Microservices for Web Based Applications and Security

Matthew Knill
Overview

1. Monolithic approaches to web development
2. Microservice approaches to web development
3. Describe what Clean Architecture and Domain Driven Design is
4. Explain Microservice design and Implementation
5. Outline the benefits of using microservices for large projects
6. Describe the security challenges of microservices
7. What does this mean
Monolithic approaches to web development

- Typically has a web server (such as Apache/PHP) and a database (such as MySQL)
- Different components behave as a single component
  - Even if web server and database on separate machines, this is not a microservice approach as they must operate in tandem with each other
Microservice approaches to web development

- Separate application into individually serviced components
- Different from a monolithic approach as each microservice can operate entirely on its own
- Can each have different architecture
- Communicate through an API or some other form of message passing
Clean Architecture

• Project last year “Refactoring To Clean Architecture”

Core Concepts
• All dependencies point inwards
• Database, UI, external agents etc... can be swapped out or altered without affecting business logic
• Relies on the Dependency inversion principle
Domain Driven Design

- **Ubiquitous Language** – project specific language that both technical and non-technical stakeholders can understand
- **Model** – ‘foundation of the design’ which is a combination of both analytical models and design models
- **Context** – statement or word in the model and Ubiquitous Language
- **Bounded Context** – hypothetical boundary that separates contexts in a large project
Microservice Design & Implementation

1. Identify Domain
2. Define Bounded Contexts
3. Identify Microservices
4. Design Microservice Specific Architecture
5. Implement
6. Deploy

流程图说明：
- 从“Identify Domain”开始
- 继续“Define Bounded Contexts”
- 接着“Identify Microservices”
- 然后是“Design Microservice Specific Architecture”
- 再到“Implement”
- 最后是“Deploy”
Microservice Design Example
Securing Microservices

Microservices bring new security challenges

• Perimeter defence no longer considered safe
• Increased difficulty in separating business logic from security in DDD

Current Methods

• Token Based Security
• Certificate Based Security
• Security as a Service

Authentication – authenticate user into system
Authorization – authenticate user to a specific task
Token Based Security in DDD

User Payments MS Authentication MS

verifyToken(token)
return authentication success

requestProtectedResource(token)
return protected resource

Authentication Success Failure

authenticateUser()
return token

checkUser(token)
return user access failure

checkToken(token)
return authentication failure

Authorization

Authentication Success Failure
Issues – Man-in-the-middle attack

Solution – short expiration time for tokens
Certificate Based Security

- Uses public/private key
- Certificate is an electronic document used to verify the integrity of the sender
- Issued by a trusted Certificate Authority

Certificate Authority verifies the identity of Mario Rossi and encrypts with its Private Key

Source 4
Security as a Service

• No overlap of bounded context with other bounded contexts
• Remains secure as the *Security MS* analyses all input and output data from relevant microservice
• Resource intensive
What does this mean?

• Microservice security is difficult!
  – Think about it early on in the project development
  – Sometimes a combination of all three approaches can be used

• Strict principles of Domain Driven Design will likely need to be relaxed
  – May result in a microservice design that is slightly less modular, however, trumped by having an implementation that is more secure

• More research is required that is backed up by stringent experimentation to provide a clearer understanding of microservice security
Any Questions

Source 5
Bibliography

Image Sources