev7k3j0ts.s3-website-us-east-1.amazonaws.com

S3 is Serverless!

- No servers (from our perspective)
- Pay by the request (and the byte)
- Highly available (intra-region)
- Highly scalable

S3 latency (milliseconds)



S3 website caveats

- S3 is relatively slow
- S3 is regional, so requests must go to the regional data center
- Custom domain require a specific bucket name
  - S3 global bucket namespace == squatting, requires support intervention
- S3 request/transfer pricing

S3 website pricing example (us-east-1)

- 10GB data storage ($0.23)
- 100,000,000 HTTP GET requests ($40)
- 10TB data transfer out to the internet ($921)
- $961/month

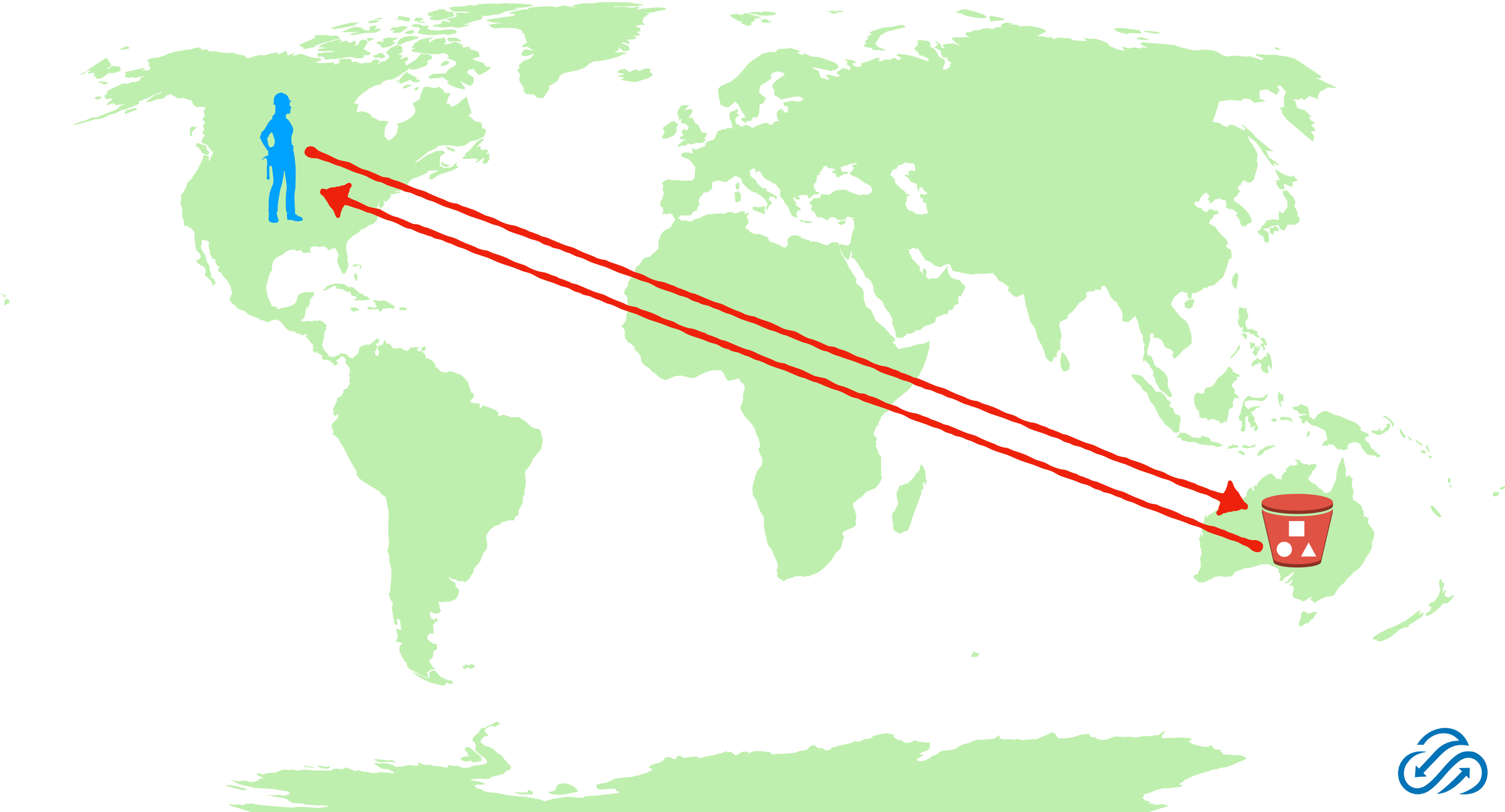
Serverless to the rescue!

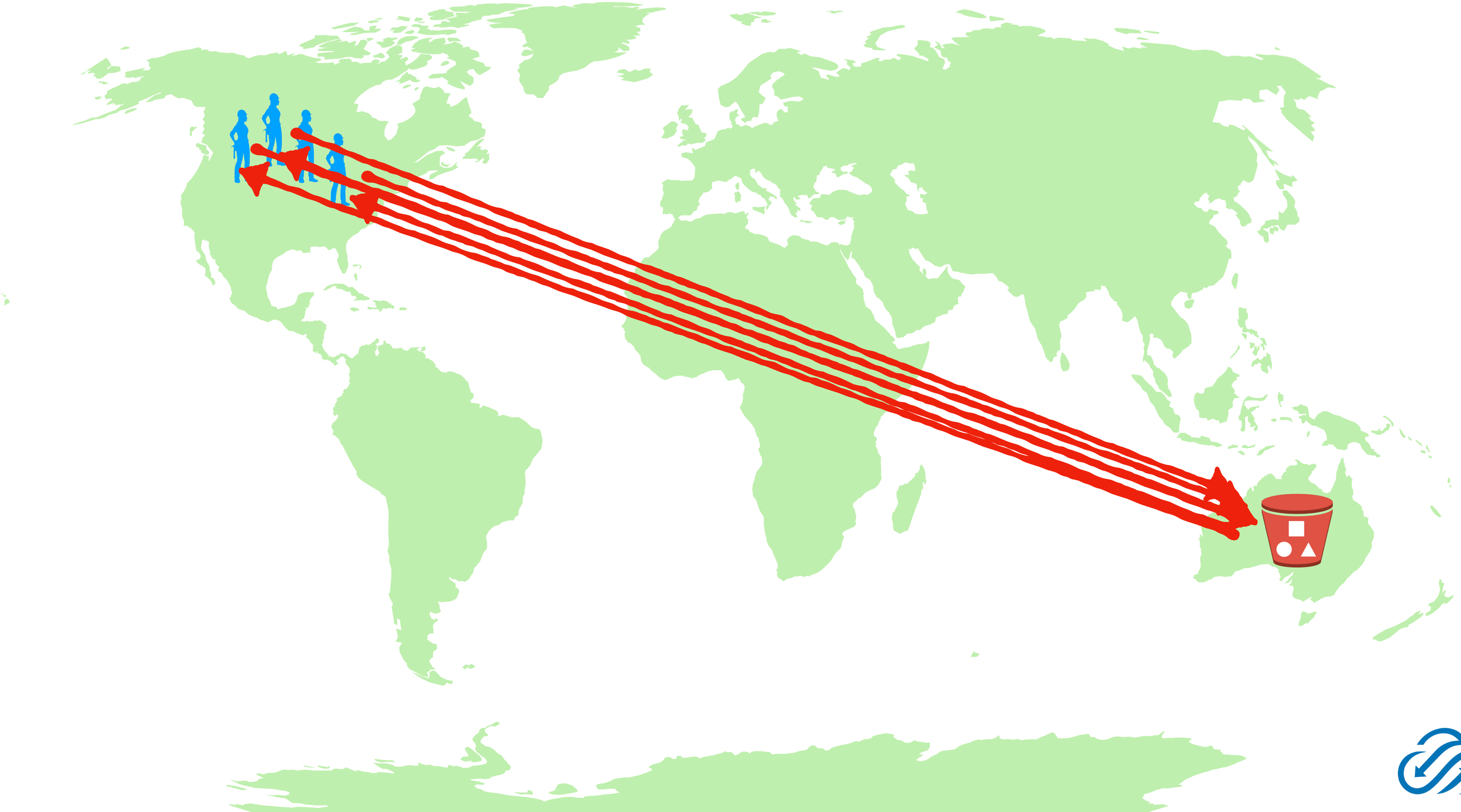
CloudFront basics

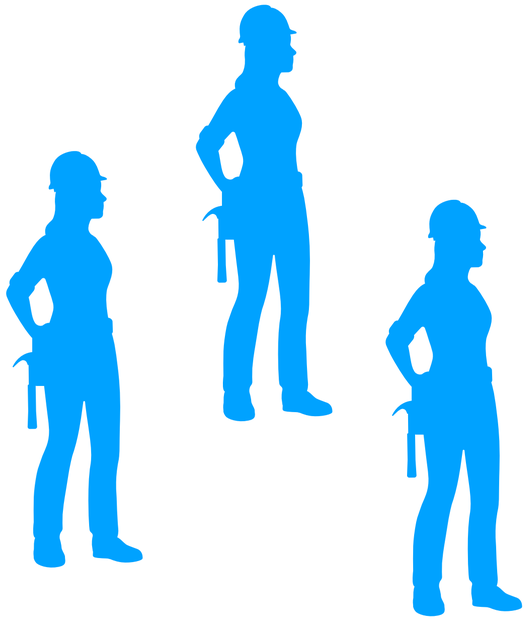
- Content delivery network (CDN)
- Launched in November, 2008
- ~~96~~ 112 edge locations, 11 regional caches
- Faster for serving static content, cheaper for bandwidth
- More difficult to update, requests and invalidations cost $

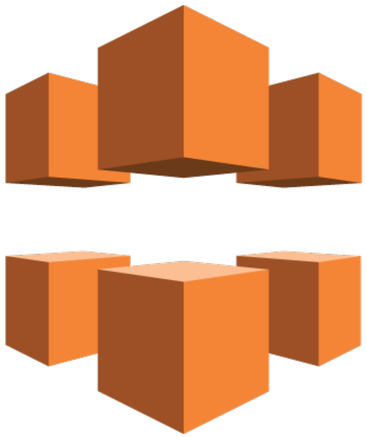
S3 + CloudFront demo

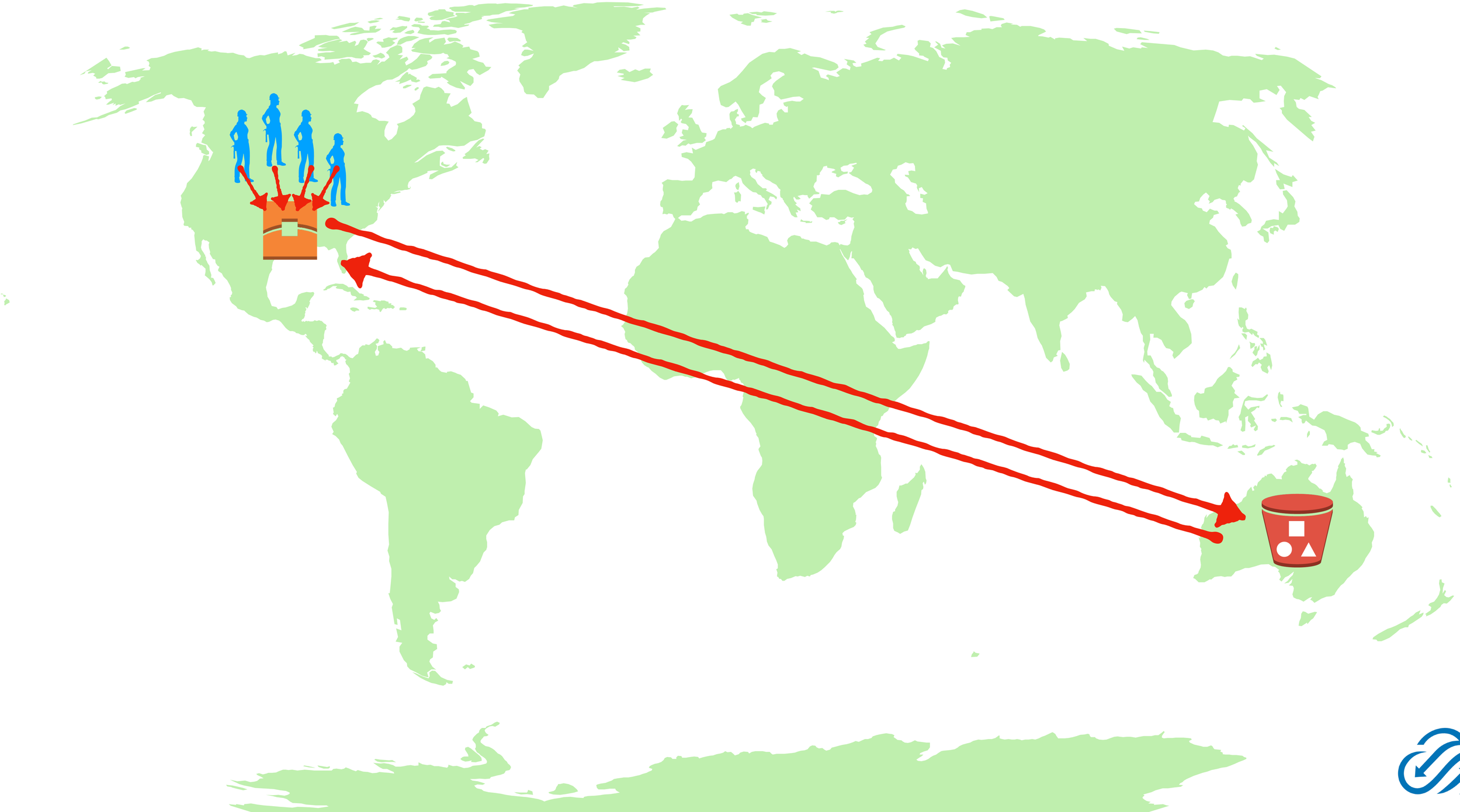
http://d18k0jpkksinsd.cloudfront.net

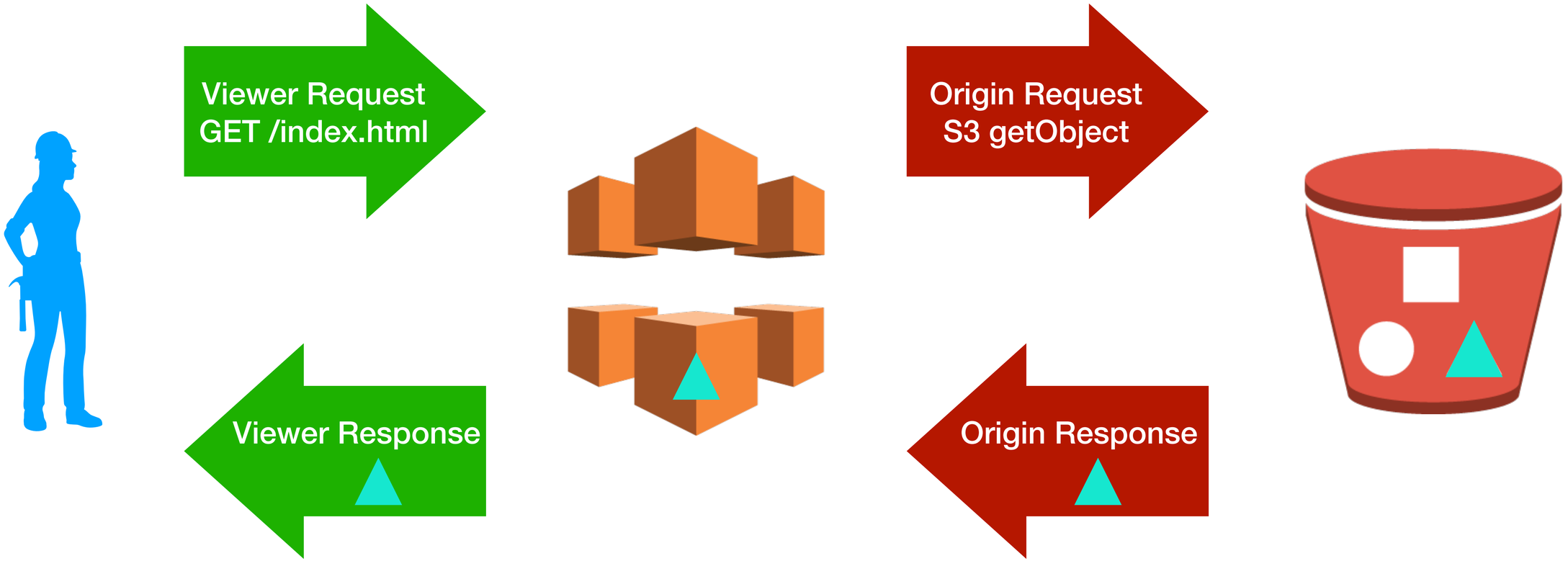


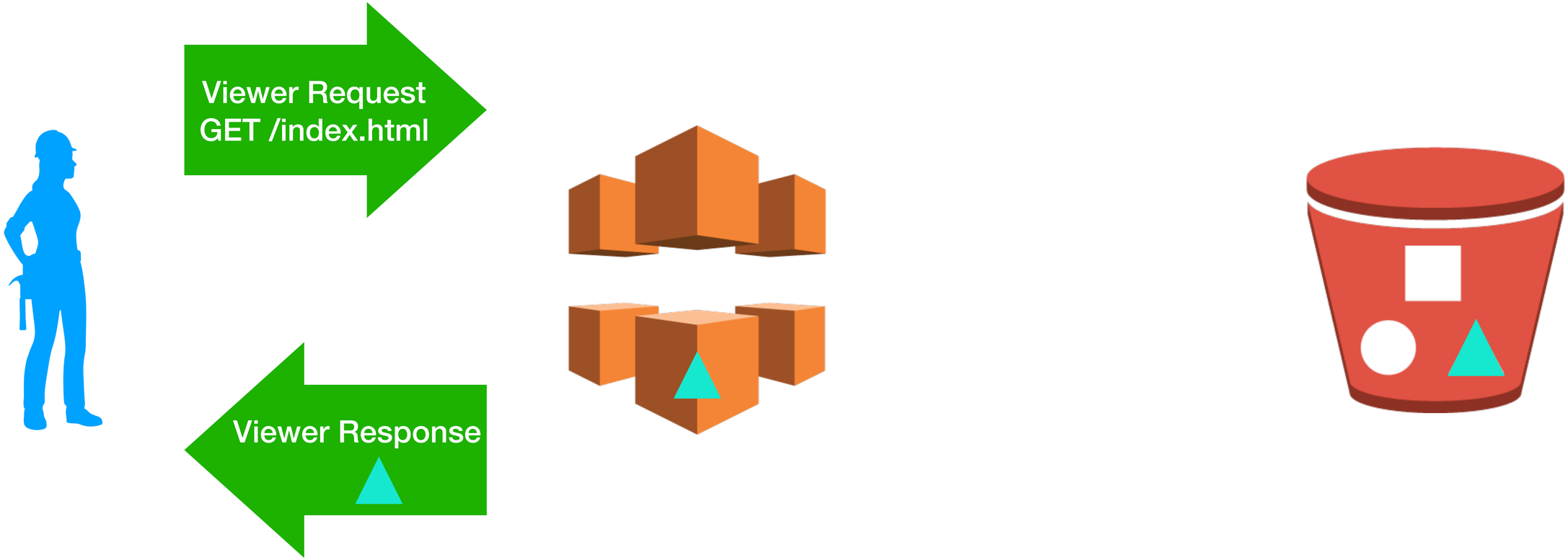




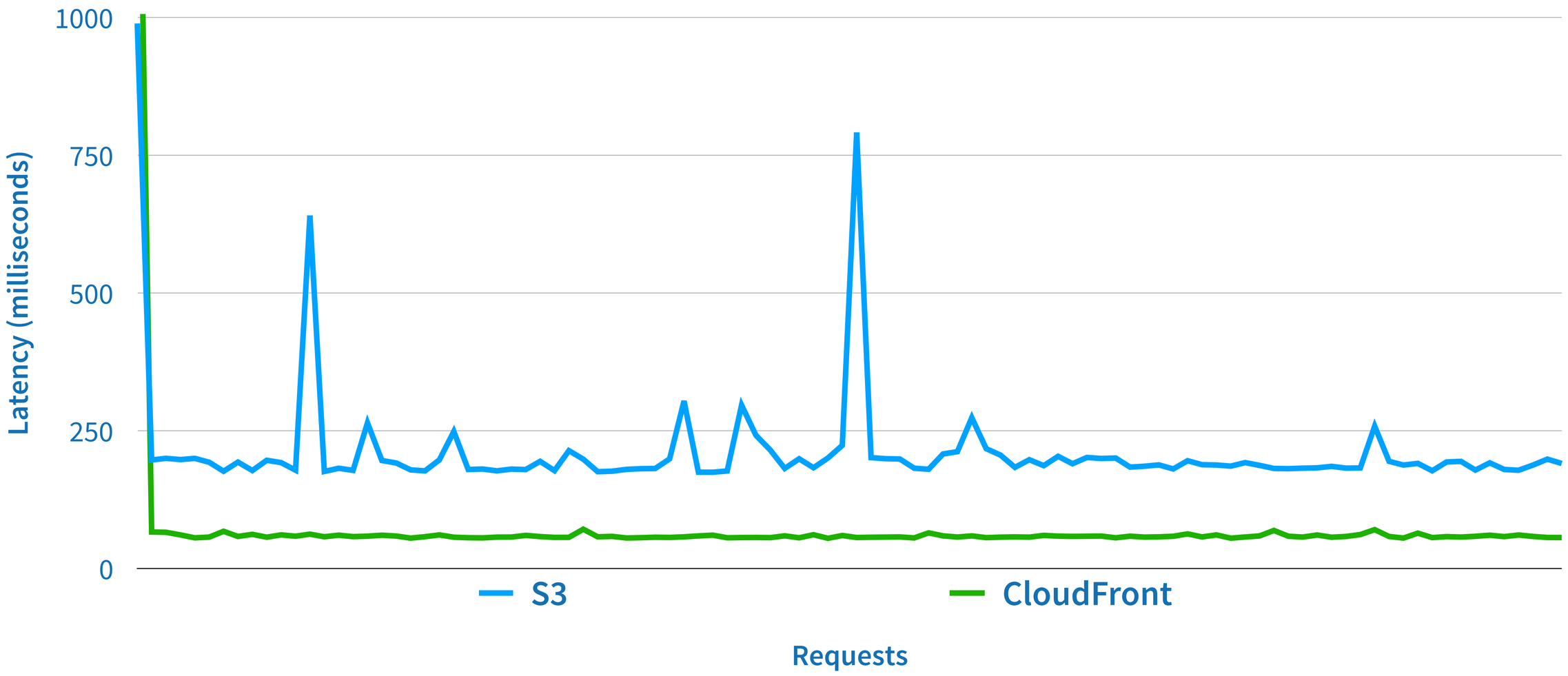





Viewer Request
GET /index.html
Origin Request
S3 getObject
Viewer Response
Origin Response


Viewer Request
GET /index.html
Viewer Response

S3 vs CloudFront latency



S3 vs CloudFront latency

	Min	Max	Average	Std Dev
S3	175 ms	791 ms	206 ms	78 ms
CloudFront	55 ms	71 ms	59 ms	3 ms

Aesthetics

- S3 URLs (especially for auto-generated buckets) can be very long:
  - http://oscon-static-bucket-v6lev7k3j0ts.s3-website-us-east-1.amazonaws.com
- CloudFront URLs are shorter, but ugly:
  - http://d18k0jpkksinsd.cloudfront.net

Serverless to the rescue (again)!

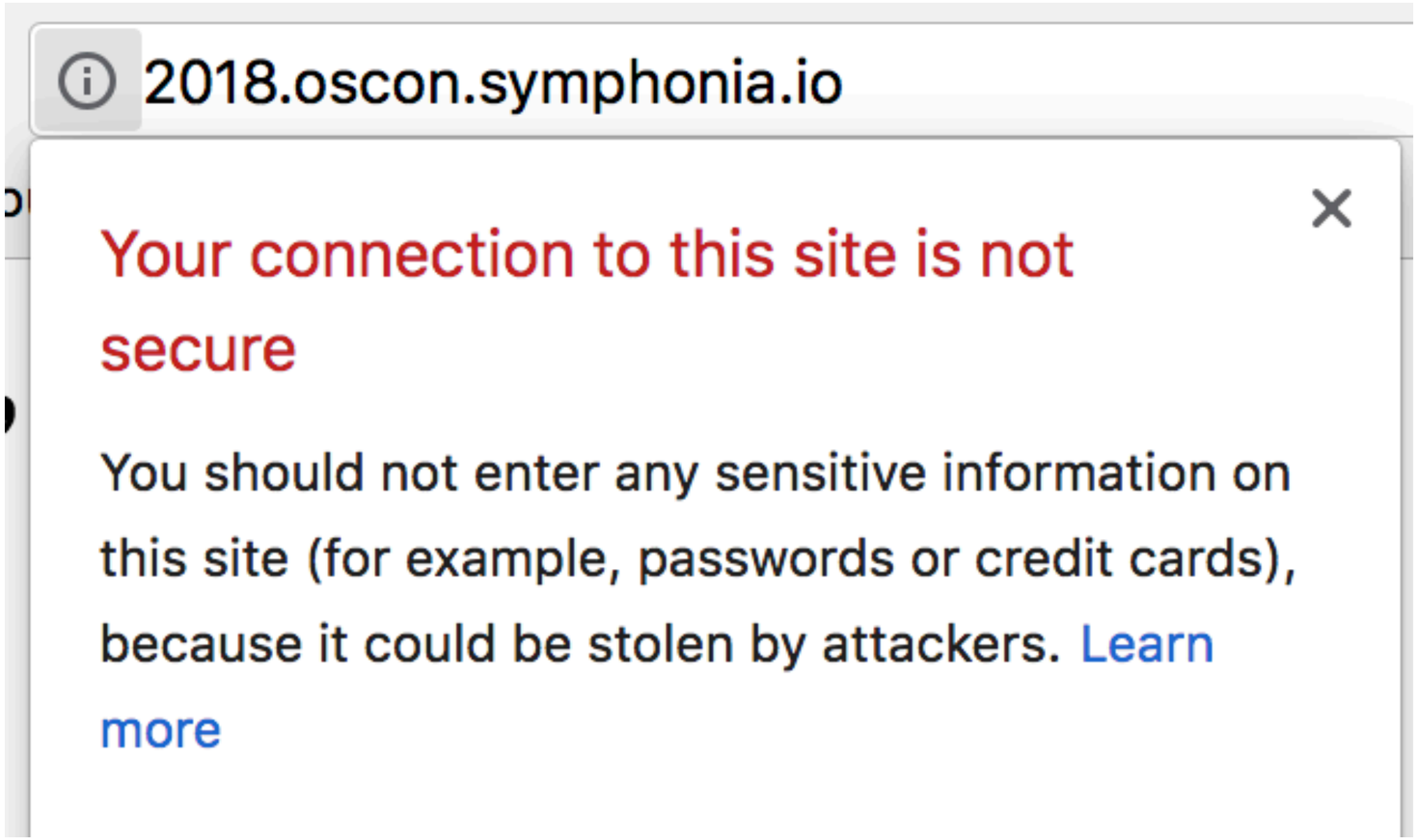
Route 53 basics

- Managed "Cloud" DNS
- "100% Available" (https://aws.amazon.com/route53/sla/)
- Capabilities like...
  - Health checks / failover
  - Round-robin
  - ALIAS records (pointers to AWS resources)

CloudFront + Route 53 demo

http://2018.oscon.symphonia.io

This still isn't up to 2018 standards...


2018.oscon.symphonia.io
Your connection to this site is not secure
You should not enter any sensitive information on this site (for example, passwords or credit cards), because it could be stolen by attackers. Learn more

_____ *to the rescue!*

ACM basics

- AWS Certificate Manager
- Managed SSL/TLS certificates
- API-driven
- Human-in-the-loop for verification
- Integrated with CloudFront, Elastic Load Balancer, API Gateway
- Supports wildcard domains

CloudFront + SSL demo

https://2018.oscon.symphonia.io

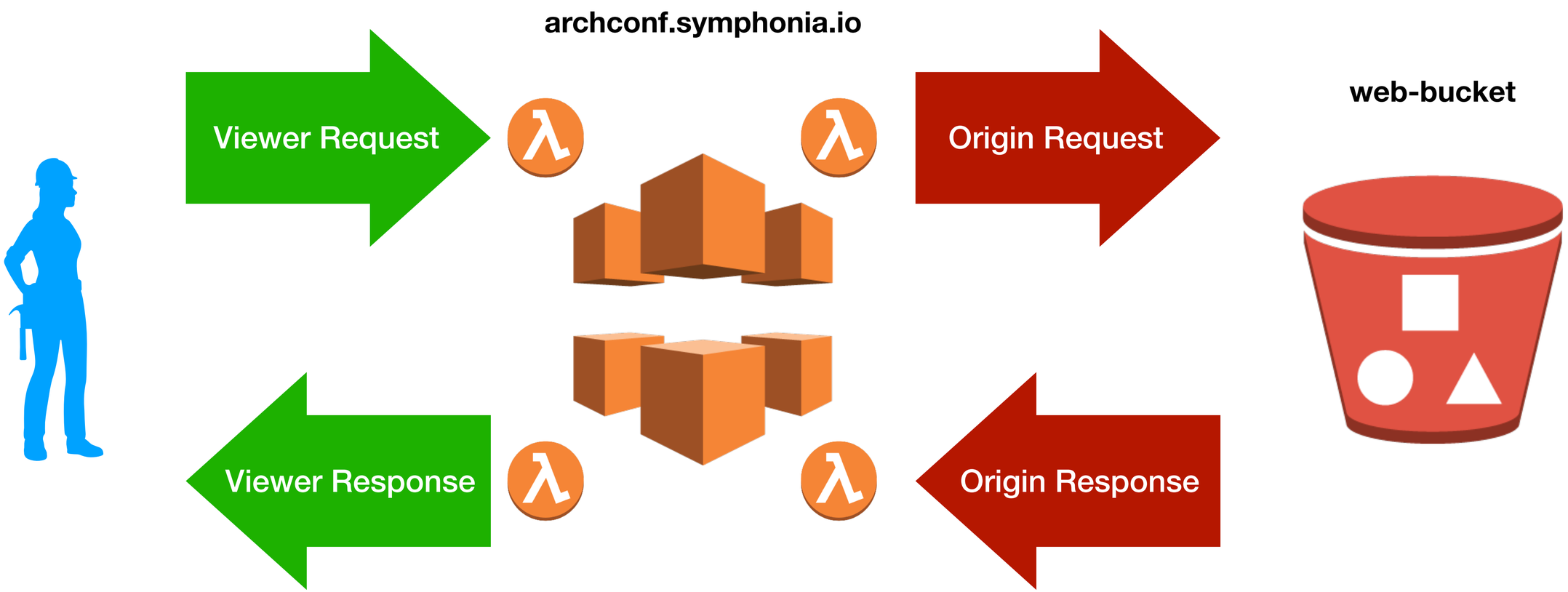
S3 + CloudFront + Route 53 + ACM

- S3 stores our content
  - No restrictions on bucket name
- CloudFront distributes it to edge locations
  - 70% lower latency than S3
- Route 53 adds custom domain names
- AWS Certificate Manager adds transport security

It wouldn't be a Serverless presentation without...

Lambda@Edge basic

- A "flavor" of Lambda that runs *in* CloudFront!
- Hooks into viewer and origin request/response events
- Node.js runtime only
- Viewer request/response has limited capability (5 secs runtime, 128MB)
- Origin request/response capabilities like "normal" Lambda


archconf.symphonia.io
Viewer Request
Origin Request
web-bucket
Viewer Response
Origin Response

Lambda@Edge demo

https://2018.oscon.symphonia.io/secure/secret.html

The final configuration

- S3 + CloudFront + Route 53 + ACM as described earlier
- Lambda@Edge function on "viewer-request"
- CloudFront distribution, two origins
  - Pass OriginAccessIdentity to secure origin
- S3 Bucket policy to lock down "/secure" path using OriginAccessIdentity

Discussion

Discussion topics

- General Serverless
  - Testing / debugging / tracing
  - Vendor lock-in
  - BaaS-only applications
  - Cost management
  - Deployment
- Serverless Content Delivery
  - CloudFront vs other CDNs
  - Lambda@Edge limitations
  - API Gateway
  - CloudFormation delays
  - S3 vs CloudFront costs

Stay in touch!

john@symphonia.io

@johnchapin

@symphoniacloud

symphonia.io/events

blog.symphonia.io