IT University of Copenhagen

(Thesis)

Information-Flow Secure Programming on Matrix: A Case Study

Authors:
Ans Uddin anud@itu.dk

Supervisor:
Willard Rafnsson

Co-supervisor:
Carsten Schürmann

February, 2018

Contents

1 Introduction

1.1 Data privacy and protection

With GDPR becoming effective in 2018 the focus on data privacy is at its peak. Privacy violation is when sensitive data is exposed to unauthorized actors. OWASP top ten ranks *sensitive data exposure* as 3rd biggest security threat.

Recent cases of data leakage has put more attention on data privacy and protection. Some cases are due to poor security measures and could arguably have been prevented. Examples of cases are:

- The infamous Facebook - Cambridge Analytica scandal. Third parties were able to collect data through Facebook Login API.
- Google Plus leak. 500.000 users private data was exposed to third parties through APIs.
- Medicaid leak. A medical assistant had accessed patients' health records and exchanged mails with another employee containing the patients' private data.

The cases above failed to achieve end-to-end security and the improper handling of sensitive data could have been prevented with appropriate security policies and enforcement technique that enforces these policies.

There is more awareness on how applications deal with data. This add extra concern to the programmer and the application being developed on how sensitive data is handled and protected.

The well-known security enforcement techniques like access controls, firewalls and encryption are inadequate alone and does not ensure end-to-end security.

1.2 Information Flow Control

There exist useful security enforcement mechanisms for protecting confidential information such as firewalls, encryption and access control. However, these mechanisms each have their drawbacks.

- *Access* control prevents unauthorized access to information but once access is granted there is no guarantee how that confidential information is handled.
- *Firewall* limits communication from the outside hence isolate and protect information. Yet the firewall have no way of telling if the communication going through violates confidentiality.

- *Encryption* secures information on a channel with only the endpoints being able to access that information. However there is no assurance that once the data is decrypted that the confidentiality of that information is ensured.

The mechanisms mentioned above all have in common that they lack control of how the information flows. Information-flow security aims at protecting confidentiality and integrity of information by enforcing security policies. Information-Flow Control allows the programmer to define and enforce policies in a language-based way.

1.3 Matrix

Matrix is an open standard protocol for messaging over HTTP and synchronizing data. Matrix provides secure real-time communication over a decentralized federated network. Matrix secures data by providing end-to-end encryption.

Matrix cover use cases such as instant messaging, VoIP, Internet of Things communication and is generally applicable anywhere for subscribing and publishing data over standard HTTP API.

The fragmentation of IP communication is the problem Matrix essentially wants to solve. Making calls and messages between users needless of which app they use. However they define their longer term goal as *"to act as a generic HTTP messaging and data synchronisation protocol for the whole web"*.

1.4 The case study

The goal of the case study is to make secure implementation of a prototype using Information-Flow Control. The case study will use Matrix as the communication channel and strengthen the security at the endpoints using IFC.

Journal system

The prototype implements a journal system and is loosely based on the Danish E-journal system.

Medical privacy is a well-known issue. Sensitive data about patients needs to be handled carefully. Patients have access to their medical records through E-journal. A patient's journal on E-journal is available to 90.000 different medical employees.

There are clear policies about who and under what conditions should access a journal. It is legally required that an employee accessing the journal must have the patient in care and that the lookup must be relevant for the employee. Safety measures have been applied through logging and audit trails with random sampling checks however they do not prevent access to journals. Any medical employee can access the patient journal and even if prevention mechanism were established there would be no limitation to what a medical employee could see once access was granted.

The mechanisms in the current journal system might restrain malicious intend. However it does not guarantee prevention of unintentional access or disclosure of information. What is missing is the enforcement of secure information flow policies. Unintentional access or disclosure of information can be prevented by enforcing policies that define secure information flow.

The prototype will model the scenario of hospitals with different actors accessing a patient journal. The bulk of information on the journal system is extracted from newspaper articles hence there is a high uncertainty of how the system really works. Therefore many assumptions are made and the scenario is simplified when programming the prototype.

Scope

The objective of the project is to do a secure implementation of the prototype described above. Secure exchange of patient journal is ensured using Matrix and the endpoints are secured using IFC.

A successful project is one that fulfills these criteria:

- Evaluation of Matrix security model
- Survey of IFC tools and selection of tool.
- Implement a prototype distributed system running on Matrix, using the chosen tools
- Demonstrate increased security guarantee with Matrix and IFC

Why Matrix?

In the Digital Strategy 2016-2020 the Danish Agency of Digitisation defines initiative 7.2 as "Common standards for secure exchange of information". The large number of software systems in the Danish public sector has created a need for an uniform way of exchanging data across different application in a secure manner.

The initiative has similarities to the issue Matrix is trying to solve with fragmented IP communication. With Matrix security guarantees and their long term goal as a generic HTTP messaging protocol there is a strong case for using Matrix as a communication channel in this case study.

1.5 Threat model

The threat model is defined in the context of confidentiality and integrity.

- The adversary has the ability to observe information sent over the network.
- The adversary can generate input to the system .
- The adversary can observe public output.

1.6 Contribution

The contributions to the field are the findings of secure implementation using Paragon and how they compare to similar findings from secure implementation with JIF.

The thesis also contributes with the interface created between Paragon and Matrix making it possible to develop other secure applications on top of secure communication channel Matrix provides.

1.7 Structure of thesis

Chapter 2 sets the foundation for the thesis and introduces relevant information and background. Chapters 3 analyzes the Matrix security model and survey IFC tools. Chapter 4 goes in depth with design of the solution. The results are then presented and discussed in Chapter 5. The thesis is wrapped up in the conclusion section Chapter 6.

2 Background

2.1 Information Flow Control

Confidentiality

Integrity

The Lattice Model

Noninterference

Static policies

Dynamic policies

Declassification

Taking some specific information and changing it to a lower security classification.

Identify: What to classify, who declassifies, where the declassification happens and when the declassification happens

2.2 Matrix

Architecture

Client/Server API

Security model

3 Analysis

3.1 Survey of IFC Tools

JIF

Fabric

Paragon

JSFlow

Not possible to use Matrix library with JSFlow because of missing support for libaries such as require (in node). Also overhead with configuring JSFlow to be the interpretor.

Swift, SIF, FlowR, JFlow, LIO

Selection of IFC tool

The selection of IFC tool put emphasis on the practical usage in combination with Matrix.

3.2 Evaluation of Matrix security model

4 Design

4.1 Object Oriented Analysis and Design

Requirement

Analysis

Design

4.2 Matrix Design

Design choices

5 Results

5.1 Implementation

5.2 Discussion

6 Conclusion

Appendices

Bibliography

[1] Website example. `https://google.com`. Accessed: xxxx-xx-xx.

[2] John Doe. *Lorem ipsum*. Unknown, 2018. ISBN 0521865719, 9780521865715.