WYZSZA SZKOLA BANKOWA W POZNANIU

BACHELOR THESIS

The Mango Messenger

Authors:
Petro Kolosov, Serhii Holishevskyi, Illia Zubachov, Arslanbek Temirbekov

Supervisor:
Dr. Szymon Murawski

A thesis submitted in fulfillment of the requirements for the degree of Bachelor of Computer Science

in the

Wyzsza Szkola Bankowa w Poznaniu
Department of Computer Science

July 11, 2021

Declaration of Authorship

We, Petro Kolosov, Serhii Holishevskyi, Illia Zubachov, Arslanbek Temirbekov, declare that this thesis titled, “The Mango Messenger” and the work presented in it are my own. We confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

Partner Details

Mentor's details

First name and surname	Szymon Murawski
Degree	
Date and signature	

Team members' details

First name and surname	Petro Kolosov
Course of study	
Type of study program	
Date and signature	

First name and surname	Serhii Holishevskyi
Course of study	
Type of study program	
Date and signature	

First name and surname	Illia Zubachov
Course of study	
Type of study program	
Date and signature	

First name and surname	Arslanbek Temirbekov
Course of study	
Type of study program	
Date and signature	

"I fear not the man who has practiced 10,000 kicks once, but I fear the man who has practiced one kick 10,000 times."

Bruce Lee

WYZSZA SZKOLA BANKOWA W POZNANIU

Abstract

Computer Science
Department of Computer Science

Bachelor of Computer Science

The Mango Messenger

by Petro Kolosov, Serhii Holishevskyi, Illia Zubachov, Arslanbek Temirbekov

Among many types of social network applications, instant messaging is one of the applications that consider the privacy and the security are two crucial features due to that data exchanged between users are often private and not for public. In this work, a secure Instant Messenger (IM) mobile application is designed and implemented. Many techniques are used to provide privacy and another to achieve security through suitable cryptographic method. The limited and varied specifications of users' mobile devices are considered for implementing the concept of end-to-end encryption. The application also providing the main functions of instant messaging applications such as profile creation, access control management, and finding friend.

Acknowledgements

We would like to thank our mentor Szymon Murawski for his useful comments and suggestions and support over the whole process of writing this thesis.

Contents

Declaration of Authorship **i**

Abstract **iv**

Acknowledgements **v**

1 Introduction **1**
1.1 General overview of IM systems 1
1.2 Security vulnerabilities of IM systems 1
1.2.1 Revealing confidential information 1
1.2.2 Spreading viruses and worms 1
1.2.3 Exposing the network to backdoor Trojans 2
1.2.4 Denial of Service Attacks 2
1.2.5 Hijacking Sessions 2
1.2.6 Copyright infringement 2

2 Secure IMS Requirements **4**
2.1 Functional requirements 4
2.2 Non-functional requirements 4

3 Secure IMS Implementation **5**
3.1 Application architecture and UML modeling 5
3.1.1 Motivation 5
3.1.2 Initial concept diagram and discussion 6
3.1.3 Planned technologies 6
3.1.4 Monolith Architecture: Cons and Props 6

4 Mango Messenger **8**
4.1 Web API Documentation 8

Bibliography **9**

List of Abbreviations

IMS	Instant Messaging System
IM	Instant Messenger
CQRS	Command Query Responsibility Segregation

Chapter 1

Introduction

1.1 General overview of IM systems

Nowadays, the Instant Messaging Systems (IMS) became the most widely-used and convenient way of communication between people via Internet. These systems offer a simple and inexpensive way to continuance existing relationships and forming new by providing an attractive means for sharing information and digital social interactions. The quick development of IMS and the widening of their popularity sometimes moves the focus from possible security risks. In the worst case, IMS exposes vulnerable to security and privacy channels to hackers and intruders [1] [2]. In existing IMS, there are multiple privacy and security issues that need to be resolved in order to protect user's confidential information and shared data via these messaging applications [3]. Source [3] gives an analysis of Telegram Messenger and the related MTProto Protocol with cryptography behind Telegram. Moreover, an overview of current security status for some major IMS is provided. Meanwhile, the researchers in [6] discussed types of threats on privacy of IMS and ranges of thereat effects for both, user and provider. In this thesis, the most major security threats of IMS is described. In order to reflect best practices we provide a prototype-application written as example.

1.2 Security vulnerabilities of IM systems

There are numerous risks associated with the use of IM and as with any form of electronic communication one must take certain steps to mitigate those risks. Such risks include:

1.2.1 Revealing confidential information

Revealing confidential information over an unsecured delivery channel. Public Instant Messaging transmits unencrypted information, so it should never be used for sensitive or confidential information. The information is on the Internet and may be accessed by anyone.

1.2.2 Spreading viruses and worms

Instant Message (IM) programs are fast becoming a preferred method for launching network viruses and worms. The lack of built-in security, the ability to download files and built-in "buddy list" of recipients create an environment in which viruses and worms can spread quickly. The threat is growing so fast that IM is quickly catching up to e-mail as a primary point of attack.

1.2.3 Exposing the network to backdoor Trojans

1.2.4 Denial of Service Attacks

In computing, a denial-of-service attack (DoS attack) is a cyber-attack in which the perpetrator seeks to make a machine or network resource unavailable to its intended users by temporarily or indefinitely disrupting services of a host connected to the Internet. Denial of service is typically accomplished by flooding the targeted machine or resource with superfluous requests in an attempt to overload systems and prevent some or all legitimate requests from being fulfilled, see Gu and Liu, 2007. In a distributed denial-of-service attack (DDoS attack), the incoming traffic flooding the victim originates from many different sources. This effectively makes it impossible to stop the attack simply by blocking a single source. A DoS or DDoS attack is analogous to a group of people crowding the entry door of a shop, making it hard for legitimate customers to enter, thus disrupting trade.

Criminal perpetrators of DoS attacks often target sites or services hosted on high-profile web servers such as banks or credit card payment gateways. Researches at Prince, 2016; Halpin, 2012 conclude that revenge, blackmail and activism can motivate these attacks.

1.2.5 Hijacking Sessions

In computer science, session hijacking, sometimes also known as cookie hijacking is the exploitation of a valid computer session—sometimes also called a session key—to gain unauthorized access to information or services in a computer system. In particular, it is used to refer to the theft of a magic cookie used to authenticate a user to a remote server. It has particular relevance to web developers, as the HTTP cookies used to maintain a session on many web sites can be easily stolen by an attacker using an intermediary computer or with access to the saved cookies on the victim's computer (see HTTP cookie theft). After successfully stealing appropriate session cookies an adversary might use the Pass the Cookie technique to perform session hijacking. By Bugliesi et al., 2015, the cookie hijacking is commonly used against client authentication on the internet. Modern web browsers use cookie protection mechanisms to protect the web from being attacked.

A popular method is using source-routed IP packets. This allows an attacker at point B on the network to participate in a conversation between A and C by encouraging the IP packets to pass through B's machine.

If source-routing is turned off, the attacker can use "blind" hijacking, whereby it guesses the responses of the two machines. Thus, the attacker can send a command, but can never see the response. However, a common command would be to set a password allowing access from elsewhere on the net.

An attacker can also be "inline" between A and C using a sniffing program to watch the conversation. This is known as a "man-in-the-middle attack".

1.2.6 Copyright infringement

Copyright infringement (at times referred to as piracy) is the use of works protected by copyright law without permission for a usage where such permission is required, thereby infringing certain exclusive rights granted to the copyright holder, such as the right to reproduce, distribute, display or perform the protected work, or to make derivative works. The copyright holder is typically the work's creator, or a publisher or other business to whom copyright has been assigned. Copyright holders routinely invoke legal and technological measures to prevent and penalize copyright infringement.

Copyright infringement disputes are usually resolved through direct negotiation, a notice and take down process, or litigation in civil court. Egregious or large-scale commercial infringement, especially when it involves counterfeiting, is sometimes prosecuted via the criminal justice system. Shifting public expectations, advances in digital technology and the increasing reach of the Internet have led to such widespread, anonymous infringement that copyright-dependent industries now focus less on pursuing individuals who seek and share copyright-protected content online,[citation needed] and more on expanding copyright law to recognize and penalize, as indirect infringers, the service providers and software distributors who are said to facilitate and encourage individual acts of infringement by others.

Estimates of the actual economic impact of copyright infringement vary widely and depend on other factors. Nevertheless, copyright holders, industry representatives, and legislators have long characterized copyright infringement as piracy or theft – language which some US courts now regard as pejorative or otherwise contentious, see Powell Jr et al., 1984; Li and Correa, 2009.

Chapter 2

Secure IMS Requirements

2.1 Functional requirements

2.2 Non-functional requirements

Chapter 3

Secure IMS Implementation

3.1 Application architecture and UML modeling

3.1.1 Motivation

As a programmers, I believe all we have faced the cases of crucial over-engineering during the implementation of some software product. For the programmer, it is a vital point to follow two separated, but closely related software development principles, such that KISS (Keep It Simple and Stupid), and YAGNI (You Aren't Gonna Need It). As the main topic of our thesis is the security and privacy aspects of Instant Messaging Systems, we consider following previously discussed principles KISS and YAGNI and use a well-known Monolithic architecture. One would suggest to use nowadays popular Microservice Architecture, thinking about scalability, an ability of the system to handle large numbers of users distributed over geographically large areas without notably affecting the overall performance of the system. However, the effect of Microservices is felt only for quite large and complex systems, not the case of Instant Messaging System we implement in chapter [number]. It is worthless to divide the functional requirements, discussed in section [number] into microservices and that's central point in motivation to use Monolithic Architecture. Following plot demonstrates the relation between complexity of system and architecture.

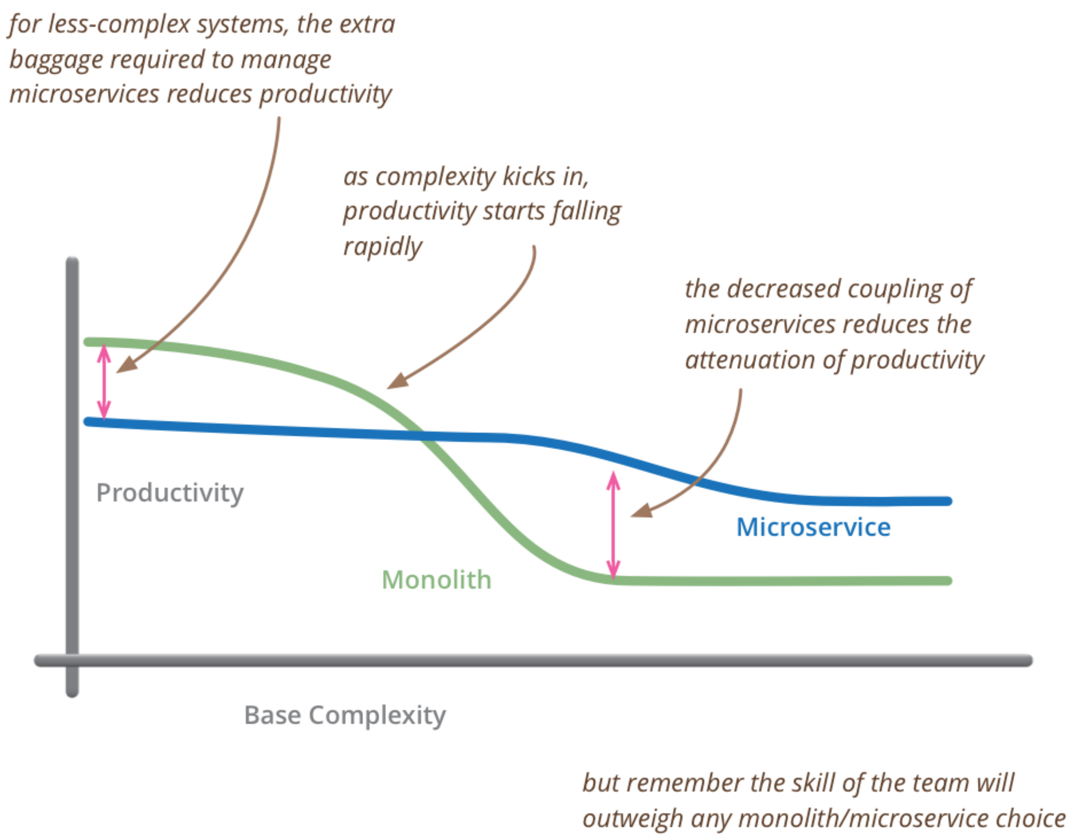

FIGURE 3.1: Relation between system complexity and architectures. Source: https://martinfowler.com/bliki/MicroservicePremium.html

3.1.2 Initial concept diagram and discussion

3.1.3 Planned technologies

3.1.4 Monolith Architecture: Cons and Props

A monolith is built as a large system with a single code base and deployed as a single unit, usually behind a load balancer. It typically consists of four major components: a user interface, business logic, a data interface and a database.

Monoliths offer several advantages, particularly when it comes to operational overhead requirements. Here are some of those basic benefits:

Simplicity: Monolithic architectures are simple to build, test and deploy. These apps can scale horizontally, in one direction, by running several copies of the application behind a load balancer. Cross-cutting concerns: With a single codebase, monolithic apps can easily handle cross-cutting concerns, such as logging, configuration management and performance monitoring. Performance: Components in a monolith typically share memory which is faster than service-to-service communications using IPC or other mechanisms.

But one major drawback of monolithic architectures is tight coupling. Over time, monolithic components become tightly coupled and entangled. This coupling effects management, scalability and continuous deployment. Other cons that stem from tight coupling include:

Reliability: An error in any of the modules in the application can bring the entire application down. Updates: Due to a single large codebase and tight coupling, the entire application would have to deploy for each update. Technology stack: A monolithic application must use the same technology stack throughout. Changes to the technology stack are expensive, both in terms of the time and cost involved.

Add here about CQRS

As the main concern we focus on is webb application, in the ongoing sections we discuss a details of both, front-end and back-end development of our project – Mango Messenger.

Chapter 4

Mango Messenger

4.1 Web API Documentation

Bibliography

Bugliesi, Michele et al. (2015). "CookiExt: Patching the browser against session hijacking attacks". In: *Journal of Computer Security* 23.4, pp. 509–537.

Gu, Qijun and Peng Liu (2007). "Denial of service attacks". In: *Handbook of Computer Networks: Distributed Networks, Network Planning, Control, Management, and New Trends and Applications* 3, pp. 454–468.

Halpin, Harry (2012). "The philosophy of Anonymous: Ontological politics without identity". In: *Radical Philosophy* 176.

Li, Xuan and Carlos María Correa (2009). *Intellectual property enforcement: international perspectives*. Edward Elgar Publishing.

Powell Jr, Lewis F et al. (1984). "Dowling v. United States". In.

Prince, Mathew (2016). *Empty DDoS Threats: Meet the Armada Collective"*.