

Making Your ColdFusion Apps Highly Available

Brian Klaas

Johns Hopkins Bloomberg School of Public Health

bklaas@jhsph.edu

[@brian_klaas](https://twitter.com/brian_klaas)



JOHNS HOPKINS
BLOOMBERG
SCHOOL *of* PUBLIC HEALTH

How much downtime
can you afford?

99% = 14 minutes/day

99.9% = 1.4 minutes/day

99.999% = 8.6 seconds/day

99.9999% = 1 second/day



How much availability
can you afford?

Um, hey,
I'm not Google.

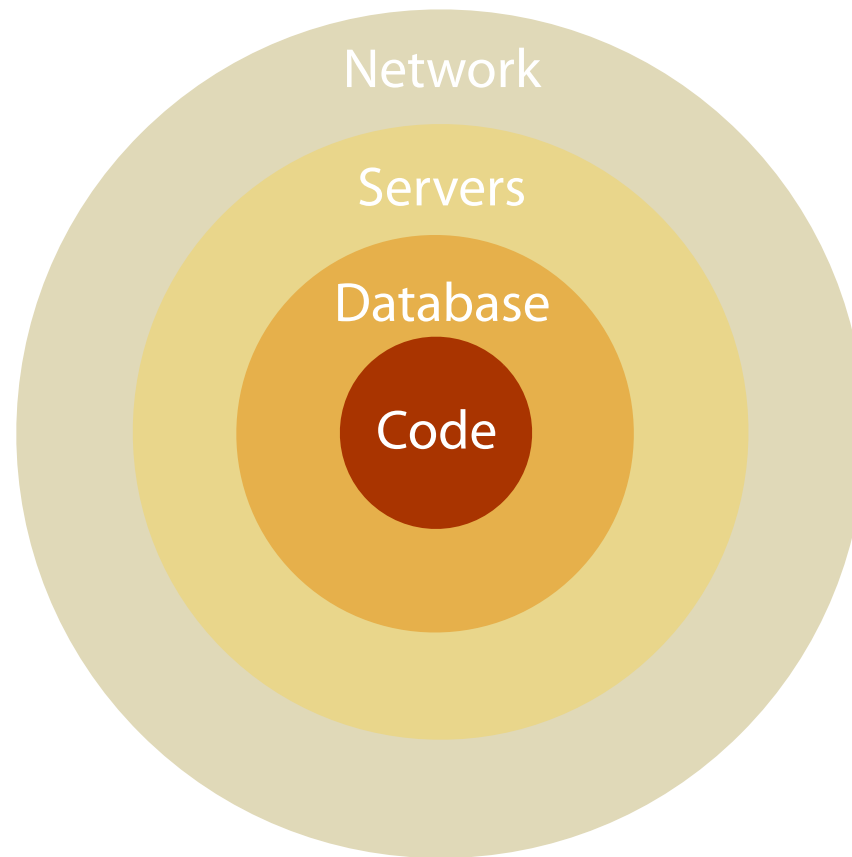
Scale out



Replication + Redundancy
= Availability

Virtualization is your friend.

Changes



Changes to App Code

```
enrollmentInfoNameInCache = "mc4getEnrollmentByUser" & arguments.  
cachedEnrollmentInfo = cacheGet(enrollmentInfoNameInCache)>  
is isn't already in the CF Object Cache or this is an enrollment  
(cachedEnrollmentInfo) OR (arguments.enrollDate IS "1/1/1990"  
iteCompare(arguments.enrollDate, Now()) NEQ 0>  
let maxBackDate = arguments.enrollDate />  
  
/ name="qryEnrollment" datasource="#variables.dsn#"  
:CT enrollmentID, stuID, userLevel, firstName, lastName, email  
:name, startDate, endDate, courseID, courseTitle, courseOffer  
:tAccessDate, extendAccessDate as endAccessDate, lastModified  
: vEnrollment2003  
IE stuID = <cfqueryparam cfsqltype="cf_sql_integer" value="#:  
AND (  
    (endDate >= <cfqueryparam cfsqltype="cf_sql_date" value=  
    OR (extendAccessDate >= <cfqueryparam cfsqltype="cf_sql_  
ite(Now())#)">  
    <!-- Also grab situations where the person is course fa  
        "perpetual" access like course faculty, of record. -->  
        OR (userLevel = 250 and startDate >= DateAdd(month, -10,  
    )  
:R BY startDate DESC  
:y>  
returnQuery = qryEnrollment />  
culty need to have their course listed ad infinitum, so we r  
the courses on which the user is primary faculty -->  
/ name="qryGetFacultyRole" datasource="#variables.dsn#"  
:CT courseID  
: CourseFaculty  
IE stuID = <cfqueryparam cfsqltype="cf_sql_integer" value="#:  
:y>  
yGetFacultyRole.recordCount GT 0>  
let listOfCourseIDsAsFaculty = ValueList(qryGetFacultyRole.cc  
let listOfCourseIDsFromEnrollment = ValueList(qryEnrollment.c  
loop list="#listOfCourseIDsAsFaculty#" index="idxThisCourseID"  
<cfif NOT ListFind(listOfCourseIDsFromEnrollment, idxThisCou  
    <cfset listOfCoursesToGet = ListAppend(listOfCoursesToGe  
    </cfif>  
'loop>  
if ListLen(listOfCoursesToGet) GT 0>  
    <cfloop list="#listOfCoursesToGet#" index="idxThisCourseID">  
        <cfquery name="qryGetSingleEnrollment" datasource="#vari  
            SELECT TOP 1 enrollmentID, stuID, userLevel, firstNa  
            username, startDate, endDate, courseID, courseTi  
            startAccessDate, extendAccessDate as endAccessDa  
            FROM vEnrollment2003  
            WHERE stuID = <cfqueryparam cfsqltype="cf_sql_ir  
erID#">  
                AND courseID = <cfqueryparam cfsqltype="cf_s  
eID#">  
                ORDER BY startDate DESC
```

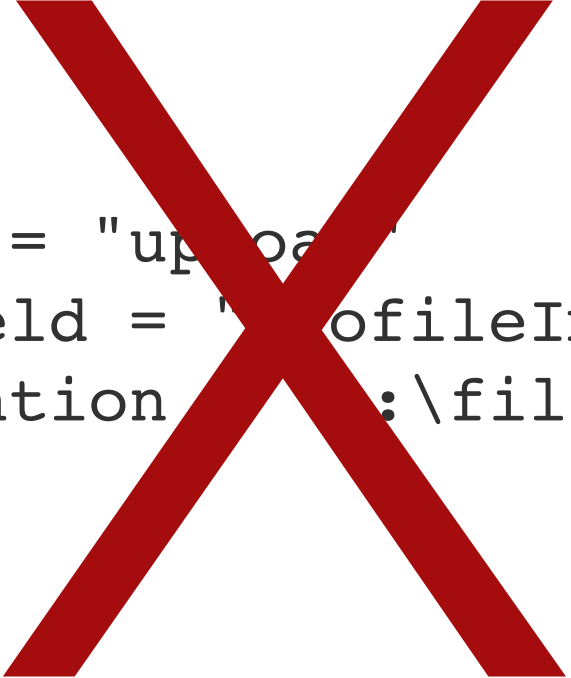


x



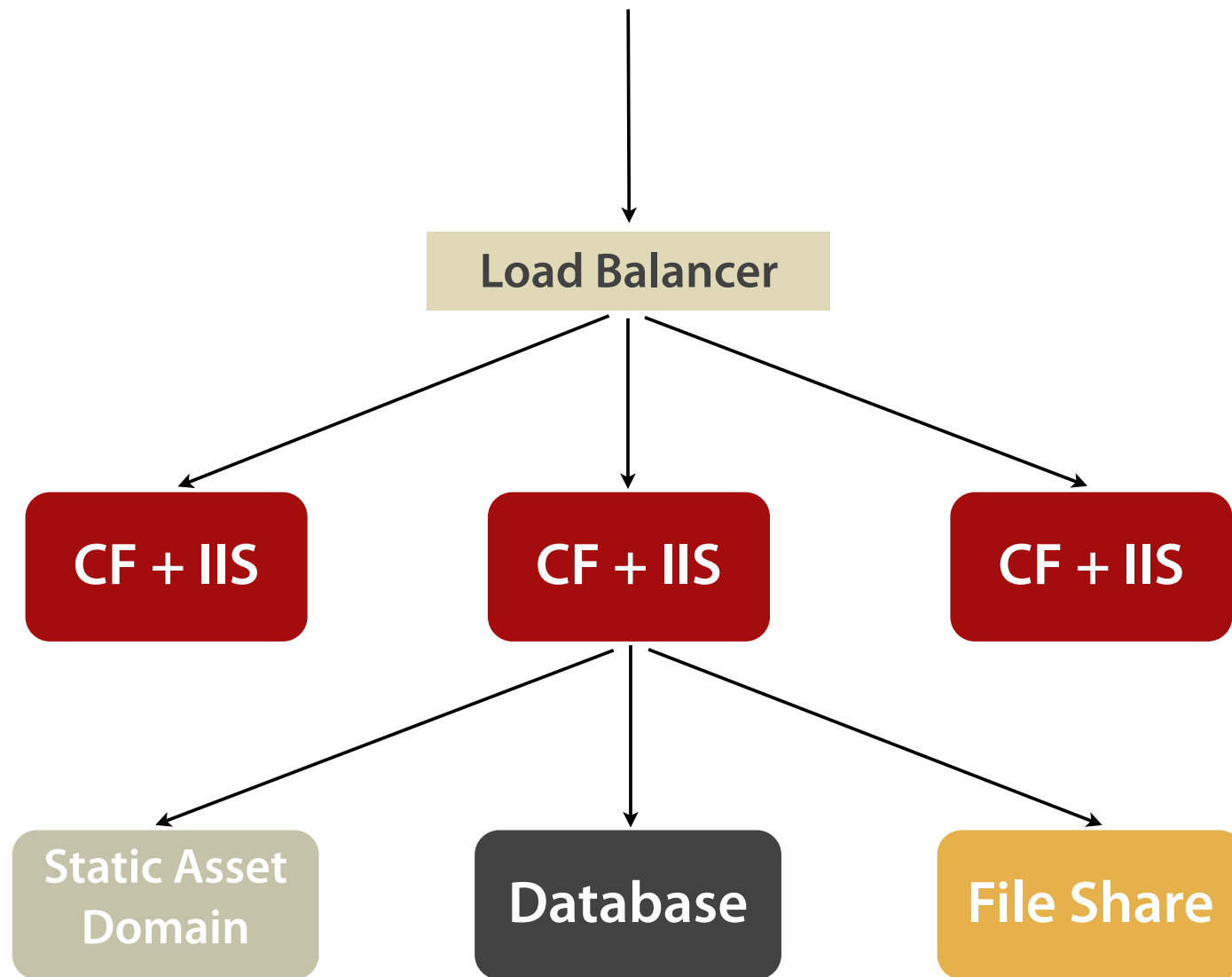
!=





```
<cffile action = "upload"
        fileField = "profileImage"
        destination = ":\\files\\upload\\" />
```

Use network shares
or Amazon S3.



```
<cfset pathToFiles = expandPath("
    /fileShareDirectory/uploads" />
```

```
<cffile action = "upload"
    fileField = "profileImage"
    destination = "#pathToFiles#" />
```

Two Tips

1. Set up mapped directory to the file share at the OS level.
2. Make sure CF has permission to access the share.

What server am I on?

Add to App Startup Code

```
<cfset var inetAddressObj = createObject("java",  
"java.net.InetAddress") />
```

```
<cfset globalVars.machineName =  
inetAddressObj.getLocalHost().getCanonicalHostName() />
```

Environment Config

'Cause you're really, really, really
going to need a testing environment.

Environment Config

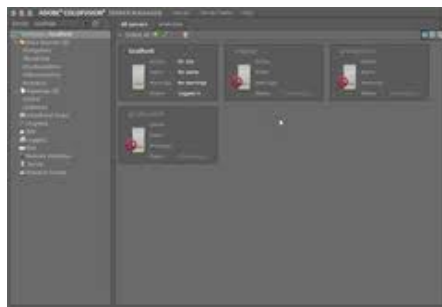
- Development, testing, production?
- Compare server name

Handling Sessions

- Sticky sessions
- Database persistence
- Out-of-process or distributed cache

Basic Monitoring

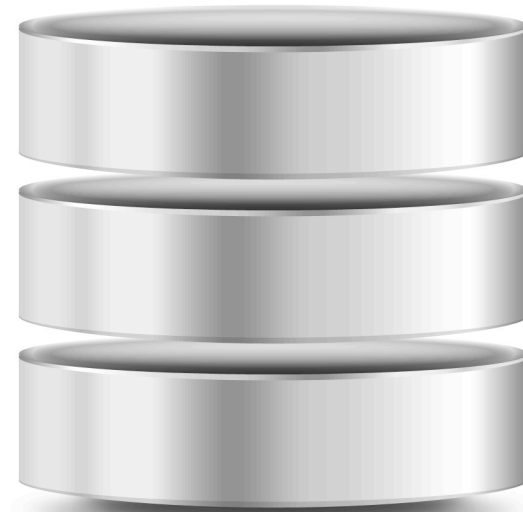




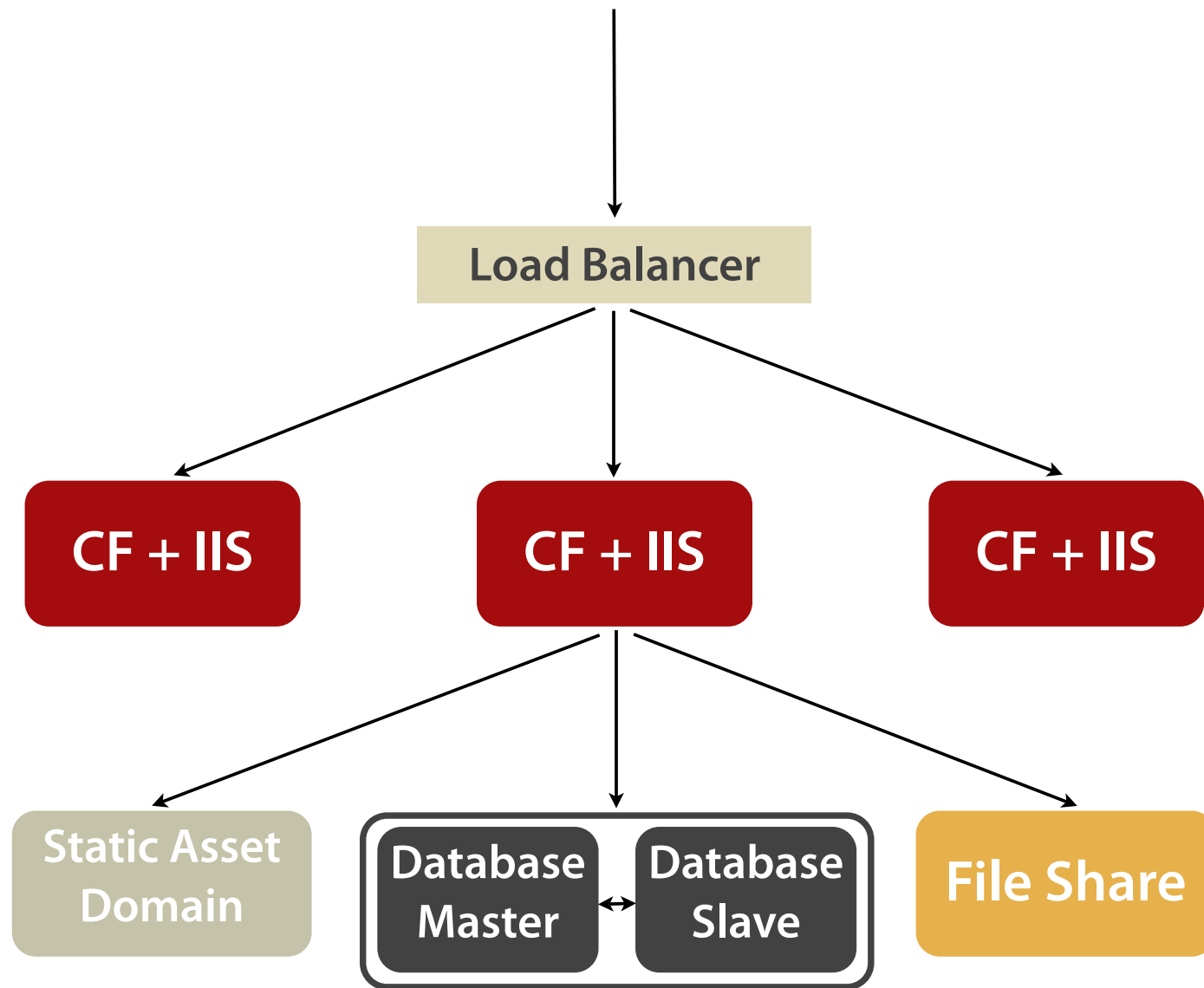
(Very) Basic Monitoring

1. Loop through list of servers to check
2. Make `<cfhttp>` call to a “heartbeat” event on each
3. Result = error, send an email/SMS alert

Changes to Database Setup



Replicate

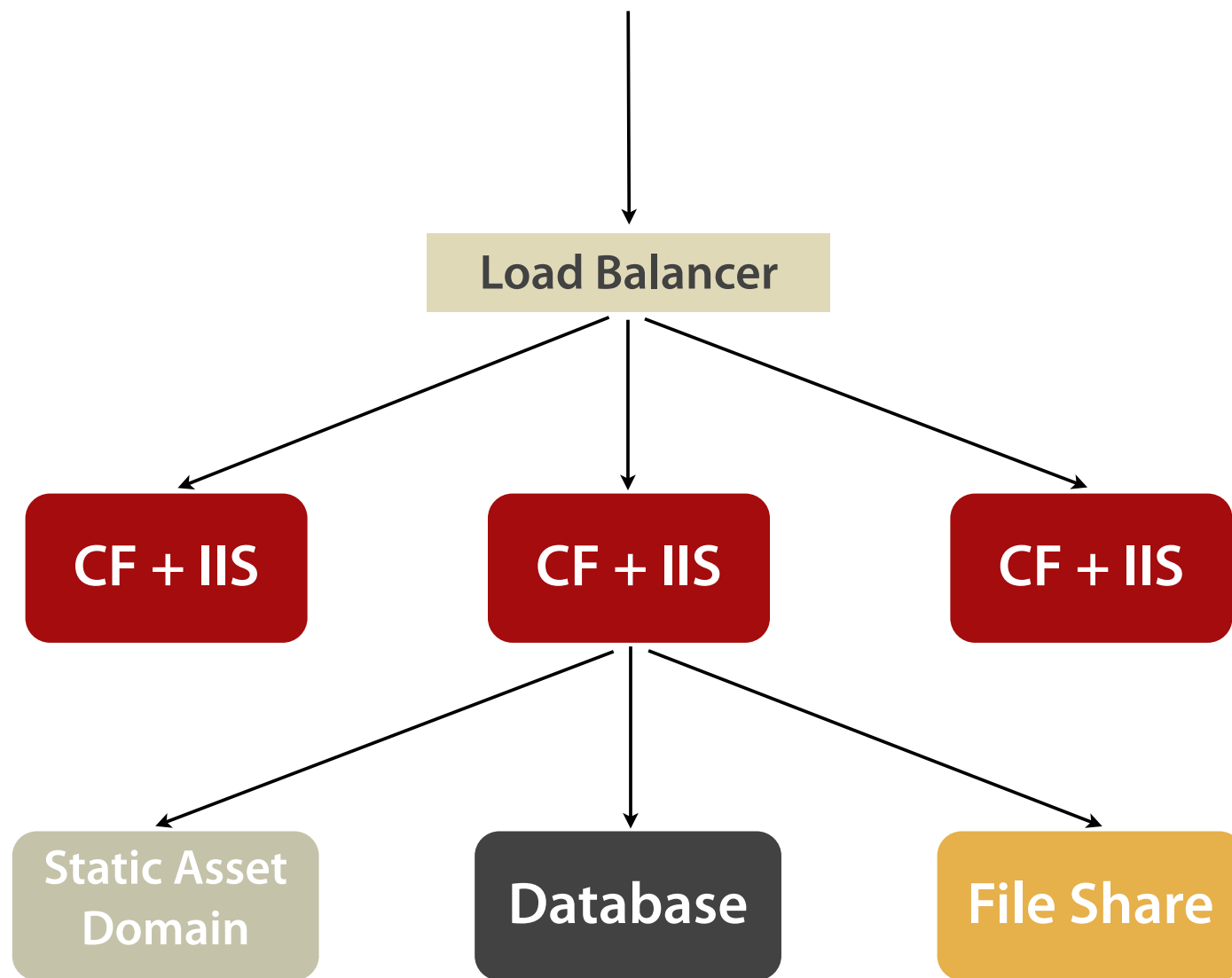


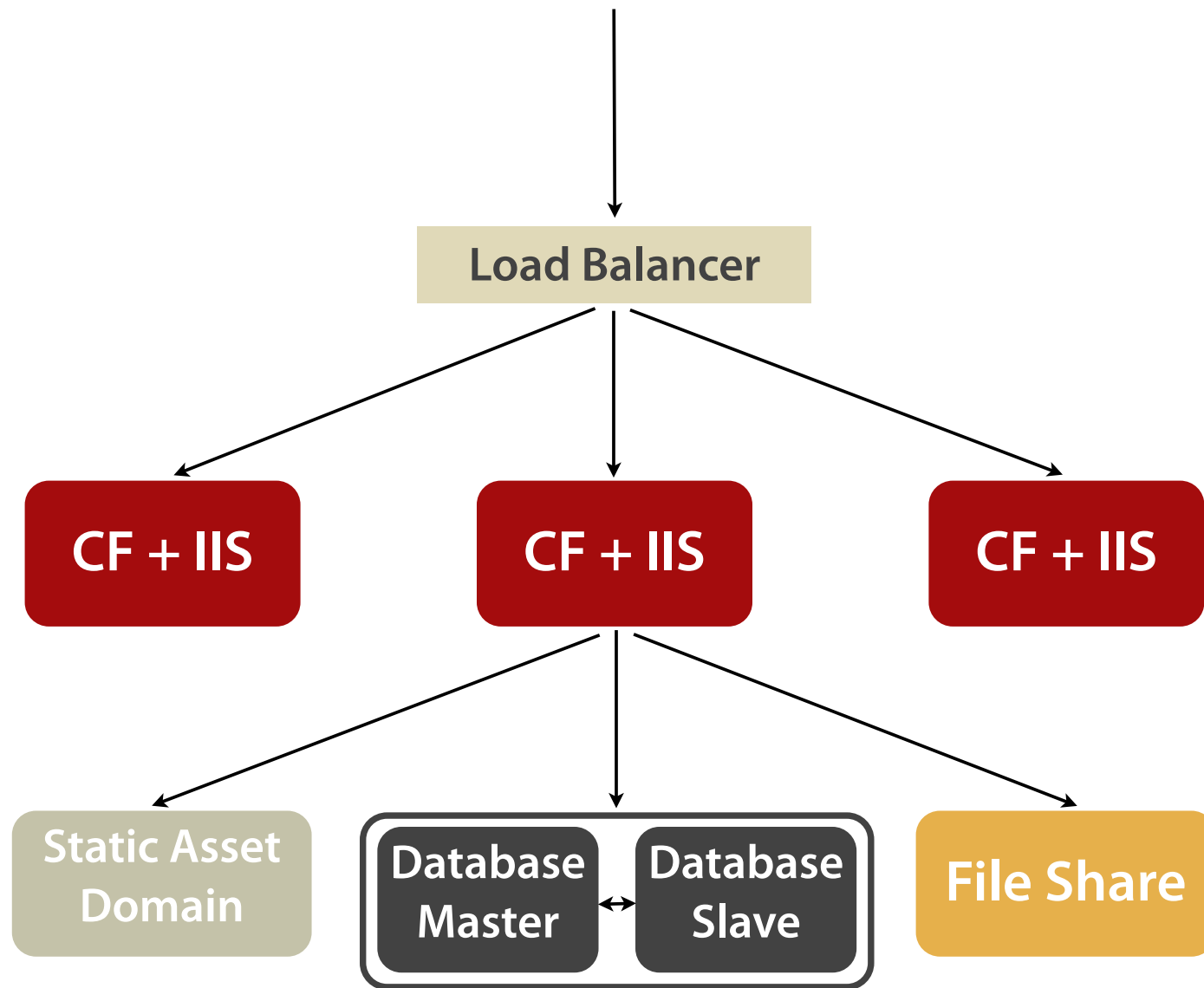
Distributed Databases

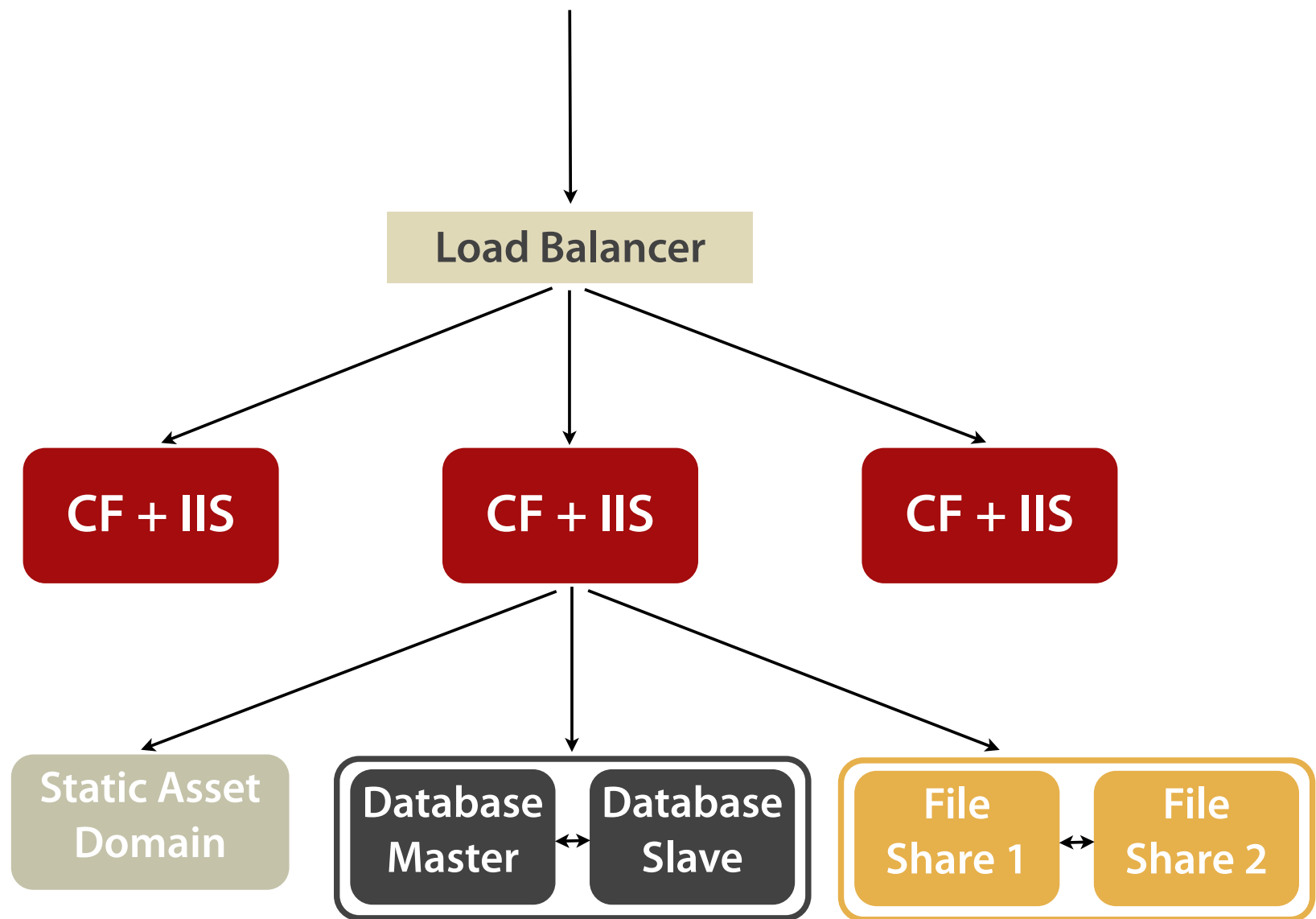
eep

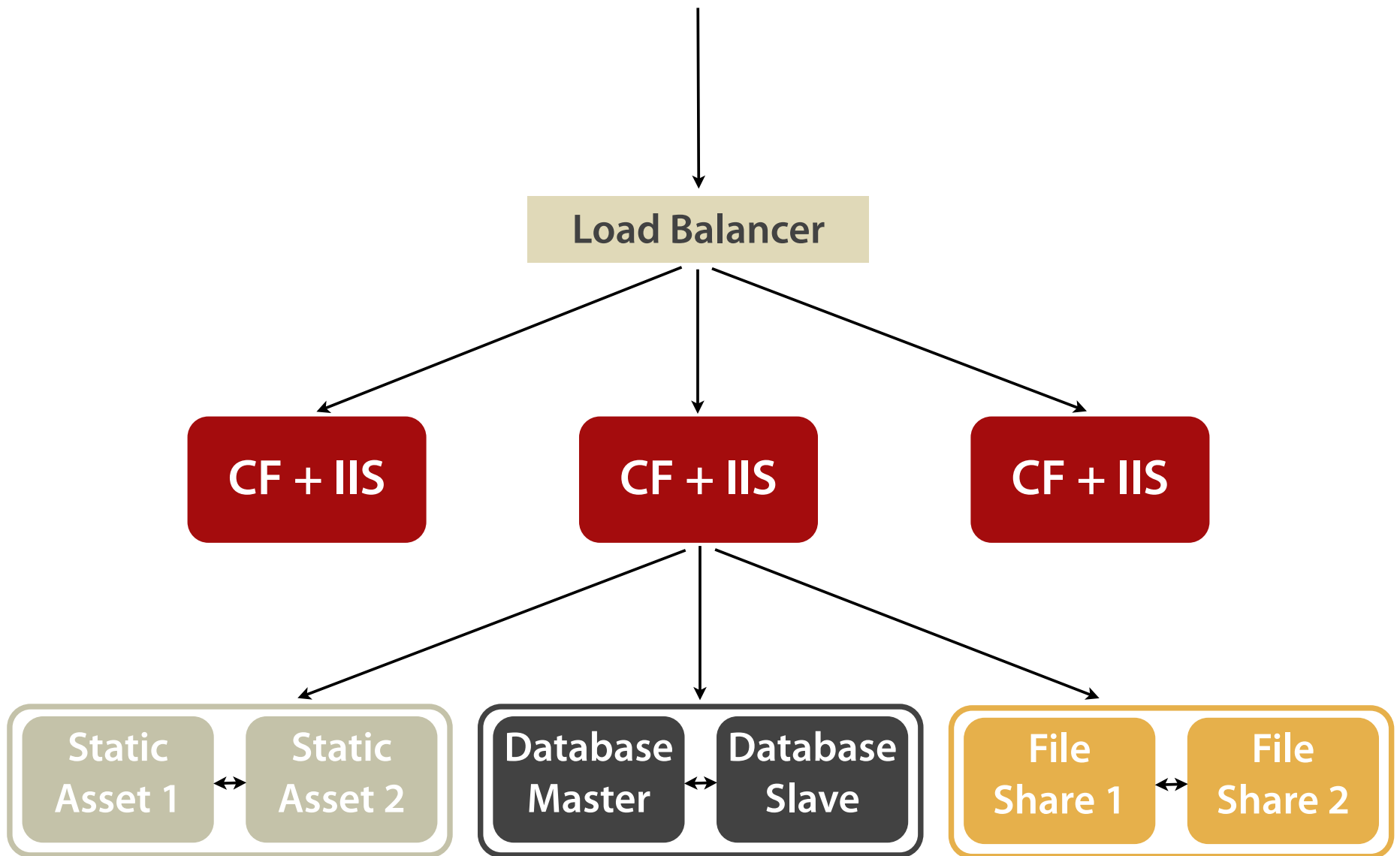
Changes to Server Config

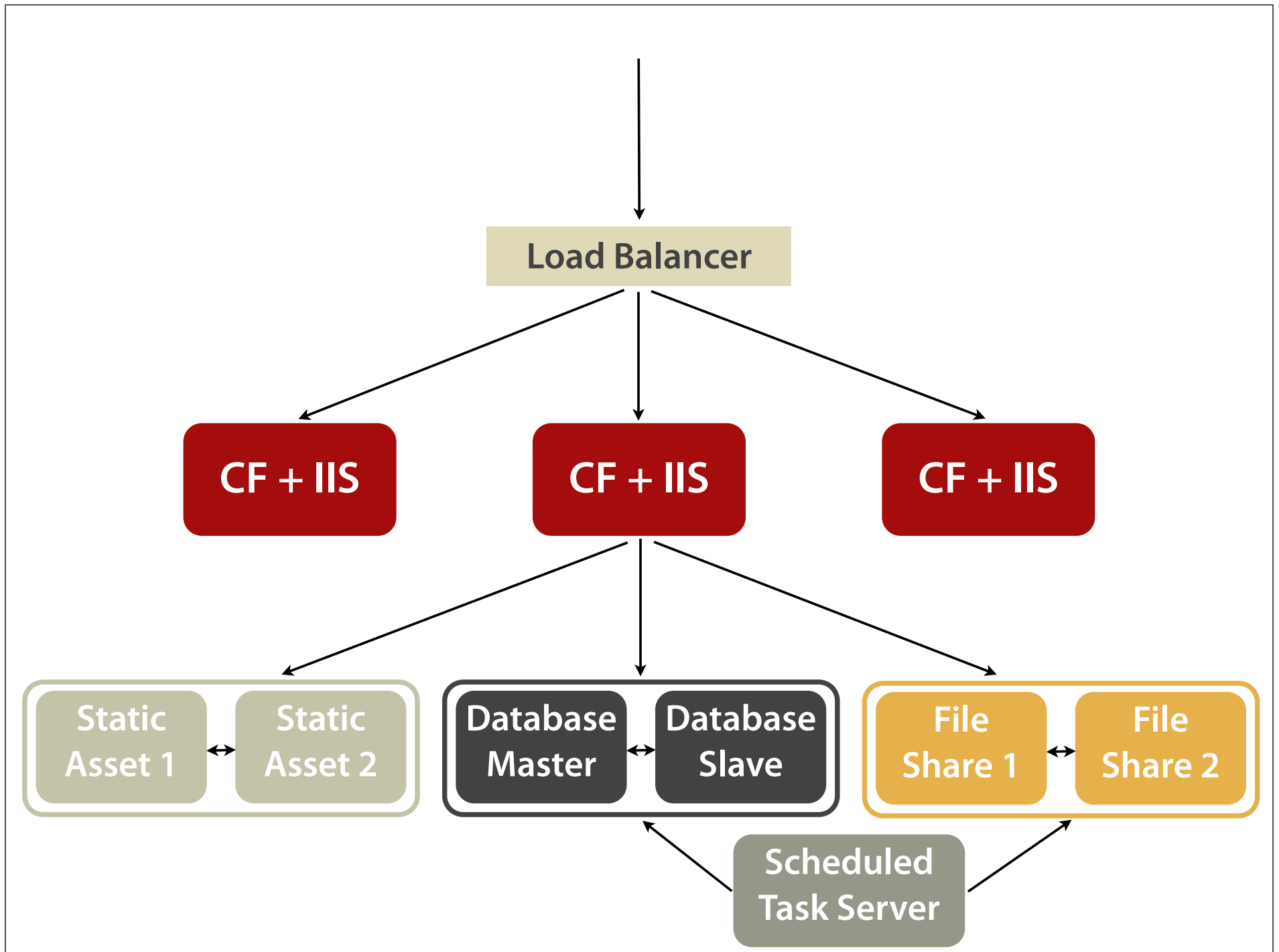




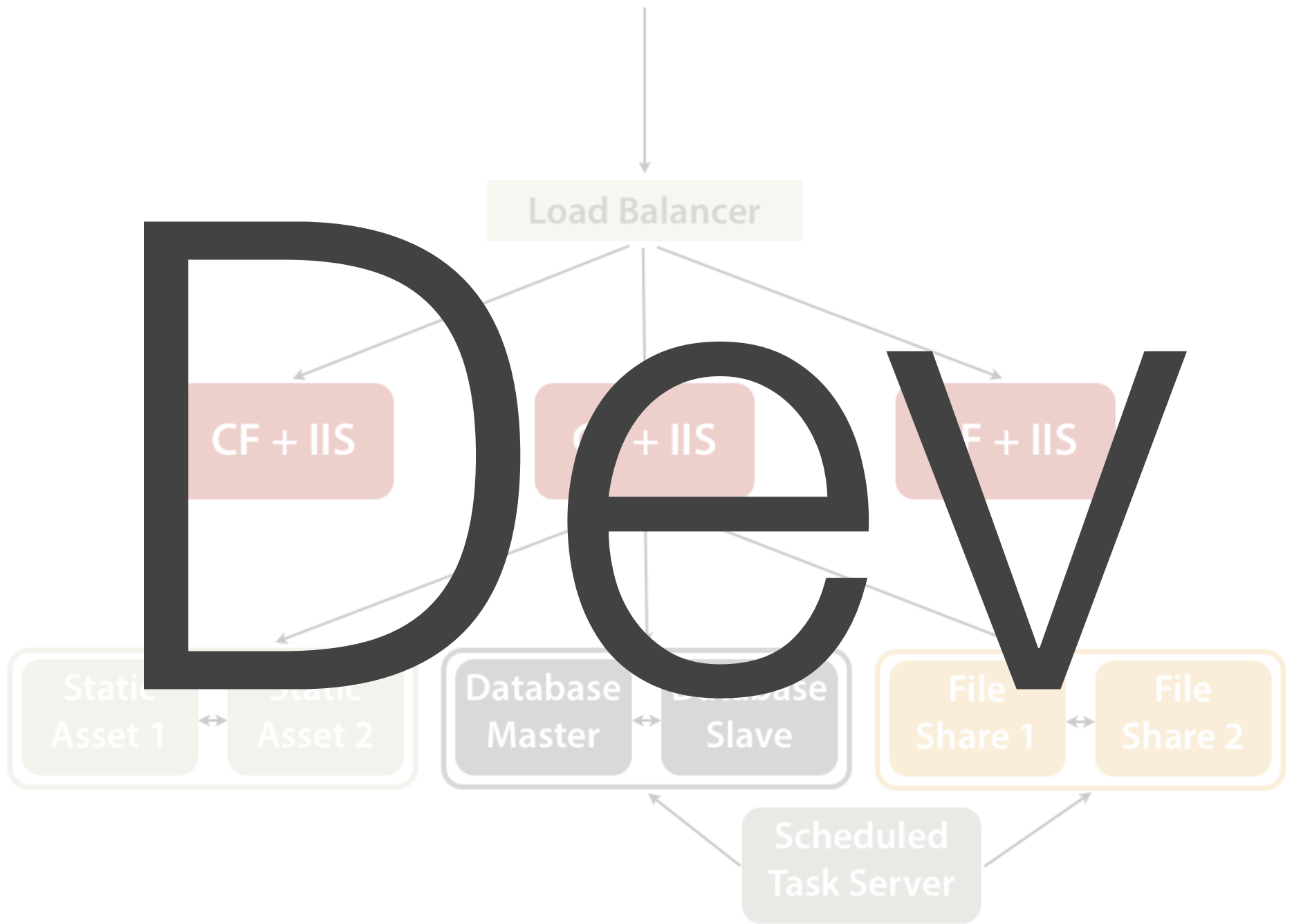




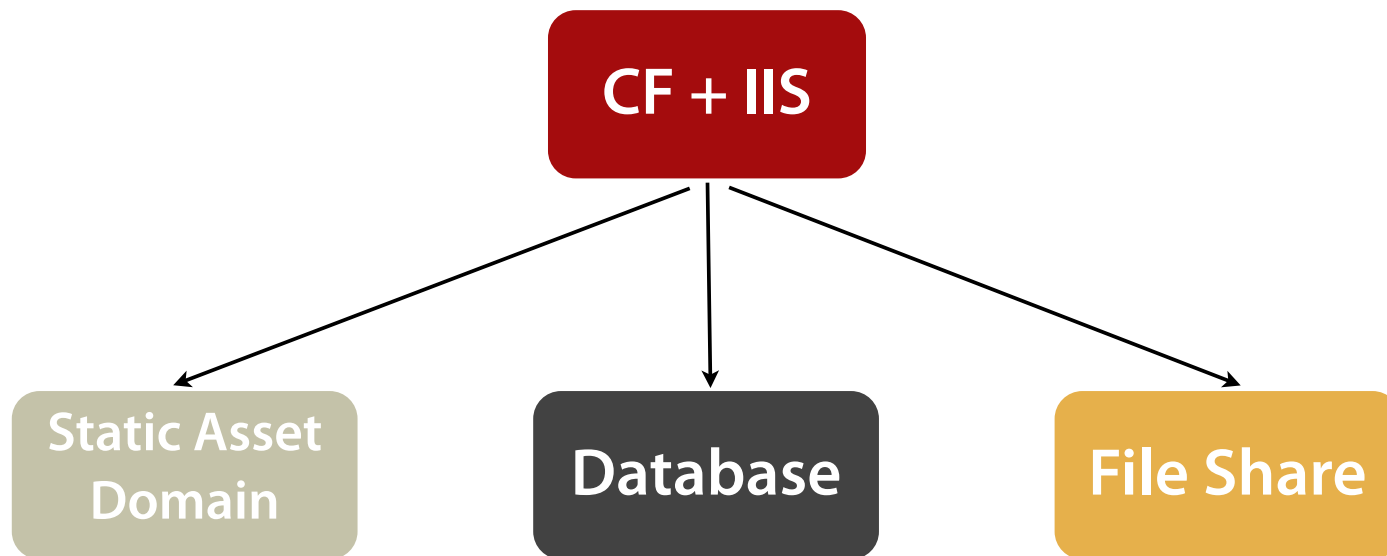




Dev



Dev



What's your
patch
strategy?

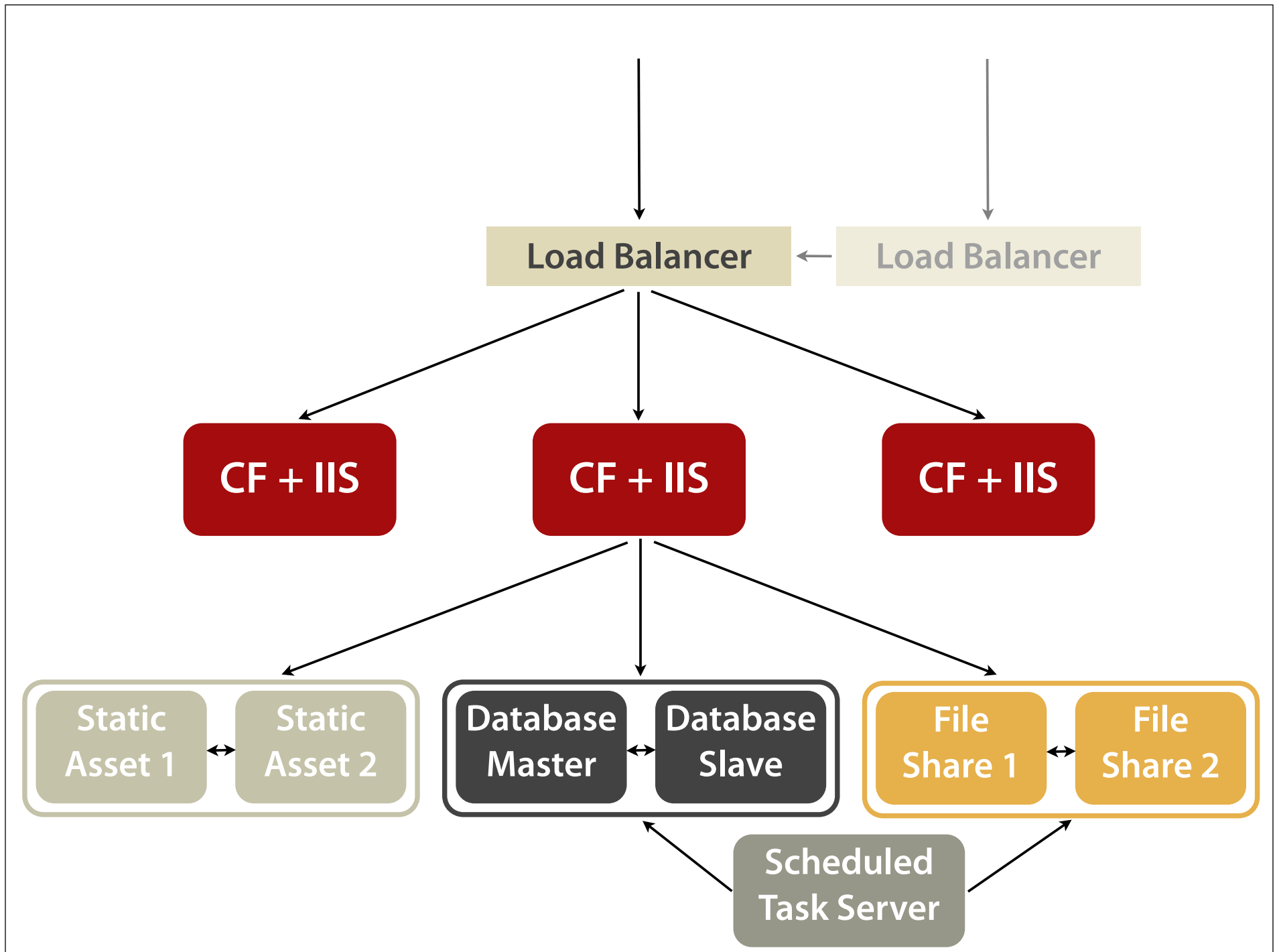


How do I get code there?

FTP Ant Jenkins Chef Puppet

Changes to the Network





And what if the metal blows up?

Node 1

Load Balancer

CF + IIS

CF + IIS

Database
Master

File
Share 2

Node 2

Load Balancer

CF + IIS

Database
Slave

File
Share 1

Static Asset
Domain

And what if the data center
blows up?

Considering AWS*



* Applies to other cloud service providers too.

AWS* = unlimited
virtualized servers in data
centers around the globe.

* And friends.

EC2

AMI on EC2



EBS or S3

↑
Code

↑
Assets

Databases:

Your own

EC2 + EBS

Amazon RDS

MySQL

Oracle

MS SQL Server

NoSQL DB

MongoDB

Cassandra

CouchDB

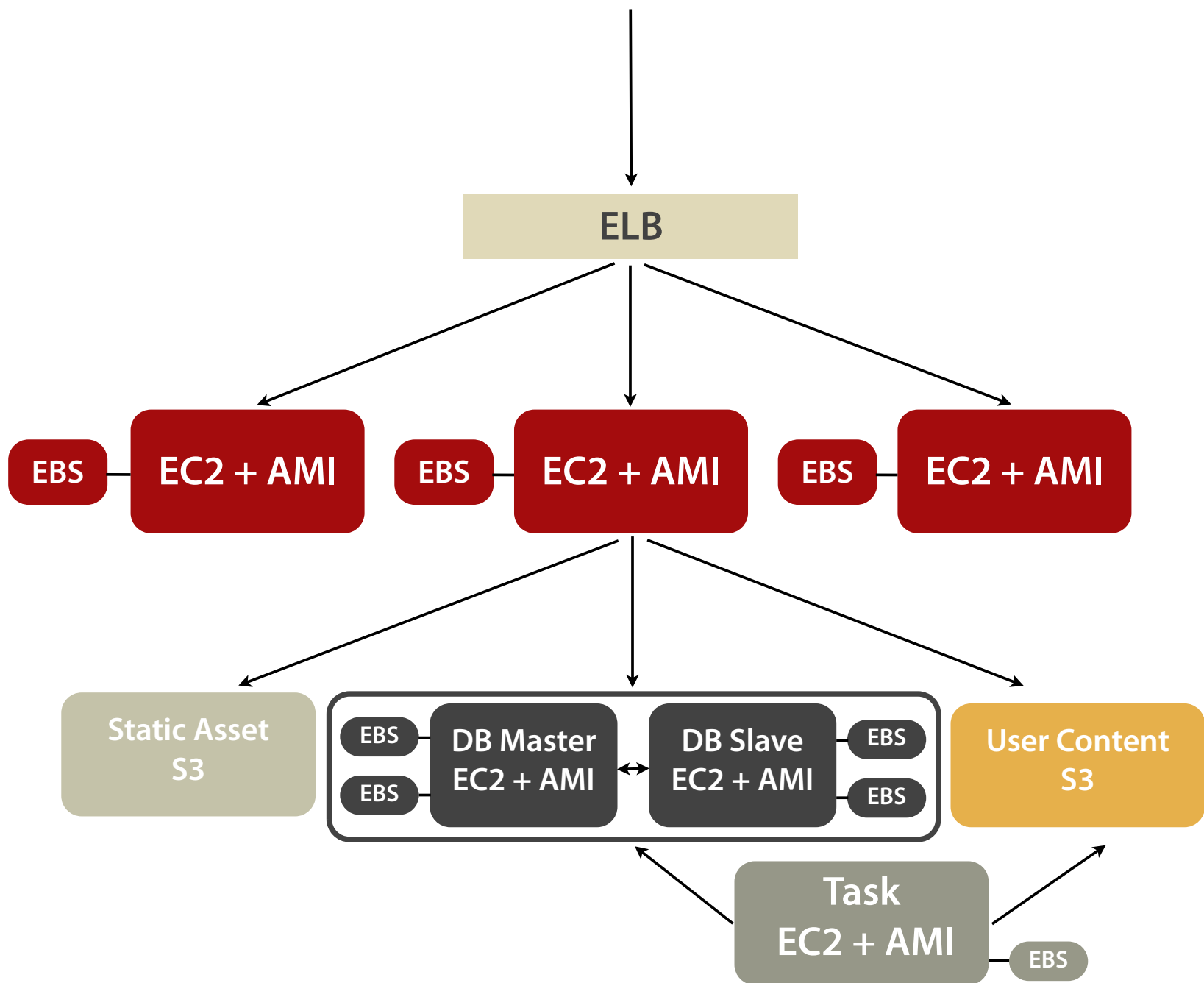
Riak

Redis

AZ Dynamo DB

AZ Simple DB

ELB



IOPS

Cache, cache, cache.

Everything fails.

You are responsible for
redundancy.

Autoscale.

CloudWatch

Elastic Beanstalk

Or another PaaS provider.

Use multiple availability zones.

Region = data center around the globe
Availability zone = segment of a data center

Uh, so is it worth it?

Meta-Network Availability







How much availability
can you afford?

Thank you!

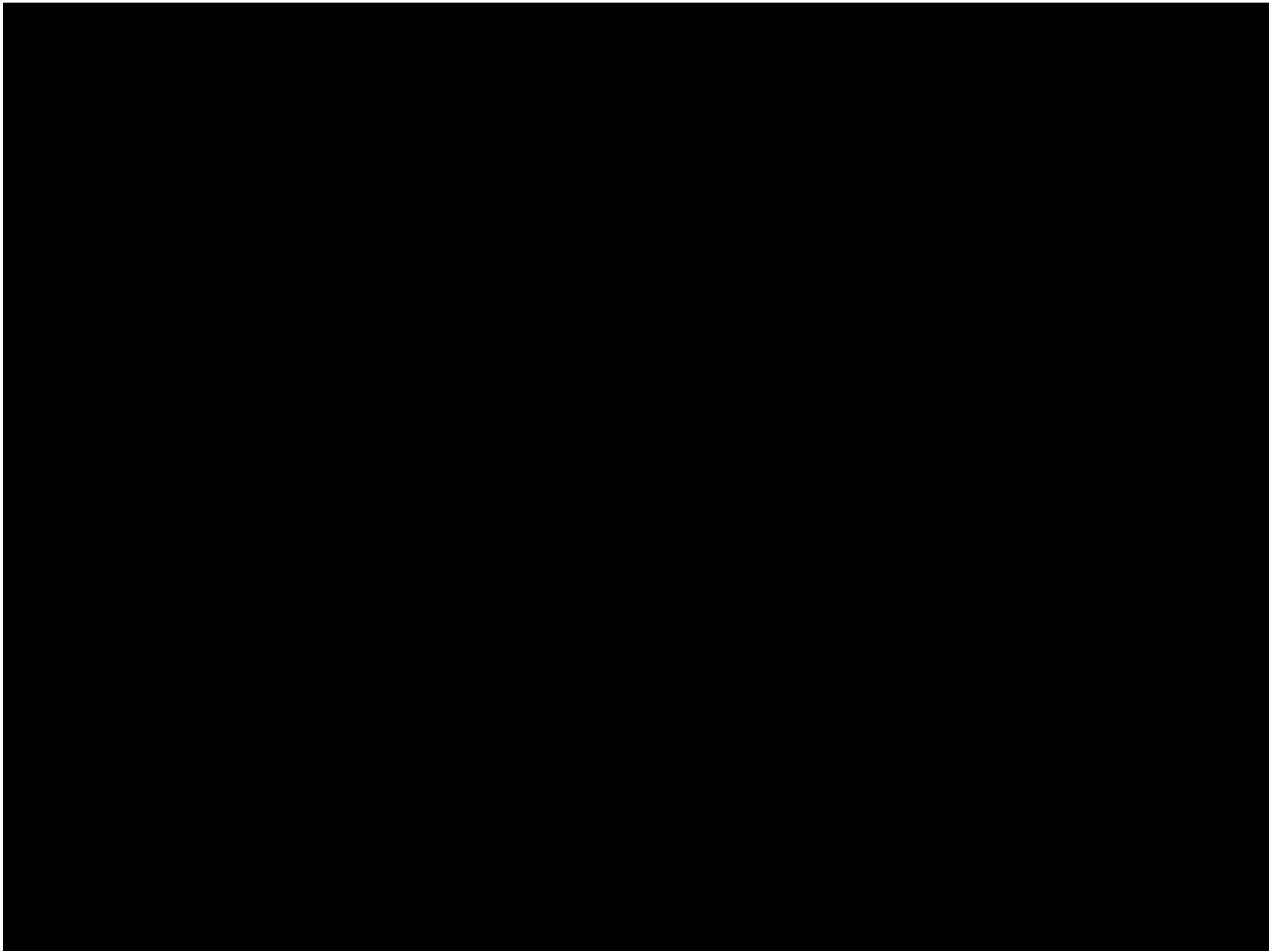
Brian Klaas

Johns Hopkins Bloomberg School of Public Health

bklaas@jhsph.edu

[@brian_klaas](#)

www.iterateme.com



Resources

- Amazon AWS
aws.amazon.com
- Amazon Elastic Compute Cloud
aws.amazon.com/ec2/
- Amazon Machine Images
aws.amazon.com/amis/
- Amazon Elastic Block Storage
aws.amazon.com/ebs/
- Amazon Simple Storage Service (S3)
aws.amazon.com/s3/

Resources

- Amazon Relational Database Service
aws.amazon.com/rds/
- Amazon Elastic Load Balancing
aws.amazon.com/elasticloadbalancing/
- Amazon Elastic Beanstalk (Autoscaling)
aws.amazon.com/elasticbeanstalk/
- Amazon ElastiCache
aws.amazon.com/elasticache/
- Amazon DynamoDB (NoSQL Service)
aws.amazon.com/dynamodb/

Resources

- Amazon Route 53 (DNS Service)
aws.amazon.com/route53/
- Amazon Autoscale
aws.amazon.com/autoscale/
- The Official Word on CF10 Licensing Changes
blogs.coldfusion.com/post.cfm/coldfusion-10-eula
- Jelastic (PaaS Vendor that Supports ColdFusion)
jelastic.com
- Netflix's Chaos Monkey
github.com/Netflix/SimianArmy

Resources

- Apache ANT
ant.apache.org
- Puppet
puppetlabs.com
- Chef
www.opscode.com/chef/
- FusionReactor
www.fusion-reactor.com
- Database Sharding
www.codefutures.com/database-sharding/