Case Study

Memi Lavi
www.memilavi.com

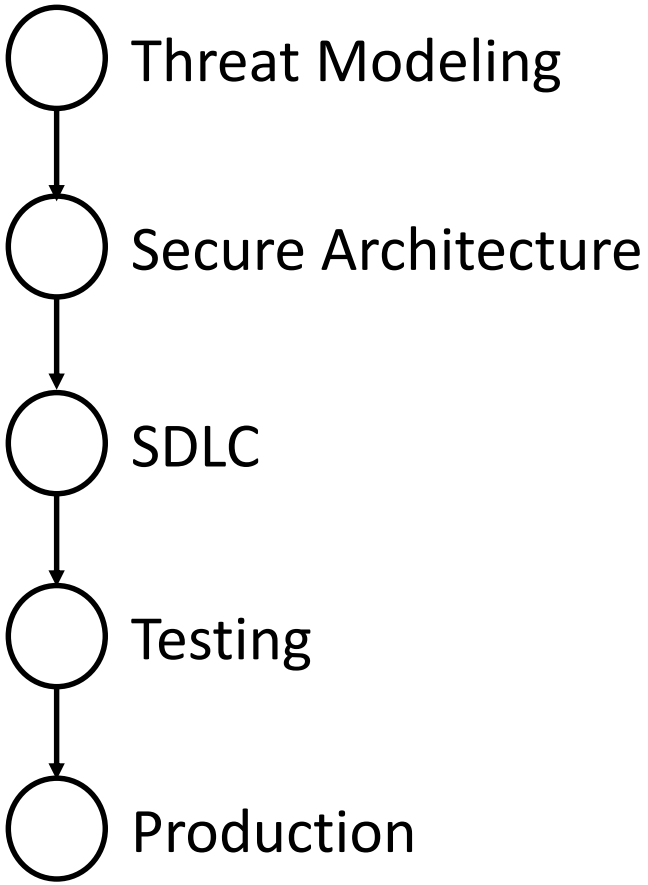
Case Study

- Apply the Secure Architecture Process on a real world app
- Great opportunity to exercise the knowledge
- Make sure you understand everything

How We're Going to Do That

- Present a fictitious system
- Take a look at its architecture
- Start securing it ☺

Secure Architecture Process



Use Case Study #1 from the Case Studies Course (Dunderly)

Threat Modeling

- Based on 4 core questions:
  - What do we build?
  - What can go wrong?
  - How can we mitigate that?
  - Did we succeed?

What Do We Build

- An internal HR system to manage employees, salaries and vacations
- Web based
- Has interface to external systems

What Can Go Wrong

- Sensitive data (mainly salaries) might leak
- Unauthorized access might lead to unauthorized data changes
- Payment data might be tampered on the way to the external system

How Can We Mitigate That

Sensitive Data Might Leak

- Implement Least-Privilege authorization in the database
- Encrypt sensitive data

Unauthorized Access

- Implement strong authentication for any access to the data
- Utilize extensive logging & monitoring

Payment Data Tampering

- Encrypt the data
- Use secure communication

Did We Succeed?

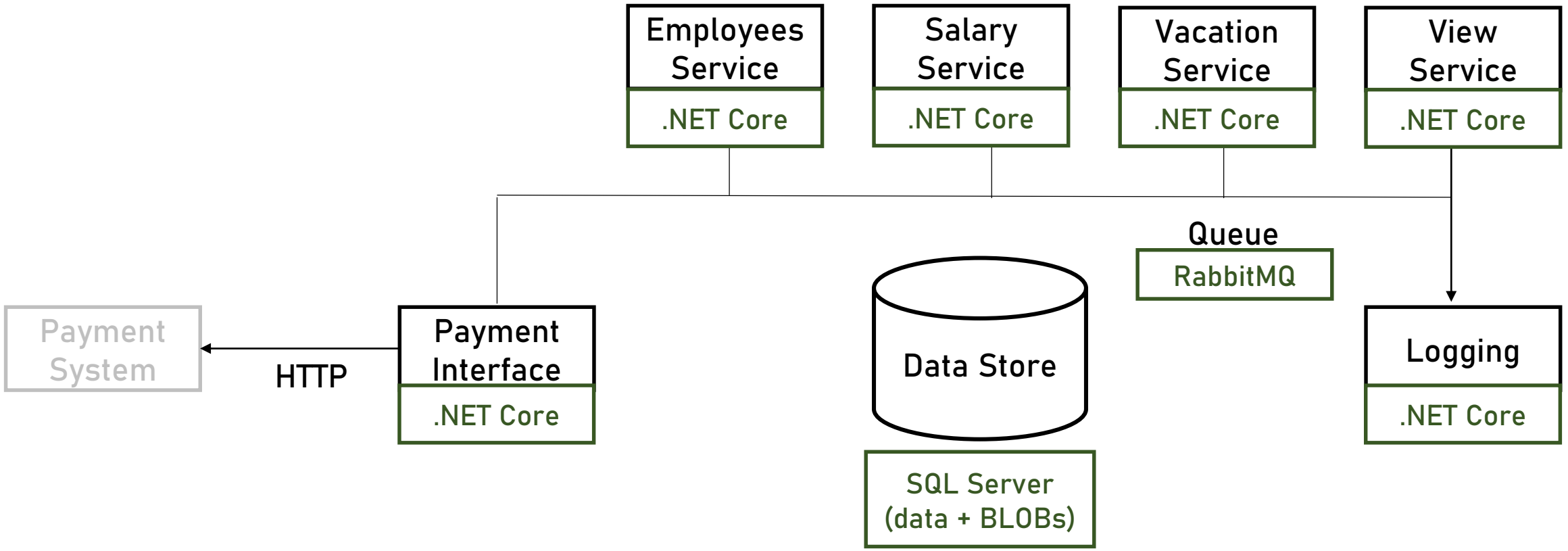
- We'll see…

Secure Architecture

- We'll go through the threats defined in threat modeling
- For each one of them, we'll design a security measure
- See how it fits in the architecture

Dunderly

Technical Diagram



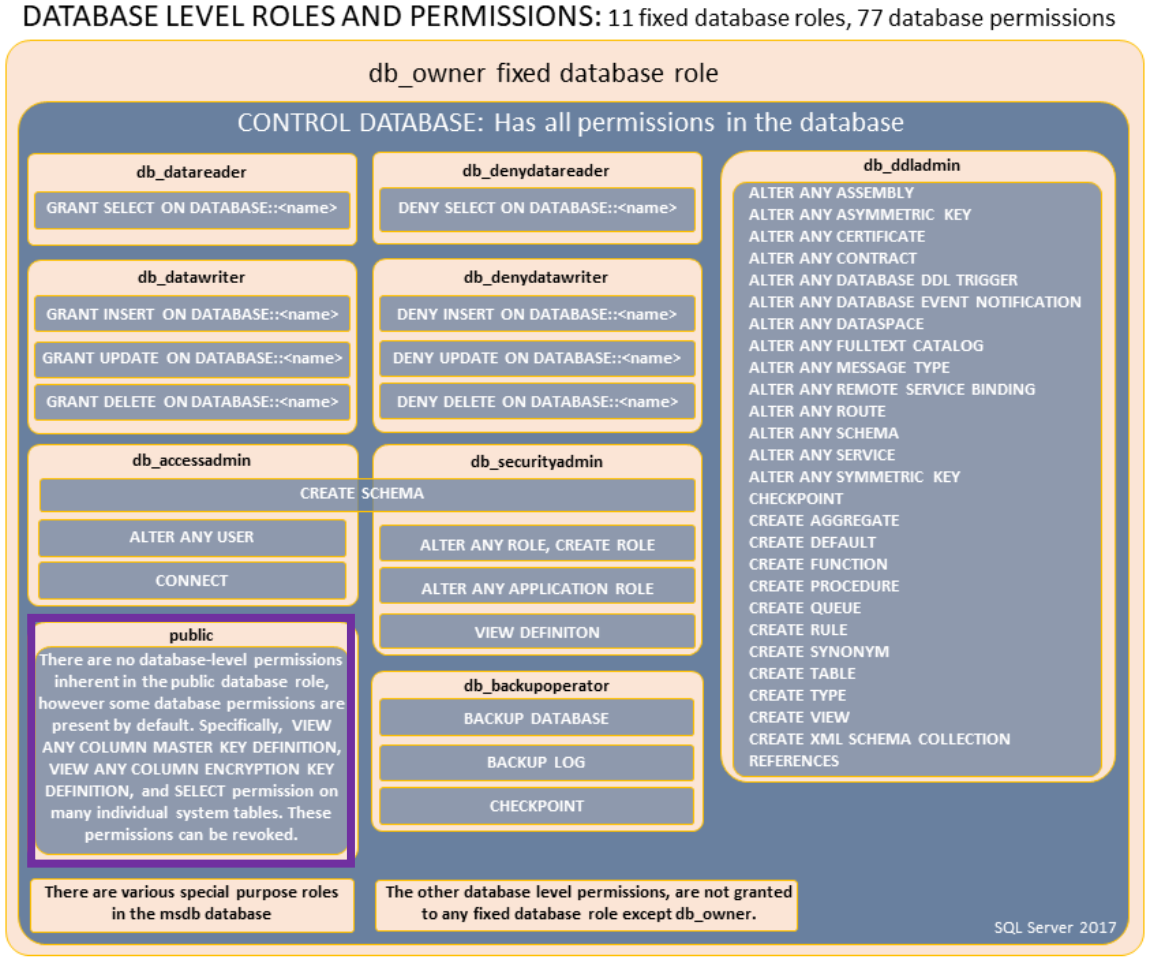
How Can We Mitigate That

Sensitive Data Might Leak

- Implement Least-Privilege authorization in the database
- Encrypt sensitive data

Least Privilege Authorization

- Database is SQL Server, so…



Source: https://docs.microsoft.com/en-us/sql/relational-databases/security/authentication-access/database-level-roles?view=sql-server-ver15

- Database-level app-user role should be ‘public’
- Grant access per table
- Consult with the DBA

Source: https://docs.microsoft.com/en-us/sql/t-sql/statements/grant-object-permissions-transact-sql?view=sql-server-ver15

Encrypt Sensitive Data

- Database is SQL Server, so…
  - Implement Always Encrypted
  - https://docs.microsoft.com/en-us/sql/relational-databases/security/encryption/always-encrypted-database-engine?view=sql-server-ver15
  - Decide on the Key Store
    - For internal apps: Windows Certificate Store
    - For public apps: Azure KeyVault

How Can We Mitigate That

Unauthorized Access

- Implement strong authentication for any access to the data
- Utilize extensive logging & monitoring

Strong Authentication

- Implement two-factor authentication
- .NET Core has excellent support for active directory integration, so…

Strong Authentication

- If app is public:
  - Implement MFA using Azure AD
- If not:
  - Implement MFA using AD + 3rd party authentication engine
    - RSA, Centrify etc.

Logging and Monitoring

- Already implemented in the architecture
- Make sure to include security-related log records
  - Validation problems, log-ins data

How Can We Mitigate That

Payment Data Tampering

- Encrypt the data
- Use secure communication

Encrypt the Data

- Encrypt data in code since it's not a database
- Store key in Windows Certificate Store or Azure KeyVault
- Use the System.Security.Cryptography library

Use Secure Communication

- Use TLS communication with the external system
- Make sure the system supports it

Secure Development Lifecycle

- Work closely with the dev team
- Make sure they're aware of security-related code practices

Testing

- Help shape the penetration testing
- We need to test for:
  - Fraudulent log-ins
  - Database authorizations
  - Data encryption

Testing

- No need for DDoS testing
- The system is internal

Penetration Testing

- Test type:
  - We'll go for white box
  - Internal system -> the attacker will probably have a lot of info about the system

Testing Results

- The attacker succeeded to log-in with someone's else credentials
- Analysis showed that MFA was not activated
- Make sure MFA is activated and re-test

Production

- Follow SQL Server vulnerabilities
  - https://www.cvedetails.com/vulnerability-list/vendor_id-26/product_id-251/Microsoft-Sql-Server.html
- Follow .NET Core vulnerabilities
  - https://www.cvedetails.com/vulnerability-list/vendor_id-26/product_id-43007/Microsoft-.net-Core.html
- Participate in Security Reviews

Case Study Summary

- We went through all the steps
- Main work was in the Threat Modeling and Secure Architecture
- Your involvement in important in all stages
- This is THE way to treat security!