Automatic Policy Generation for Inter-Service Access Control of Microservices

Xing Li\textsuperscript{1,2}, Yan Chen\textsuperscript{2}, Zhiqiang Lin\textsuperscript{3}, Xiao Wang\textsuperscript{2}, and Jim Hao Chen\textsuperscript{2}

\textsuperscript{1}Zhejiang University, \textsuperscript{2}Northwestern University, \textsuperscript{3}The Ohio State University
A Cloud Application
A Cloud Application

User -> HTTP -> Frontend Machine

Frontend Machine -> Backend Machines

Backend Machines -> Database Machines

Database Machines -> Logging Machines
A Cloud Application
A Cloud Application

User

HTTP

Proxy [mTLS]

Frontend

Proxy [mTLS]

Backend

Proxy [mTLS]

Database

Logging

Proxy [mTLS]
apiVersion: security.istio.io/v1beta1
category: AuthorizationPolicy
metadata:
  name: backend-v1-to-database
  namespace: default
spec:
  selector:
    matchLabels:
      app: database
  rules:
  - from:
    source:
      principals: ["cluster.local/ns/default/sa/backend"]
    to:
      operation:
        ports: ["9000"]
• Manual Policy Configuration?  
  Time-consuming, error-prone, inflexible  
• Automatic Security Policy Generation Approaches for Distributed Systems?  
  1. Document-based approaches?  
     Low accuracy, poor availability  
  2. History-based approaches?  
     Requiring complete historical data  
  3. Model-based approaches?  
     Poor agility and scalability
What’s new in microservices?

- Microservices are small: a single service has low internal complexity.
- The inter-service invocation manner in the same application is relatively uniform.
- The amounts of involved protocols and libraries are limited.

Extract the normal system behavior with static analysis

**Static Analysis-Based Request Extraction**

- Code of Microservices
- Request Extraction Phase

**Graph-Based Policy Management**

- Inter-Service Invocation Logic
- Inter-Service Access Control Policies
- Policy Management Phase
The deployment of microservice E

AutoArmor

- Static Analysis Engine
- Permission Engine
- Policy Generator
import requests
from flask import request, session
...
reviews = {
    "name" : "http://reviews:9080",
    "endpoint" : "reviews"
}
...
@app.route('/api/v1/products/<product_id>/reviews')
def reviewsRoute(product_id):
    headers = getForwardHeaders(request)
    user = session.get('user', '')
    status, reviews = getProductReviews(product_id, headers)
    ...

def getProductReviews(product_id, headers):
    try:
        url = reviews['name'] + "/" + reviews['endpoint'] + "/" + str(product_id)
        res = requests.get(url, headers=headers, timeout=3.0)
    ...

Step-I:
Identifying the statements that initiate inter-service invocations
import requests
from flask import request, session
...

reviews = {
    "name" : "http://reviews:9080",
    "endpoint" : "reviews"
}
...

@app.route('/api/v1/products/<product_id>/reviews')
def reviewsRoute(product_id):
    headers = getForwardHeaders(request)
    user = session.get('user', '')
    status, reviews = getProductReviews(product_id, headers)
    ...

def getProductReviews(product_id, headers):
    try:
        url = reviews['name'] + "/" + reviews['endpoint'] + "/" + str(product_id)
        res = requests.get(url, headers=headers, timeout=3.0)
    ...

Phase 1: Request Extraction

Library: requests
   Method: get(url, params=None, **kwargs)
   Semantics: HTTP-GET
   Key parameters: url (Semantics: HTTP-URL)

Step-II:
Performing backward taint propagation to get the program slices associated with each invocation
Phase 1: Request Extraction

```python
reviews = {
    "name": "http://reviews:9080",
    "endpoint": "reviews"
}

@app.route('/api/v1/products/<product_id>/reviews')
url = reviews['name'] + '/' + reviews['endpoint'] + '/' + str(product_id)
res = requests.get(url, headers=headers, timeout=3.0)
```

**Program Slice**

```
{
    "type": "HTTP",
    "url": "http://reviews:9080/reviews/*",
    "path": "/reviews/*",
    "method": "GET"
}
```

**Extracted Request**

Step-III:
Extracting the detailed attributes of invocations
Phase 2: Policy Management

Integrate the permissions shared by all versions of the same service
- Eliminate redundant policies
- Eliminate unnecessary policy updates
Evaluation

- **Materials:** 5 popular open-source microservice applications

<table>
<thead>
<tr>
<th>Name</th>
<th># of Services</th>
<th>LoCs</th>
<th>Type</th>
<th>Multiple Languages</th>
<th>★ on GitHub</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bookinfo</td>
<td>6</td>
<td>2,702</td>
<td>Demo</td>
<td>✓</td>
<td>24.7k</td>
</tr>
<tr>
<td>Online Boutique</td>
<td>11</td>
<td>23,219</td>
<td>Demo</td>
<td>✓</td>
<td>8.8k</td>
</tr>
<tr>
<td>Sock Shop</td>
<td>13</td>
<td>20,150</td>
<td>Demo</td>
<td>✓</td>
<td>2.5k</td>
</tr>
<tr>
<td>Pitstop</td>
<td>13</td>
<td>45,028</td>
<td>Demo</td>
<td>✗</td>
<td>630</td>
</tr>
<tr>
<td>Sitewhere</td>
<td>21</td>
<td>53,751</td>
<td>Industrial</td>
<td>✗</td>
<td>717</td>
</tr>
</tbody>
</table>

- **Hardware:** a 3-node Kubernetes cluster (v.1.18.6) with Istio (v1.6.8); Each node is equipped with eight 2.30-GHz Intel(R) Core(TM) CPUs (i5-8259U) and 32 GB of RAM
<table>
<thead>
<tr>
<th>Application</th>
<th>Microservice</th>
<th>Language</th>
<th>LoCs</th>
<th>Identified Requests</th>
<th>Extracted Attributes</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HTTP</td>
<td>gRPC</td>
<td>TCP</td>
</tr>
<tr>
<td>Bookinfo</td>
<td>productpage</td>
<td>Python</td>
<td>2,061</td>
<td>3/3</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>details</td>
<td>Ruby</td>
<td>122</td>
<td>1/1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>reviews</td>
<td>Java</td>
<td>301</td>
<td>1/1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>ratings</td>
<td>JavaScript</td>
<td>218</td>
<td>-</td>
<td>2/2</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>front-end</td>
<td>Go</td>
<td>3,666</td>
<td>-</td>
<td>1/11</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>cartservice</td>
<td>C#</td>
<td>5,941</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>productcatalogservice</td>
<td>C#</td>
<td>2,460</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>currencyservice</td>
<td>JavaScript</td>
<td>359</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>paymentservice</td>
<td>JavaScript</td>
<td>343</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>shippingservice</td>
<td>Go</td>
<td>2,458</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>emailservice</td>
<td>Python</td>
<td>2,146</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>checkoutservice</td>
<td>Go</td>
<td>2,816</td>
<td>-</td>
<td>8/8</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>recommendationservice</td>
<td>Python</td>
<td>2,112</td>
<td>-</td>
<td>1/1</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>adservice</td>
<td>Java</td>
<td>918</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Online Boutique</td>
<td>front-end</td>
<td>JavaScript</td>
<td>9,922</td>
<td>33/33</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>orders</td>
<td>Java</td>
<td>2,187</td>
<td>-</td>
<td>2/2</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>payment</td>
<td>Go</td>
<td>863</td>
<td>-</td>
<td>-</td>
<td>4/8</td>
</tr>
<tr>
<td></td>
<td>user</td>
<td>Go</td>
<td>2,515</td>
<td>-</td>
<td>24/24</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>catalogue</td>
<td>Go</td>
<td>1,439</td>
<td>-</td>
<td>8/8</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>carts</td>
<td>Go</td>
<td>1,840</td>
<td>-</td>
<td>7/7</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>shipping</td>
<td>Java</td>
<td>929</td>
<td>-</td>
<td>3/3</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>queue-master</td>
<td>Java</td>
<td>926</td>
<td>-</td>
<td>3/3</td>
<td>-</td>
</tr>
<tr>
<td>Sock Shop</td>
<td>webapp</td>
<td>C#</td>
<td>40,461</td>
<td>16/16</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>customernmanagementservice</td>
<td>C#</td>
<td>423</td>
<td>-</td>
<td>5/5</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>vehiclenmanagementservice</td>
<td>C#</td>
<td>451</td>
<td>-</td>
<td>5/5</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>workshopsmanagement</td>
<td>C#</td>
<td>1,563</td>
<td>4/4</td>
<td>20/20</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>workshopmanagementservice</td>
<td>C#</td>
<td>685</td>
<td>10/10</td>
<td>-</td>
<td>14/14</td>
</tr>
<tr>
<td></td>
<td>auditlogservice</td>
<td>C#</td>
<td>136</td>
<td>1/1</td>
<td>-</td>
<td>2/2</td>
</tr>
<tr>
<td></td>
<td>notificationservice</td>
<td>C#</td>
<td>511</td>
<td>7/7</td>
<td>-</td>
<td>12/12</td>
</tr>
<tr>
<td></td>
<td>invoicenexception</td>
<td>C#</td>
<td>641</td>
<td>9/9</td>
<td>-</td>
<td>14/14</td>
</tr>
<tr>
<td></td>
<td>timeservice</td>
<td>C#</td>
<td>157</td>
<td>1/1</td>
<td>-</td>
<td>1/1</td>
</tr>
<tr>
<td>Phitstop</td>
<td>web-rest</td>
<td>Java</td>
<td>6,648</td>
<td>-</td>
<td>215/215</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>instance-management</td>
<td>Java</td>
<td>4,069</td>
<td>-</td>
<td>35/35</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>event-sources</td>
<td>Java</td>
<td>6,619</td>
<td>-</td>
<td>1/1</td>
<td>3/3</td>
</tr>
<tr>
<td></td>
<td>inbound-processing</td>
<td>Java</td>
<td>825</td>
<td>-</td>
<td>2/2</td>
<td>4/4</td>
</tr>
<tr>
<td></td>
<td>device-management</td>
<td>Java</td>
<td>6,381</td>
<td>-</td>
<td>-</td>
<td>74/74</td>
</tr>
<tr>
<td></td>
<td>event-management</td>
<td>Java</td>
<td>4,799</td>
<td>-</td>
<td>4/4</td>
<td>60/60</td>
</tr>
<tr>
<td></td>
<td>asset-management</td>
<td>Java</td>
<td>5,993</td>
<td>-</td>
<td>10/10</td>
<td>10/10</td>
</tr>
<tr>
<td></td>
<td>schedule-management</td>
<td>Java</td>
<td>1,964</td>
<td>-</td>
<td>10/16</td>
<td>16/16</td>
</tr>
<tr>
<td>SiteWrench</td>
<td>batch-operations</td>
<td>Java</td>
<td>2,122</td>
<td>-</td>
<td>6/6</td>
<td>16/16</td>
</tr>
<tr>
<td></td>
<td>device-registration</td>
<td>Java</td>
<td>1,075</td>
<td>-</td>
<td>10/4</td>
<td>4/4</td>
</tr>
<tr>
<td></td>
<td>device-state</td>
<td>Java</td>
<td>1,739</td>
<td>-</td>
<td>1/1</td>
<td>7/7</td>
</tr>
<tr>
<td></td>
<td>event-search</td>
<td>Java</td>
<td>769</td>
<td>4/4</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>label-generation</td>
<td>Java</td>
<td>1,379</td>
<td>-</td>
<td>10/10</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>rule-processing</td>
<td>Java</td>
<td>1,091</td>
<td>-</td>
<td>2/2</td>
<td>2/2</td>
</tr>
<tr>
<td></td>
<td>command-delivery</td>
<td>Java</td>
<td>3,417</td>
<td>-</td>
<td>6/6</td>
<td>3/3</td>
</tr>
<tr>
<td></td>
<td>streaming-media</td>
<td>Java</td>
<td>736</td>
<td>-</td>
<td>10/10</td>
<td>10/10</td>
</tr>
<tr>
<td></td>
<td>outbound-connectors</td>
<td>Java</td>
<td>4,125</td>
<td>-</td>
<td>13/13</td>
<td>2/2</td>
</tr>
</tbody>
</table>

| Total              | 48 unique services | 6 languages | 96/96 | 290/290 | 369/369 | 751/755 | 96/96 | 369/369 | - | 

**Q1:** Can AUTOARMOR extract the inter-service invocation logic?

- Request identification rate: 100%
- Request attribute extraction rate: 99.5%
- Average static analysis time: 57 s/svc
Q2:
Can AUTOARMOR efficiently generate, manage, and update access control policies?

- The policy generation time for each evaluation application is less than 2 s.
- It takes less than 12 s to generate 900 policies for a large application with 1,000 services.
Q3:

Can AUTOARMOR improve the application’s performance via the optimized policy set?

- By eliminating redundant policies, it enables microservice applications to achieve better end-to-end performance.
The first automatic policy generation tool for inter-service access control of microservices

- A static analysis-based request extraction mechanism
- A graph-based policy management mechanism
- Effectively bridge the policy generation gap with only a minor overhead
Thanks!

xing.li@zju.edu.cn