Workflow Integration Alleviates Identity and Access Management in Serverless Computing

Arnav Sankaran, Pubali Datta, Adam Bates

ILLINOIS
• Why Serverless?
  • Autoscaling
  • Less management
  • Pay per use model
  • Rapid prototyping
Serverless Cloud Computing

- Why Serverless?
  - Autoscaling
  - Less management
  - Pay per use model
  - Rapid prototyping
Serverless Cloud Computing

- Why Serverless?
  - Autoscaling
  - Less management
  - Pay per use model
  - Rapid prototyping
Serverless Cloud Computing

- Why Serverless?
  - Autoscaling
  - Less management
  - Pay per use model
  - Rapid prototyping
Serverless Cloud Computing

• Why Serverless?
  • Autoscaling
  • Less management
  • Pay per use model
  • Rapid prototyping

• Serverless Programming Model
  • Small ephemeral functions
  • Application logic expressed as set of workflows
  • Isolated execution

Serverless Retail Application

Retail Website
Serverless Retail Application

Retail Website
Serverless Retail Application

Retail Website
Serverless Retail Application

Retail Website

Purchase Product
Serverless Retail Application

Retail Website

Purchase Product → Authorize Credit card
Serverless Retail Application

Retail Website

Purchase Product

Authorize Credit card

Credit Card Registry
Serverless Retail Application

1. Purchase Product
2. Authorize Credit card
3. Credit Card Registry

Retail Website
Serverless Retail Application

1. Retail Website
2. Purchase Product
3. Authorize Credit card
4. Credit Card Registry
5. Notify Patient
Serverless Retail Application

1. **Retail Website**
   - Users purchase products.

2. **Purchase Product**
   - Product purchase.

3. **Send Purchase Status**
   - Status sent.

4. **Authorize Credit**
   - Credit card authorized.

5. **Credit Card Registry**
   - Credit card details stored.

The process flow involves a seamless interaction from the retail website to the authorization server, ensuring secure transactions and status updates.
Serverless Retail Application

Retail Website

Purchase Product

Authorize Credit card

Credit Card Registry

Send Purchase Status
Serverless Retail Application

Retail Website

Purchase Product

Authorize Credit Card

Send Purchase Status

Credit Card Registry
Serverless Retail Application

Browse Product Catalog

Purchase Product

Authorize Credit card

Credit Card Registry

Send Purchase Status

Product Catalog
Serverless Retail Application

- Browse Product Catalog
- Purchase Product
- Authorize Credit Card
- Credit Card Registry
- Send Purchase Status
Serverless Retail Application

1. Browse Product Catalog
2. Purchase Product
3. Authorize Credit card
4. Credit Card Registry
5. Send Purchase Status
6. Create Product
Serverless Retail Application

- Browse Product Catalog
- Purchase Product
- Send Purchase Status
- Create Product
- Authorize Credit card
- Credit Card Registry
Retail Application Attack Scenario

Browse Product Catalog → Purchase Product → Authorize Credit card → Credit Card Registry

Create Product → Authorize Credit card

Send Purchase Status
Retail Application Attack Scenario

- Browse Product Catalog
- Purchase Product
- Create Product
- Authorize Credit card
- Send Purchase Status

Credit Card Registry
Retail Application Attack Scenario

Browse Product Catalog

Create Product

Authorize Credit card

Send Purchase Status

Purchase Product

Product Catalog

Credit Card Registry
Retail Application Attack Scenario

Browse Product Catalog

Product Catalog

Create Product

Authorize Credit Card

Credit Card Registry

Send Purchase Status

Purchase Product
Retail Application Attack Scenario

1. Browse Product Catalog
2. Create Product
3. Purchase Product
4. Authorize Credit Card
5. Send Purchase Status

Credit Card Registry

SqlDump
Retail Application Attack Scenario

- Browse Product Catalog
- Create Product
- Purchase Product
- Authorize Credit card
- Send Purchase Status

Credit Card Registry

sqldump

Credit card data
Retail Application Attack Scenario

1. Browse Product Catalog
2. Create Product
3. Authorize Credit card
4. Purchase Product
5. Send Purchase Status

Credit Card Registry

Credit card data

gldump
**Retail Application Attack Scenario**

1. **Browse Product Catalog**
   - User views the product catalog, possibly leading to a product selection.

2. **Create Product**
   - A product is created, potentially involving a process that needs to be secured.

3. **Authorize Credit card**
   - The credit card authorization process is initiated, which could be vulnerable to attacks.

4. **Credit Card Registry**
   - The credit card data is stored in a registry, which might be a target for attackers.

5. **Send Purchase Status**
   - The purchase status is sent, possibly to update the system or notify the user.

6. **SQldump**
   - The credit card data is vulnerable to SQL injection attacks, which could lead to unauthorized access.

The diagram illustrates the flow of events from the user's interaction with the application to the potential points of attack. It highlights the importance of securing the credit card authorization process and the maintenance of the credit card registry to prevent unauthorized access.
Retail Application Attack Scenario

Product Catalog

Create Product

Authorize Credit card

Credit Card Registry

Browse Product Catalog

Purchase Product

Send Purchase Status

Credit card data

gldump
Retail Application Attack Scenario

- Browse Product Catalog
- Product Catalog
- Create Product
- Authorize Credit card
- Send Purchase Status
- Purchase Product
- Credit Card Registry
- Credit Card data
Traditional IAM policies do not encode the semantics of the workflows in a serverless application and are inadequate to prevent sophisticated attacks!
Key Idea
Key Idea
Key Idea

**Problem:** Application logic is expressed in form of workflows, but access control is performed on individual functions.
**Problem:** Application logic is expressed in form of workflows, but access control is performed on individual functions.

**Solution:** Workflow sensitive access control for serverless applications - Will.IAM!
Protection State Graph

- **Purchase product**
- **Authorize credit card**
- **Send purchase status**

- **Browse product**
- **Create product**

- **data:creditCards-read**
- **data:productCatalog-read**
- **data:productCatalog-write**
Labeling State Graph

- A graph defining the workflows and required function permissions

- 
  - Public
    - data:creditCards-read
  - Customer
    - data:creditCards-read
  - Admin
    - data:productCatalog-read
  - Inventory Manager
    - data:productCatalog-write
  - Public
Simplified Will.IAM Architecture
Simplified Will.IAM Architecture

Application User

Invoke foo

External request handler

API gateway

Role R

Authentication server

FaaS platform layer
Simplified Will.IAM Architecture

Application User

Invoke foo

API gateway

Role R

Authentication server

Faas platform layer

External request handler

Mandatory Policy Evaluation Routine

Policy Evaluation Service
Simplified Will.IAM Architecture

Application User

Invoke foo

API gateway

Role R

Authentication server

Faas platform layer

Mandatory Policy Evaluation Routine

Policy Evaluation Service

Labeling state database

Permissions P granted to R

Role R

External request handler

FaaS platform layer
Simplified Will.IAM Architecture

Application User

Invoke foo

Protection state database

Ingress foo

Foo workflow permissions

Role R

Labeling state database

Permissions P granted to R

External request handler

API gateway

Role R

Authentication server

Faas platform layer

Policy Evaluation Service

Mandatory Policy Evaluation Routine

Ingress foo

Foo workflow permissions

Role R

Permissions P granted to R

Invoke foo
Simplified Will.IAM Architecture

1. Application User invokes "foo".
2. External request handler.
3. API gateway.
4. Authentication server.
5. Role R.
6. Ingress foo.
7. Foo workflow permissions.
8. Role R.
9. Permissions P granted to R.
12. Labeling state database.
13. Protection state database.

Ingress

Protection state database

Labeling state database

Foo workflow permissions

Permissions P granted to R

Application User

Invoke foo

External request handler

API gateway

Role R

Authentication server

Faas platform layer

Policy Evaluation Service
Simplified Will.IAM Architecture

Policy Evaluation Service

Mandatory Policy Evaluation Routine

External request handler

Ingress foo

FaaS platform layer

API gateway

Role R

Authentication server

FaaS platform layer

Permission evaluation routine

Labeling state database

Protection state database

Reject

Invoke foo

Application User

Foo workflow permissions

Permissions P granted to R

Role R

Ingress foo

Reject
Simplified Will.IAM Architecture

Policy Evaluation Service
- Conditional permission look-up
  - Table T
    - bar C1
    - func2 C2
    - func3 C3
    - func4 C4

Mandatory Policy Evaluation Routine
- Ingress foo
- Foo workflow permissions
- Role R
- Permissions P granted to R

External request handler
- API gateway
- Role R

Authentication server

FaaS platform layer

Application User
- Invoke foo

Ingress foo
- Protection state database
- Labeling state database

Forward proxy
- Accept
Simplified Will.IAM Architecture

Application User

Invoke foo

External request handler

API gateway

Role R

Authentication server

FaaS platform layer

Ingress foo

FaaS workflow permissions

Role R

Permissions P granted to R

Mandatory Policy Evaluation Routine

Policy Evaluation Service

Conditional permission look-up table T

bar C1
func2 C2
func3 C3
func4 C4

Forward proxy

Accept

Container

Request Handler

Function foo

Container

Protection state database

Labeling state database

foo workflow permissions

Ingress foo

Forward proxy

Accept
Simplified Will.IAM Architecture

Policy Evaluation Service

Conditional permission look-up table T

<table>
<thead>
<tr>
<th>Function</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>bar</td>
<td>C1</td>
</tr>
<tr>
<td>func2</td>
<td>C2</td>
</tr>
<tr>
<td>func3</td>
<td>C3</td>
</tr>
<tr>
<td>func4</td>
<td>C4</td>
</tr>
</tbody>
</table>

Mandatory Policy Evaluation Routine

Ingress foo

Protection state database

Foo workflow permissions

Role R

Labeling state database

Permissions P granted to R

External request handler

API gateway

Role R

Authentication server

Faas platform layer

Forward proxy

Invoke bar

Conditional permission look-up table T

Invoke foo

Application User

Request Handler

Function foo

Container

Conditional permission look-up

Table T

<table>
<thead>
<tr>
<th>Function</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>bar</td>
<td>C1</td>
</tr>
<tr>
<td>func2</td>
<td>C2</td>
</tr>
<tr>
<td>func3</td>
<td>C3</td>
</tr>
<tr>
<td>func4</td>
<td>C4</td>
</tr>
</tbody>
</table>

Invoke bar

Conditional permission look-up

Table T

<table>
<thead>
<tr>
<th>Function</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>bar</td>
<td>C1</td>
</tr>
<tr>
<td>func2</td>
<td>C2</td>
</tr>
<tr>
<td>func3</td>
<td>C3</td>
</tr>
<tr>
<td>func4</td>
<td>C4</td>
</tr>
</tbody>
</table>

Invoke foo

Application User

Request Handler

Function foo

Container
Simplified Will.IAM Architecture

Policy Evaluation Service
Conditional permission look-up table T
bar C1
func2 C2
func3 C3
func4 C4

Mandatory Policy Evaluation Routine

Conditional Policy Evaluation Routine
Forward proxy
Invoke bar

Ingress foo
Foo workflow permissions
Role R

Permissions P granted to R

Labeling state database
Protection state database

External request handler
API gateway
Role R

Authentication server

Application User
Invoke foo

Faas platform layer

Request Handler
Function foo

Conditional Policy

Conditional Policy

Protection state database
Simplified Will.IAM Architecture

- Application User
- Invoke foo
- Faas platform layer
- External request handler
- API gateway
- Role R
- Authentication server
- Ingress foo
- Foo workflow permissions
- Role R
- Permissions P granted to R
- Mandatory Policy Evaluation Routine
- Policy Evaluation Service
- Conditional policy look-up table T
- Conditional Policy Evaluation Routine
- Forward proxy
- Conditional permission look-up
- Accept
- Invoke bar
- Container
- Request Handler
- Function foo
- Ingress foo
- Foo workflow permissions
- Role R
- Permissions P granted to R
- Mandatory Policy Evaluation Routine
- Policy Evaluation Service
- Conditional policy look-up table T
- Conditional Policy Evaluation Routine
- Forward proxy
- Conditional permission look-up
- Accept
- Invoke bar
- Container
- Request Handler
- Function foo
- Ingress foo
- Foo workflow permissions
- Role R
- Permissions P granted to R
- Mandatory Policy Evaluation Routine
- Policy Evaluation Service
- Conditional policy look-up table T
- Conditional Policy Evaluation Routine
- Forward proxy
- Conditional permission look-up
- Accept
- Invoke bar
- Container
- Request Handler
- Function foo
- Ingress foo
- Foo workflow permissions
- Role R
- Permissions P granted to R
- Mandatory Policy Evaluation Routine
- Policy Evaluation Service
- Conditional policy look-up table T
- Conditional Policy Evaluation Routine
- Forward proxy
- Conditional permission look-up
- Accept
- Invoke bar
- Container
- Request Handler
- Function foo
Simplified Will.IAM Architecture

**Policy Evaluation Service**
- Conditional permission look-up table T:
  - bar C1
  - func2 C2
  - func3 C3
  - func4 C4

**Mandatory Policy Evaluation Routine**
- Permissions P granted to R
- Role R

**Conditional Policy Evaluation Routine**
- Conditional Policy Evaluation Routine
  - Accept
  - Reject

**Application User**
- Invoke foo

**External request handler**
- API gateway
- Role R

**Authentication server**

**FaaS platform layer**

**Ingress foo**
- Foo workflow permissions
- Role R

**Labeling state database**

**Protection state database**

**Ingress foo**
- Foo workflow permissions
- Role R

**Conditional Policy**
- Conditional Policy Evaluation Routine
  - Conditional Policy Evaluation Routine
  - Accept
  - Reject

**Forward proxy**

**Ingress foo**
- Foo workflow permissions
- Role R

**Labeling state database**

**Protection state database**

**Invoke bar**

**Conditional Policy**
- Conditional Policy Evaluation Routine
  - Conditional Policy Evaluation Routine
  - Accept
  - Reject

**Forward proxy**

**Invoke foo**

**Application User**

**External request handler**
- API gateway
- Role R

**Authentication server**

**FaaS platform layer**

**Invoke bar**

**Container**
- Request Handler
- Function foo

**Invoke bar**

**Conditional Policy**
- Conditional Policy Evaluation Routine
  - Conditional Policy Evaluation Routine
  - Accept
  - Reject
Simplified Will.IAM Architecture

Proactive denial of requests with insufficient permissions for downstream functions improve performance!
Will.IAM Implementation

• Will.IAM is implemented on top of OpenFaaS and deployed on Kubernetes
• 400 lines of Go code

• Serverless function testset
  • Hello-retail: 12 functions
Evaluation: End-to-end workflow latency

Will.IAM shows average overhead ~0.51%
Evaluation: End-to-end workflow latency

Will.IAM reduces concurrent request latency with increasing percentage of requests with insufficient permissions.
An access control mechanism for preemptive rejection of unauthorized requests

• Low buildtime and runtime overhead

• Will.IAM outperforms Vanilla by 22% when load-testing at bad request proportion of 30%

• Code available at https://bitbucket.org/sts-lab/will.iam/src/master/

Thanks & Questions

pdatta2@illinois.edu