PHB: We need a new service to process messages from our gizmos

You: (Oh man oh man, I think I can do this with serverless, it’s the new hotness!)

You: Sure, we can build that in half a day.
We Can Do It!
Hi. I'm Alexa. I'm so easy your 5 year old son could figure out how to program me.
The Promise:

AWS Lambda invokes your code only when needed and automatically scales to support the rate of incoming requests without requiring you to configure anything. There is no limit to the number of requests your code can handle.
The Reality:

AWS Lambda invokes your code only when needed and automatically scales to support the rate of incoming requests without requiring you to configure anything. There is no limit to the number of requests your code can handle.
Serverless Development Lifecycle Gaps

- Access And Permission Management
- Collaboration Mechanisms
- Testing
- Monitoring And Instrumentation
Access And Permission Management
Access And Permission Management

Scenario: A serverless function that

1. Is triggered by an uploaded image to S3 Bucket “uploads”
2. Resizes the image
3. Saves the image to S3 Bucket “resized”
4. Updates a record in DynamoDB table “resize_records”
Access And Permission Management
Access And Permission Management

Shouldn’t this just work?

Nope.

(And that’s a good thing)
serverless permissions
Access And Permission Management

Access To Do Anything In Every S3 Bucket In AWS Account!

```yaml
service: upload-to-s3-and-postprocess
frameworkVersion: ">=1.1.0"

custom:
  bucket: <your-bucket-name>

provider:
  name: aws
  runtime: nodejs4.3

function:
  name:
    - Effect: Allow
      Action:
        - s3:*
      Resource: "*"

functions:
  postprocess:
    handler: handler.postprocess
    events:
      - s3:
          bucket: $self:custom.bucket
          event: s3:ObjectCreated:*
          rules:
            - suffix: .png
```

HTTPS://GITHUB.COM/SERVERLESS/EXAMPLES/BLOB/MASTER/AWS-NODE-UPLOAD-TO-S3-AND-POSTPROCESS/SERVERLESS.YML
Access And Permission Management

Need To Scope Access To Specific Actions

Need To Scope Access To Specific Resources
Access And Permission Management

- **Effect**: Allow
- **Action**:
  - s3:GetObject
  - s3:PutObject
- **Resource**:
  - arn:aws:s3:::uploads
  - arn:aws:s3:::resized
Access And Permission Management

I Have To Do This For Every Function And Resource? How?
Access And Permission Management

Option A: Manual Generation And Provision

1. Developer Hand-Codes IAM Policies
2. Principal Architect Reviews Policies
3. DevOps Deploys Policies
4. You Can Finally Use Your Policy
DO YOU WANT AN ENTIRE DEPARTMENT TO QUIT?

BECAUSE THIS IS HOW YOU MAKE AN ENTIRE DEPARTMENT QUIT
Access And Permission Management

Option B: Let Everyone Do Whatever They Want
Access And Permission Management

Option C: Use A Framework That Automatically Generates Permissions
Access And Permission Management

Automatically Generate Permissions At Deployment Time
Access And Permission Management

Framework-based permission management enables:

- Faster development
- Less errors
- Compliance benefits for the organization
Collaboration Mechanisms
We’re Done Here, Right?
Collaboration Mechanisms

Serverless is cheap enough for every developer to have their own application instances

Serverless local development and testing is hard

I want all my developers to be able to provision into my team’s shared AWS account

But resources require unique names
Collaboration Mechanisms

Solution: Namespace resource names
Collaboration Mechanisms

Option A: Namespace Resources Manually

service: new-service
provider: aws
functions:
  hello:
    name: ${opt:stage}-hello
    handler: handler.hello
  world:
    name: ${opt:stage}-world
    handler: handler.world
Collaboration Mechanisms

Option B: Framework Namespaces
Automatically

Function Name: hello
+ Environment Name: dev
= AWS Lambda Name: dev-hello
Collaboration Mechanisms

My Own Environment
Testing
Serverless Does Not Change Testing!

Serverless Changes How You Run Tests
Testing

Unit Tests: Same As Always

System Tests: ???

Integration Tests: ???
System And Integration Tests: Two Schools Of Thought

A: Always Test In The Cloud

B: Fake Services For Local Testing
Testing

Integration Tests In The Cloud

Pros: Faithful representation, possible today

Cons: Slower, requires cloud access
Testing

Integration Tests Locally With Service Fakes

Pros: Faster, does not require cloud access

Cons: Skew in behavior vs cloud, not very well supported today

Upstream projects are trying to make this possible/easier (e.g. AWS SAM Local)
Testing

Integration Tests: Advice

(For Today Only!)

If application is only API endpoints + Functions, do local tests

Otherwise, deploy into cloud and test
So How Do I Make A Test Environment In The Cloud?

We Solved This Already With Namespaced Resources!
With The Right Approach, Serverless Is Just As Testable As Other Architectures
Monitoring And Instrumentation
Monitoring And Instrumentation

How We Do It Today

1. Organization picks a set of monitoring tools
2. Ask everyone to always instrument the same way
3. Pray
4. Draconian measures
Do you want an entire department to quit?

Because this is how you make an entire department quit.
Monitoring And Instrumentation

How We Should Do It

1. Pick a set of monitoring tools

2. Define instrumentation rules centrally

3. Framework auto-instruments every function

4. Cake
Monitoring And Instrumentation

How Can A Framework Auto-instrument?
Monitoring And Instrumentation

```javascript
// index.js
module.exports.handler = event => {
  return event.x + event.y;
};
```
Monitoring And Instrumentation

```javascript
const handler = require('./index').handler;

module.exports.handler = async event => {
  try {
    // Try to run original handler
    return Promise.resolve(handler(event));
  } catch (err) {
    // If an error occurred, report it to Rollbar
    const rollbar = require('rollbar');
    rollbar.init(process.env.ROLLBAR_TOKEN);
    // Report to Rollbar and wait for completion
    await new Promise(resolve => rollbar.handleError(err, () => resolve()));
    // Re-throw original error
    throw err;
  }
};
```
Monitoring And Instrumentation

Now Just Update The Handler:

index.handler => instrumented.handler
Monitoring And Instrumentation

Great Monitoring Solutions For Serverless

(Diatribe in person because this changes quickly over time and I don’t want to be called out for 2 year old slides)
Monitoring And Instrumentation

Metrics
Monitoring And Instrumentation

Logging
Monitoring And Instrumentation

Tracing
Monitoring And Instrumentation

Error Aggregation
Serverless Development Lifecycle Gaps

- Access And Permission Management
- Collaboration Mechanisms
- Testing
- Monitoring And Instrumentation
How Will You Manage The Gaps?
Build All The Things Yourself
Build All The Things Yourself

all the things?
Use A Toolkit That Does It For You
Thank you!

Chase Douglas, CTO @ Stackery.io
chase@stackery.io
@txase