Aalto University
School of Science
Master's Degree Programme in Security and Mobile Computing

Nadin Vazquez Torralba

Security Analysis of Mobile Payments: Direct Carrier Billing

Master's Thesis
Espoo, July, 2017

Supervisor: Prof. Tuomas Aura, Aalto University
Prof. Danilo Gligoroski, NTNU
Instructor: M.Sc. (Tech.) Thanh Bui , Aalto University

Aalto University
School of Science
Master's Degree Programme in Security and Mobile Computing

ABSTRACT OF MASTER'S THESIS

Author:	Nadin Vazquez Torralba
Title:	Security Analysis of Mobile Payments: Direct Carrier Billing
Date:	July, 2017 **Pages:** 81
Major:	Security and Mobile Computing (T3011)
Supervisor:	Prof. Tuomas Aura, Aalto University Prof. Danilo Gligoroski, NTNU
Instructor:	M.Sc. (Tech.) Thanh Bui , Aalto University

Payments are a compensation for a product or a service received. The funds are transferred from one party (consumer) to another (merchant). Mobile payments are a particular form of electronic payment where a mobile device serves as the key instrument to initiate, authorize or complete a payment. The payment methods have been continuously changing to adjust to cashless trends. Seeking to reach a larger number of customers has promoted the development of different solutions to provide means of payment. With an increasing number of mobile subscribers, mobile solutions such as carrier billing, SMS-based payments, and mobile wallets are gaining importance, permeating different markets, such as public transportation, digital content, advertisements and charity.

This thesis investigates and analyses mobile payment solutions. The main purpose is, primarily, to identify and describe the security protocols that occur during the payment transaction. Subsequently, to distinguish the mechanisms utilised to identify and authenticate consumers and the mechanisms providing integrity to the payment data. Additionally, to recognize the possible security threats overlooked during the design and deployment of payment solutions.

The analysis and tests carried out showed opportunity areas for the service providers to improve the security level of their services. We found vulnerabilities that jeopardise the integrity and authenticity of transactions from the merchant and consumer sides. The major vulnerabilities found lead to conclude that despite the development of protocols and technologies to strengthen security, an appropriate analysis is required to design and develop secure solutions. Neglecting security requirements in exchange for simplicity could come at a high price for the parties involved in mobile payments, specially, in direct carrier billing.

Keywords:	Security, Direct Carrier Billing, payment methods, threats, mobile, payments, risks, forgery, tethering
Language:	English

Acknowledgements

The realisation of this thesis would not have been possible without the support and assistance of several individuals throughout my Master's degree. I wish to acknowledge my deepest appreciation to each of them.

Firstly, I would like to express my infinite gratitude to the NordSecMob consortia for the admission to take part in the programme, and the European commission for granting me the Erasmus+ scholarship, making my participation possible.

My supervisors, specially Professor Tuomas Aura, to whom I extend special appreciation for his interest, excitement and continuous guidance during the thesis project.

My friends, for their continuous interest in my studies and well-being during this time.

Finally, but not less important, I want to thank my family for their relentless support and encouragement to pursue my Master's degree.

Espoo, July, 2017

Nadin Vazquez Torralba

Abbreviations and Acronyms

API	Application Programming Interface
CDMA	Code-Division Multiple access
CSS	Cascading Style Sheets
GSM	Global System for Mobile Communications
HMAC	keyed-Hash Message Authentication Code
HTML	HyperText Markup Language
HTTP	Hyper Text Transfer Protocol
HTTPS	Hyper Text Transfer Protocol Secure
J2ME	Java 2 Micro Edition
MNO	Mobile Network Operator
MSISDN	Mobile Station International Subscriber Directory Number
NFC	Near Field Communication
OS	Operating System
OTP	One-Time password
PC	Personal Computer
PIN	Personal Identification Number
SANS	System Administration, Networking, and Security
SIM	Subscriber Identity Module
SMS	Short Message Services
SMSC	Short Message Services Center
SSL	Secure Sockets Layer
TLS	Transport Layer Security
UMTS	Universal Mobile Telecommunications System
URL	Uniform Resource Locator
USSD	Unstructured Supplementary Service Data
VAT	Value Added Tax
WLAN	Wireless Local Area Network
XML	Extensible Markup Language

Contents

List of Figures

Chapter 1

Introduction

The beginning of mobile payments dates from back to 1997 when, in Finland, the telecom operator Sonera allowed consumers to buy sodas from vending machines using mobile phones and paying for them through the operator's bill [21]. Ever since, a variety of solutions have been developed to keep pace with digitalization and new use cases. The technologies include, gateway-based payment services, messaging-based services, stored-value-based services and mobile identification and authorization based payment services [7].

As the number of mobile subscriptions continues to grow in most countries [2], special attention has been dedicated to the development of mobile payment solutions. In the process of developing mobile payment solutions, service providers realised about the limitation to penetrate in larger markets by focusing on direct debit and credit card payments only. The limitation is the result of a global market where mobile phone users outnumber bank account holders. Thus, payment service providers worked together with mobile operators to enable direct carrier billing for their customers. In different parts of the world the direct carrier billing payment method has had wide acceptance and continues to grow [8]. In the Scandinavia region where mobile subscriptions already surpass 1.7 per capita [30], applications for public transportation tickets, electronic banking, contest voting, peer-to-peer transfers, and purchases of goods and services from online shops are already highly popular. These applications use mobile payment solutions that offer direct debit, credit card, SMS, and carrier billing payment methods.

Problem Statement

Some of the payment solutions available for online purchases require a single click from the consumer to execute the payment, which, from the usability

point of view, is a rather simple process. However, simple payment solutions have brought problems to consumers in the past. Dimoco, a global service provider for direct carrier billing, received several complaints in 2015 for serving as an interface for fraudulent merchants to make unauthorised charges to subscribers of several mobile operators [6][22]. Therefore, it is necessary to investigate and analyse mobile payments from an IT security point of view and determine if they truly offer the security required by such sensitive transactions between consumers and merchants. Thus, the specific goals of this thesis are:

1. Investigate and understand the different mobile payment technologies and solutions currently available.

2. Find specific mobile payment service providers and merchants to analyse the security protocols used for the mobile payments.

3. Identify threats and potential design weaknesses and, if found, define attack scenarios to demonstrate the vulnerabilities.

4. Report the findings and identify security requirements that need to be strengthened by the implementations studied.

Structure of the thesis

The remainder of the thesis is organized in chapters. Chapter 2 sets the foundation to understand mobile payments, it presents the different technologies, methods and security requirements for mobile payment services. Additionally, Chapter 2 describes the protocol followed for the completion of payment transactions in some well-known global payment solutions. Chapter 3 presents the security analysis made to two different Finnish payment service providers accepting direct carrier billing as a mobile payment method. The security analysis of Chapter 3 is divided into sections to describe the characteristics, adversary model and threats of the services studied. Chapter 4 evaluates the services under study and lists the security requirements met by the service providers. A discussion about the results of the security analysis is made in Chapter 5. Finally, Chapter 6 concludes the thesis.

Chapter 2

Background

This chapter establishes the foundations to understand the main topic of this thesis, mobile payments. The chapter starts by defining mobile payments and the entities involved in the execution of payments. It then presents the technologies and methods available to carry out such payments.

Mobile Payments

A payment is a financial transaction where a compensation is deposited for a product (digital/physical) or a service received. In the execution of a payment, funds are transferred from one entity to another.

Mobile payments (M-payments) are a special form of payment where a mobile device such as a mobile phone, smartphone or any wireless device ´serves to complete, securely, at least one phase of the financial transaction over a mobile network or other wireless technology [9].

Several technologies and protocols have enabled the development of mobile payments [23]. In the following sections, different mobile payment technologies and solutions are presented.

Entities

To complete a payment, different entities participate in the process (Fig. 2.1). In the simplest model of mobile payment, a minimum of four parties are required:

- **Merchant (payee):** The one selling or offering products and services, the merchant is the entity whose account is credited.

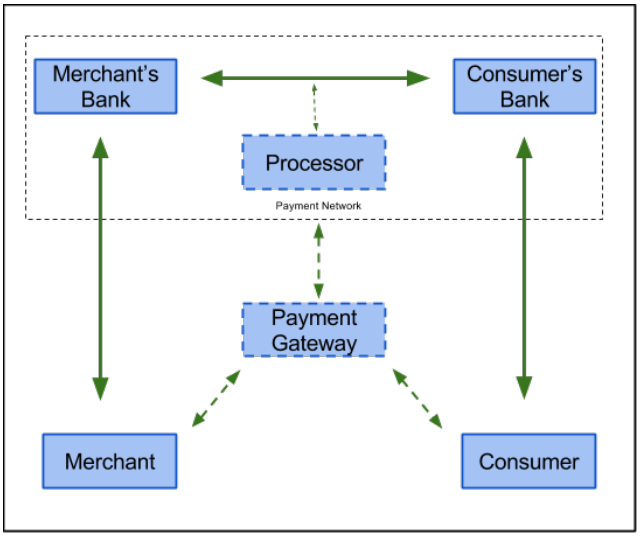

Fig. 2.1: Entities participating in the execution of a payment.

- **Consumer (payer):** The one benefiting from the from goods or services of the merchants, the consumer is the entity whose account is debited.
- **Bank of the merchant:** The institution providing a merchant account capable of receiving funds. Typically, the institution is a commercial bank. However, other institutions offer merchant accounts and financial transaction processing services.
- **Bank of the consumer:** The institution providing consumers with credit or debit accounts. Typically, the institution is a commercial bank. However, other credit providers can play the role.

Alternative models add entities to the payment process. Instead of making the financial transactions directly, some intermediaries can take part in the process:

- **Payment processor:** Authorized entity coordinating and processing transactions among financial institutions. The processor, in conjunction with the banks of the payer and payee, form the payment network.
- **Gateway:** Serves as the connection between the merchant and consumer and the payment network.

The figures of the processor and the gateway are not mandatory to complete payments and can be present as a single entity.

Communication Technologies for M-Payments

Short Message Services(SMS)

The SMS is a protocol that allows messages of up to 160 characters to be exchanged between mobile phones and a SMS centre (SMSC) using the GSM communications network. The mobile device connects to the base station of the mobile network provider to access the GSM network, in the network, the message is forwarded to the SMSC. From the service centre, the information in the message is forwarded to the merchant and/or payment processor (Fig. 2.2). Payments based on SMS use mobile terminated (sent to mobile phone) or mobile originated (sent from mobile phone) messages. The merchant must request a mobile operator for a phone number or short code [1] capable to receive and/or send messages. The merchant can choose to use the number to receive payments, in which case the operator will bill a premium fee to the consumer for sending messages to that number and credit the merchant for an amount previously agreed. From the SMSC or SMS gateway, the operator sends the information regarding the content of the message (i.e., order details) to the merchant server. Then, the merchant processes the order. Additionally, the merchant can choose to only receive information about the content of the message which can be the consumer bank account and order details. The SMSC sends the information to the merchant server and the merchant processed the order and payment.

Even though SMS uses a channel different from voice to access to the phone, it is vulnerable to different attacks, i.e., spoofing, message filtering and spam, especially on the operator side because it is a store-and-forward technology [12].

Unstructured Supplementary Services Delivery (USSD)

The USSD is a capability of the GSM standard to support the transmission of information over the GSM signaling network. USSD offers session-based communication, enabling a variety of applications including mobile payments. USSD is session and transaction-oriented technology; response times for interactive applications are shorter for USSD than SMS. The consumers indicate their intention to make a purchase to the merchant by sending an USSD request in the form of a command using their mobile device.

[1] Short code: 4-digit code map to a mobile number. Typically, the code is constant among mobile operators in one region

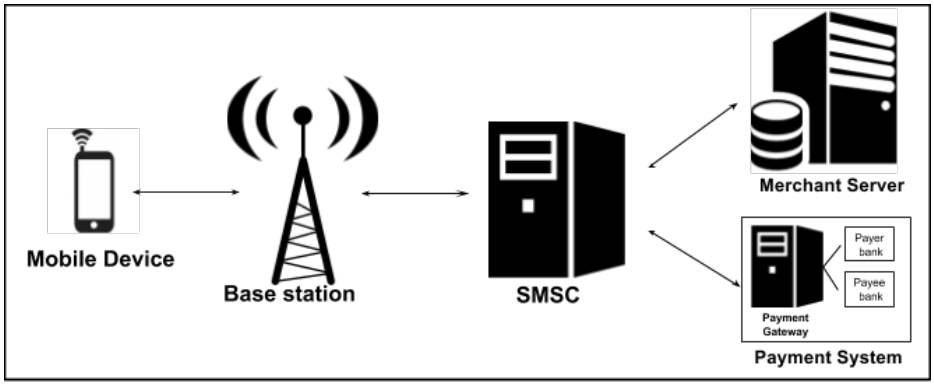

Fig. 2.2: Architecture of SMS-based mobile payment.

The commands are standardized; in a typical scenario of micropayment [2], the command has the following form:

[USSD code] * [phone number] * [transaction type] * [product ID] * [amount]

The mobile device connects to the base station of the network provider to access the GSM network. Once in the network, the command is forwarded to the USSD gateway. The USSD gateway communicates using the SS7 protocol. After receiving the requests, it creates a session and routes the information to the merchant server or payment system (Fig. 2.3). The response is received in XML form to finally confirm the transaction to the consumer in an USSD message. As in SMS-based payments, the mobile operator is tightly integrated to the payment process and provides services such as authentication and billing.

There are other forms of services [14] that use the USSD for person to person payments, mobile advertising and voting.

Mobile Broadband

Packet data over mobile networks enabled Internet communication services such as email and web browsing. Since the introduction of the GPRS (General Packet Radio Services), Internet packets can be transmitted to and from mobile devices using mobile networks. As wireless access technologies evolved, the transmission of data, voice and video acquired higher transmission rates, broadening the range of applicability of mobile devices to include mobile payments transactions over the internet, which considerably

[2]Micropayment: Financial transaction made online involving a very small amount of money.

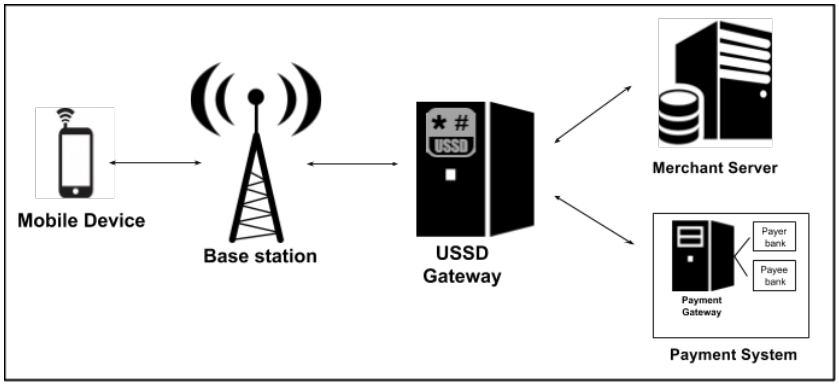

Fig. 2.3: Architecture of USSD-based mobile payment.

increased with the introduction of the 3G and 4G generation technologies such as UMTS and LTE, respectively.

To make a payment, the consumer sends the request through the LTE network. The request is routed to the merchant server. The transaction is completed by the payment gateway involving the banks of both parties, the consumer and the merchant. As will be shown in this thesis, there are major challenges in integrating operator-based authentication and billing to payment services over. This is because the operator is essentially acting as an Internet Service Provider (ISP) and it is not supposed to interfere with the transmitted data.

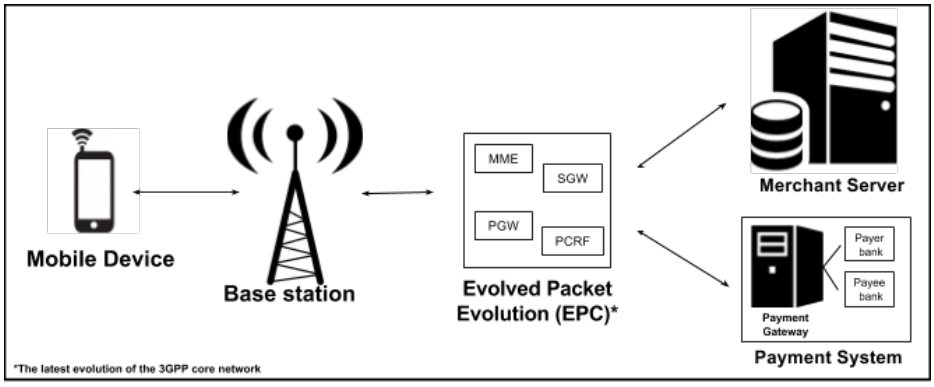

Fig. 2.4: Architecture of mobile broadband-based mobile payment.

Wireless Local Area Network(WLAN)

An alternative to mobile broadband for data communications in mobile devices are WLANs.

The Wireless Ethernet Standard family 802.11x establishes the specifications to implement WLANs. The typical and widely used technology for WLAN is known as Wi-Fi which is based on the 802.11x family and has evolved to offer greater data transmission rates. Mobile devices that support WLAN connectivity can access the internet to execute payment transactions with the merchants and bank servers. Due to the distributed management of WLANs, the network in this case cannot provide any special security features to support the payment process.

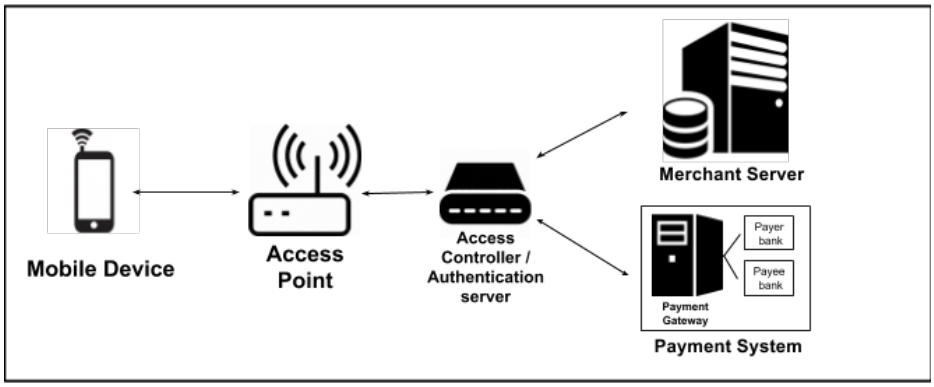

Fig. 2.5: Architecture of WLAN-based mobile payment.

Near Field Communication (NFC)

NFC is a short-range (few centimetres) wireless technology resulting from the combination of Radio Frequency IDentification (RFID) tags and a mobile phone. NFC allows three different modes of operation:

- Reader/Writer. The NFC enabled phones can read NFC tags in this mode.
- Card emulator. The mobile phone is used as a contactless card with no additional adaptors in the payment structure.
- Peer-to-peer (P2P). In this mode two NFC enabled devices can exchange data. NFC technology enables innovative applications such as ticketing systems and micro-payment systems [19].

To provide security, NFC possess a "Secure Element" (SE) that consist of a special location in the memory where trusted applications can store and retrieve sensitive information

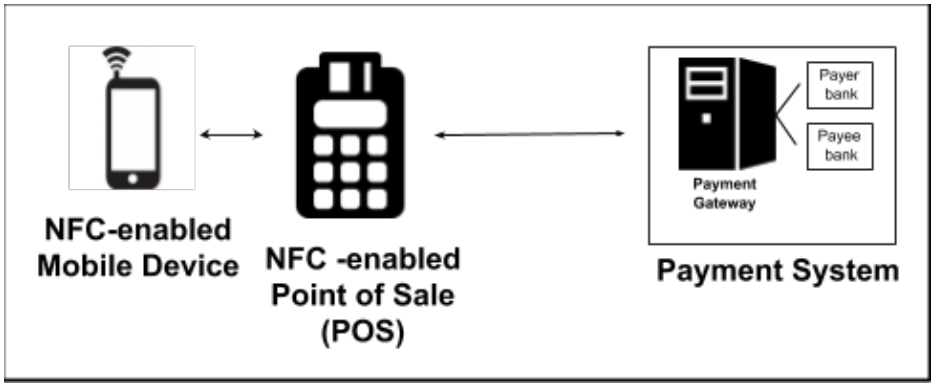

Fig. 2.6: Architecture of NFC Mobile Payment.

Software Platforms

In addition to the technologies available to transmit payment data and to process payments, the service providers leverage the software development technologies to develop clients for the payment systems in the following manners.

Mobile Applications

The client for a mobile payment system can reside on the mobile device of the consumer as an application. Depending on the architecture and design of the application it may be classified as phone-based and SIM-based.

Phone-based Application

The operating system of the mobile device determines the appropriate software stack to develop a native mobile application. The Android operating system requires the application to be developed in Java(J2ME) for GSM mobile phones or in Binary Runtime Environment for Wireless (BREW) for CDMA mobile phones. The operating system by Apple, iOS, requires the application to be developed in Objective-C.

SIM-based Application

The subscriber identity module (SIM) used in mobile phones, is a smart card with storage and processing capabilities. Typically, the SIM card is used to identify and authenticate the mobile device to the mobile operator network, however, the SIM Application Toolkit facilitates the mechanisms to develop

SIM-based applications. These applications can store sensitive information in the SIM card, the information in the card can be protected from access using cryptographic algorithms. Thus, SIM-based applications offer slightly better security than mobile application.

Mobile Wallet

A mobile wallet is a type of electronic wallet that resides in the mobile device as an application containing details about the consumer, such as, bank account and credit or debit card information, thus, allowing payments using a mobile device. Several proprietary implementations of wallets are used globally, e.g., Google Wallet and Apple Pay.

Web-based Applications

Merchants can avoid the endeavour of developing a mobile application and take advantage of the internet browsers available for mobile devices. Then, it is required to create a web application to serve as the client for the mobile payment system. Typically, a mobile client comprises JavaScript code, a style sheet (CSS), a HTML document and a presentation framework.

Mobile Payments Methods

Several payment solutions exist today. They can be classified by the technology they implement or by the type of payment they can execute. Usually, one solution results from the combination of different technologies and one or more methods of payment. The different payment methods can be classified per their connectivity [4] or per the type of payment realized. For the purposes of this thesis, the latter classification is listed below:

1. **Direct billing:** Banks, MNO and/or phone manufactures join to provide direct billing. The bank account and the phone number and mobile device of the consumer are linked. When a payment is executed, the bank account of the consumer is debited and the amount is credited to the account of the merchant.

2. **Credit card:** For this method, the credit card number is linked to the phone number (or mobile device) of the consumer. When the consumer makes a payment transaction with a merchant, the credit card is charged and the amount is credited to the account of the merchant.

3. **Direct Carrier billing:** Consumers agree to make payments to merchants for services or products using their mobile phone; the transaction is then charged in the mobile phone bill. The mobile operator identifies the user by the SIM card, allowing with this, the execution of payments without additional registration or signing in, resulting in the so called one-click payments [13]. This thesis focuses on this payment method, although we also investigate about the other payment methods.

Security Requirements

Independent of the technology and payment method adopted by a mobile payment solution, the technical security it provides and to what extent it is perceived as a secure solution becomes a significant aspect to determine its adoption and success, as several surveys show [7][28][20].

Due to the sensitivity of the transactions carried out by mobile payment solutions, these solutions should meet the following minimum security requirements [24][16], to effect secure mobile payments:

- **Identification.** The mobile payment solution should be able to identify the entities and subjects involved in the payment process. To achieve this requirement each entity should have a unique identifier. In a mobile payment system, the mobile phone number (MSISDN), a user ID or a bank account number qualify as identifiers.

- **Authentication.** Proof of identity is equally as important as identification; hence, the system should have the appropriate means to ensure an entity is who it claims to be. Three factors allow the system to verify the identity of an entity: Something it has (e.g., a token, a smart card, a certificate), something it knows (e.g., a password, a personal identification number (PIN)), or something it is (e.g., a fingerprint, voice, retina pattern).

- **Authorisation.** Limit and establish the minimum level of privileges each party owns to successfully execute transactions and control the access to the information flowing in the payment process.

- **Integrity.** End-to-end data integrity is required to guarantee messages exchanged between the trading parties in an open network contain the correct information about the transaction, this includes avoiding unauthorised data modifications as data traverses the network or while it is stored.

- **Confidentiality.** Information regarding the trading parties should remain private to the payment system to avoid its misuse. This is achieved with encryption. Several crypto algorithms are available to provide strong encryption.

- **Audit mechanisms and non-repudiation.** For audit purposes, logs of the transactions executed are required. The information these records contain may include a unique identifier, timestamps, the traders involved and the value of the transaction. By retaining audit trails, the traders should not be able to repudiate their participation in the transaction.

Meeting the requirements above mentioned has no trivial solution. In a mobile or wireless environment, constraints exist related to the computational capabilities of the mobile devices, power consumption and a large number of security threats inherent of the open-air transmission in wireless networks and roaming in untrusted access points. Furthermore, wireless networks are constrained by less bandwidth, greater latency and lower connection stability than wired networks. In an effort to protect consumers governmental and private organization have created regulation and standards for financial transactions, adding requirements for mobile payment solutions providers who must comply with global regulation such as the PCI Data Security Standard (PCI DSS) [3] and country-specific or local regulatory, such as MAPEL[4] in the case of Finland.

Global Payment Solutions

Different payment service providers exist today to facilitate platforms that enable electronic and mobile payments. These payment service providers emphasize the security of their services. Each provider has developed its own protocol to carry out mobile payments.

In this section, the protocol of three of the widely analysed and adopted [5][26] payment service providers are described.

[3] Security Standards Council for the PCI (Payment Card Industry), a global forum for the ongoing development, enhancement, storage, dissemination and implementation of security standards for account data protection.

[4] MAPEL (**MA**ksullisten **P**uhelinpalveluiden **E**ettinen **L**autakunta), Finnish self-regulatory for Toll Services

Stripe

Stripe is a payment processor that offers a complete toolkit for merchants to collect payments in a secure fashion. Stripe provides an REST API and extensive documentation for developers to integrate Stripe products into their web or mobile applications. The list of Stripe products comprises, among others, Checkout, Connect, Radar and Atlas [5].

The standard payment process using Stripe requires the merchant to integrate the Stripe Checkout form into its web or mobile application to collect data about the consumer and the payment. Payment data is sent directly to Stripe servers where it is verified. If the data is valid, Stripe will create a token to represent the payment data and send it to the merchant server, then, the merchant server can create a charge (transaction) to request for a payment using the token received by Stripe. The request is processed by Stripe who, in turn, requests the payment to the consumer credit provider (i.e. consumer bank). Finally, Stripe returns the status of the transaction (i.e., failed, successful, pending) back to the merchant server who can confirm the transaction to the consumer. The payment process flow is shown in figure 2.7.

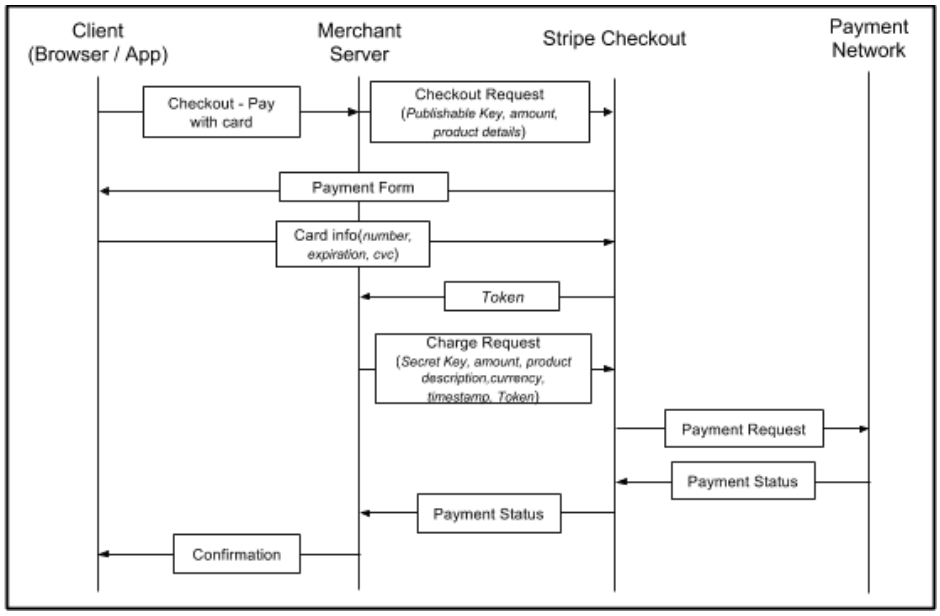

Fig. 2.7: Executing a payment with Stripe.

Stripe succeeds at providing security services such as:

[5]Stripe official products https://stripe.com

- **Authentication:** Stripe API and payment services are available only to registered merchants who receive a pair or keys during their registration: *Publishable Key* for identification and a *Secret Key* for transactions.

- **Authorization:** Stripe accepts requests from servers with a valid publishable API key. Charge requests are processed only if a valid merchant sends a valid secret key as part of the request.

- **Confidentiality:** Data sent to and from Stripe servers traverses the internet encrypted using secure HTTPS connections. Sensitive information provided by the consumer is shared only once to Stripe, by using data tokenization. Any sensitive information is protected from being sent back and forth in the messages required to complete payments.

- **Compliance:** By integrating the checkout form, merchants ensure they will comply with regulatory such as PCI because they are not handling financial information directly, instead, they transfer the responsibility to Stripe, additionally the logs of each transaction are stored for audit purposes. Integrity: Stripe procures the integrity of the information by using tokenization and secure channels to transmit payment information preventing unauthorized changes during transmission.

PayPal

Another payment company whose main payment processing products include: Express Checkout, an extension for the merchant web application to enable PayPal as a payment method; Payments Standard, a checkout page to carry out the complete payment process; Payments Pro, a customizable solution to process payments by using the payment gateway PayFlow Gateway and PayPal Here, a solution that facilitates in-store payments using a mobile device with NFC technology [6].

The common transaction flow to complete a payment using PayPal Payments Pro and PayFlow gateway is depicted in figure 2.8.

[6]PayPal available products for merchants https://www.paypal.com/fi/webapps/mpp/merchant

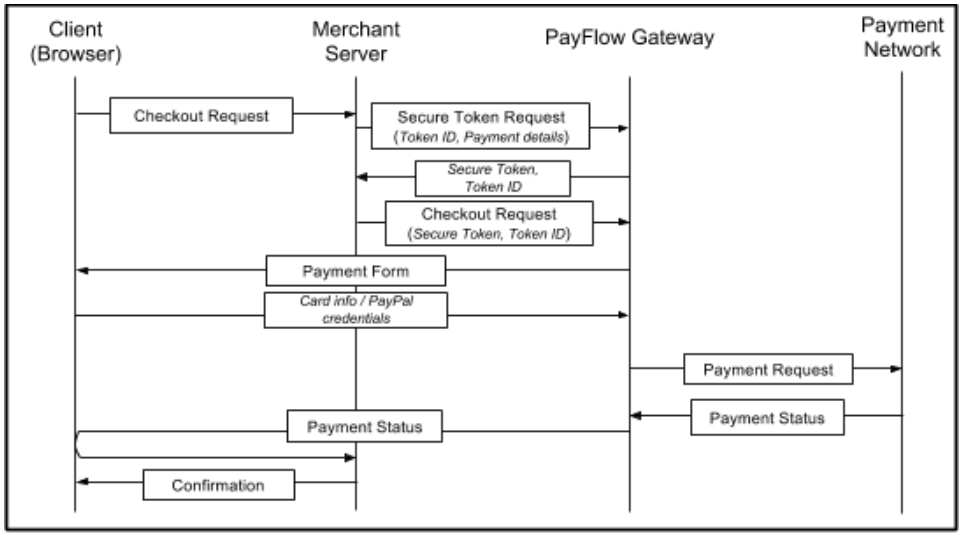

Fig. 2.8: Executing a payment with PayPal.

The consumer clicks buy on the merchant web page, the merchant server requests a *secure token* (up to 32 characters) to the gateway by passing a *token ID* (36 characters). The gateway returns to the merchant server the *secure token* created and the *token ID*. With both tokens the merchant server makes an HTTP post to the checkout page (hosted in the gateway) and redirects the client browser to the checkout page. The gateway retrieves the transaction details from the *secure token* and request card or account credentials to the consumer. The gateway processes the payments and returns the client browser to the merchant page and sends the status of the payment to the merchant. To secure the payment process, PayPal enforces:

- **Authentication:** PayPal provides to its registered merchants a unique *token ID* to post HTTP requests to the gateway and a user and password to access the gateway API.
- **Authorization:** PayPal limits the transaction capabilities per the type of account held by the merchants.
- **Confidentiality:** Data sent to and from PayPal gateway traverses the internet encrypted using secure HTTPS connections.
- **Integrity:** PayPal protects transaction data such as amount, type and currency by using tokens to present that information to the consumers.
- **Compliance:** By integrating the checkout pages hosted by PayPal, merchants ensure they will comply with regulatory requirements such

as PCI, similarly to Stripe, PayPal acquires the responsibility by handling sensitive data themselves and preventing merchants from collecting such data. Additionally, PayPal stores logs of each transaction for audit purposes.

Apple Pay

A mobile wallet available in Apple products [7], the wallet is loaded with credit and debit cards and then used for in-store and in-app purchases. For in-store purchases the NFC technology is employed. Therefore, stores accepting this payment method need a contactless point-of-sales.

To activate the wallet, the user must enter the card information which is immediately ciphered and sent to the Apple servers. In the servers, Apple deciphers the data to determine the payment network. Then, data is ciphered again and sent to the corresponding payment network with a key that only the authorized networks know. Apple sends additional data directly to the issuer bank including the iTunes account [8] activity and the device details such as phone number, model, location and time it is motion. The bank returns to Apple a *Device-specific Account Number* in addition to a key used to generate unique security codes per transaction. Finally, Apple adds the received data to the *Secure Element* (i.e., a certified chip) of the device that stores information isolated from the operating system.

After the wallet is activated, to execute a payment from a merchant web page or mobile application using Apple Pay, it is required for the merchant to own a *Payment Processing certificate* which contains a key to encrypt and decrypt data. The application sends a payment request along with the payment details to Apple Pay. Apple Play receives the request and asks the consumer for authorization. After receiving the authorization, on the device, Apple Pay sends the payment requests to the *Secure Element* where payment data, specified card and merchant ID are tokenized altogether. The token created is sent from the *Secure Element* to Apple Pay to be forwarded to an Apple server. In the server, Apple encrypts the token using the key in the *Payment Processing certificate* and signs it. To conclude with the process, the encrypted and signed token is sent back to the device where the application can decipher the token and send it directly to the payment network or a payment gateway. The entire process when paying with Apple Pay is shown in figure 2.9.

[7] List of devices that support Apple Pay https://support.apple.com/it-it/KM207105

[8] iTunes account: User account held in the Apple's iTunes Store

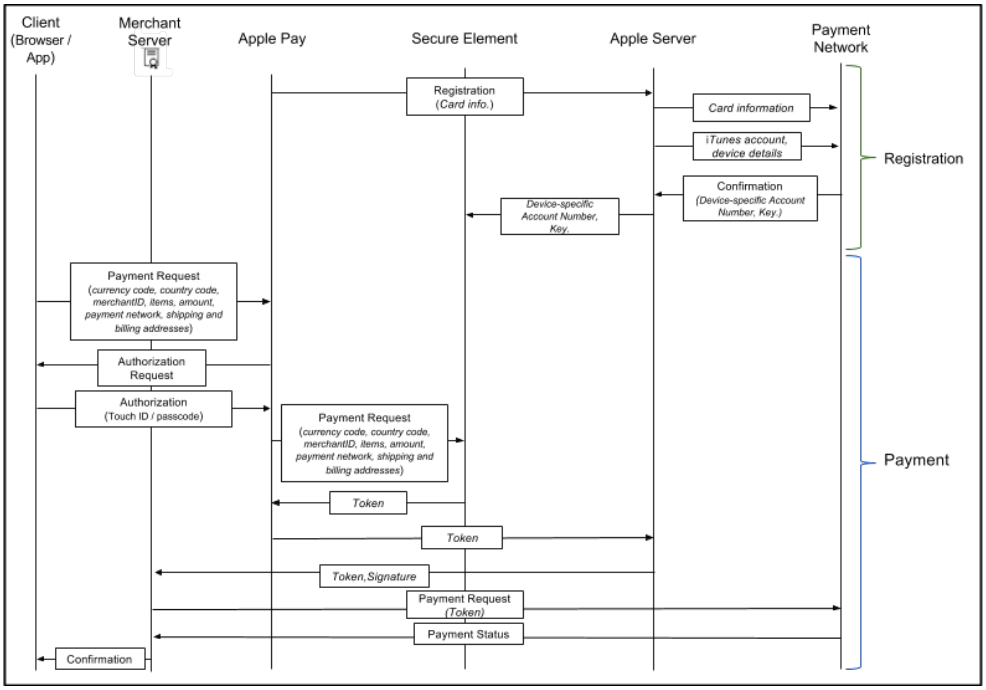

Fig. 2.9: Executing a payment with Apple Pay.

Apple Pay managed to provide security services including:

- **Authentication:** Apple Pay identifies and authenticates each merchant with a proprietary Payment Processing Certificate; for its users, an iTunes Account serves as the identification while the authentication is provisioned with a passcode or using Touch ID (Apple proprietary fingerprint sensing system [15]). After authorization is received, the payment information is encrypted, when is outside the secure element
- **Authorization:** Each transaction completed with Apple Pay requires the consumer to validate and authorize it with Touch ID or a passcode.
- **Confidentiality:** The Information transmitted between the different parties involved in the process of registration and payment is ciphered before being sent. Apple Pay compels merchants to have servers supporting TLS and serving content over HTTPS. Tokenization is used to avoid the necessity of storing sensitive data in the mobile devices or Apple server.
- **Integrity:** In addition to the secure channels employed to transmit data, which protects it from being tampered as it traverses the internet, Apple Pay utilizes a built-in secure element to store sensitive

information. The access to such information is protected using elliptic curve cryptography to prevent unauthorized access or modifications. When data leaves the secure element, it is already tokenized.

Chapter 3

Security Analysis

This chapter describes the security analysis of two mobile payment solutions offering direct carrier billing as payment method.

Case studies

We conducted an investigation to define the subjects of study and decide the scope of the security analysis presented in this thesis. We found that, in Finland, a relatively new brand for mobile payments named *Mobiilimaksu* has been adopted to allow consumers to pay for services through direct carrier billing. The payment method is accepted across merchants of different sectors, including charity, advertisement, public transportation and media content. *Mobiilimaksu* is currently available for clients of the major Finnish mobile operators [1] including Sonera[2], Elisa[3] and DNA[4]. *Mobiilimaksu* was introduced as a "self-execution payment transaction that does not require separate registration or sign-up" [5]. The simplicity of *Mobiilimaksu* to enable direct carrier billing, in conjunction with the growth of this mobile payment method in the world, make it an interesting subject of study for this thesis.

To obtain conclusive results, two merchants accepting the *Mobiilimaksu* payment method were selected to be the case studies to analyse this mobile payment method. The following criteria was considered to select the merchants for the case studies:

[1]Market shares of mobile subscriptions, Report by the Finnish Communications Regulatory Authority: www.viestintavirasto.fi/en/statisticsandreports/statistics/2013/marketsharesofmobilesubscriptions.html

[2]Sonera webpage: www.telia.fi

[3]Elisa webpage: elisa.fi

[4]DNA webpage: www.dna.fi

[5]Mobiilimaksu web page: www.mobiilimaksuinfo.fi

- Offer services for small prices.
- The services remain intangible (i.e., no physical product is received as part of the service)
- No harm caused to additional parties.
- Purchases are easily reproducible.
- Analysing the payment transaction do not require special equipment.

The merchants chosen were Tori, a web-based marketplace offering advertising services, and PayiQ Tickets, a mobile application to acquire tickets for public transportation, both accept the *Mobiilimkasu* payment method to pay for their services. The remainder of the chapter will present the security analysis of the two case studies chosen. The SANS institute proposes threat modelling as a process to ensure application security during the design of any given application. Testing the security of the application is part of the threat modelling as vulnerabilities and threats are identified [3]. The subjects of study chosen are already deployed and commercially available. Thus, the process cannot be applied as described. However, the fundamentals of the threat modelling will serve as guidance to perform the security analysis.

Characterizing the Systems

As an external entity to the development and deployment teams of the systems in study, the first step was to characterize the systems by searching for available information and acting as a legitimate user. Because of the little (publicly available) documentation about the systems studied, it was necessary to apply reverse engineering to gather sufficient information to characterize the systems. This activity is not necessary for a person who has full access to the technical specifications as developers do and it poses the risk of having an incomplete characterization of the systems. Nonetheless, the reverse engineering work made in this thesis lead to definite results for the security analysis.

The characterization of each system comprises a brief description of the system, the services it offers, the process to acquire a service or electronic product, and a detailed description of the protocol observed at the time of the payment transaction.

Tori market place with Securycast payments

Tori.fi is a Finnish online service that allows people to advertise goods. To publish a product, the users create an advertisement containing the details of the product on sale. Tori offers upgrades to promote an advertisement. Those upgrades can be bought for one day, one week or another available scheme. When users want to upgrade an advertisement, they have the possibility to pay by direct carrier billing if connected through to a MNO network.

To obtain detailed information about the process implemented by Tori to provide its services and to offer *Mobiilimaksu* as a payment method, several tests were carried out using a Lenovo notebook with Linux O.S. and Firefox browser. A Nexus 9 tablet connected to the network of Finnish MNO Elisa served as the Access Point.

Process

This section describes the process to request, pay and receive an upgrade from Tori and the protocol observed within the process.

User selects the upgrade desired for a specific advertisement (Fig. 3.1), then proceeds to checkout to select the payment method (Fig. 3.2).

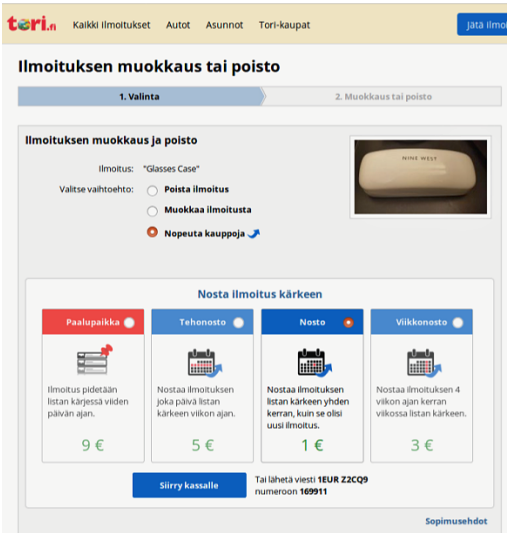

Fig. 3.1: Upgrades available.

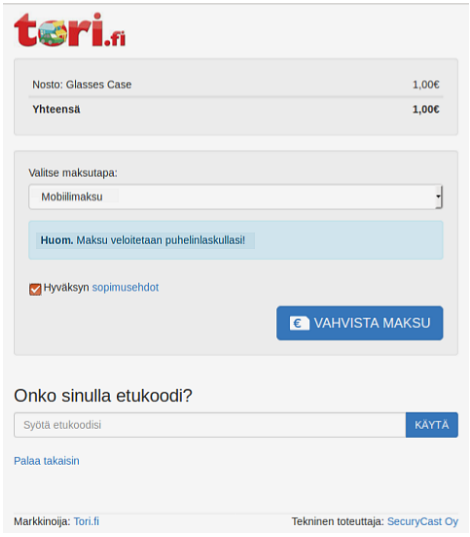

Fig. 3.2: Webcart form.

After accepting the terms and conditions, the user clicks on "Vahvista Maksu" (Confirm Payment in Finnish). If the user has sufficient funds in the mobile account (limited to 150€ per month), the payment is accepted and a confirmation including a link to download the receipt is received (Fig. 3.3).

Additionally, the user receives a confirmation email, later, the advertisement is upgraded accordingly (Fig. 3.4).

Fig. 3.3: Confirmation of payment.

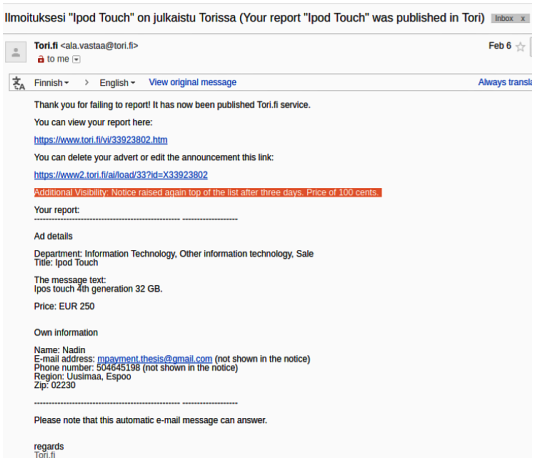

Fig. 3.4: Confirmation mail.

Protocol

To investigate further about the parties involved in the process, the traffic between the client and the web server was analysed using Wireshark. After the analysis, it was observed that most traffic is carried out through TLS. Little plaintext HTTP traffic was captured, leaving the analysis inconclusive. Therefore, a type of Man in the Middle technique [6] had to be used at this point of the analysis.

The Protocol described below is the result of the analysis of all the TLS traffic captured with the SSL proxy between the browser and the Tori web page at the time of the payment transaction. Figure 3.5 depicts the different parties involved and the steps carried out to complete the payment process.

[6]TLS Proxy: A technique where a proxy is configured in the browser to ciphered and deciphered the TLS traffic between the browser and the web servers, acting as a Man in the Middle.

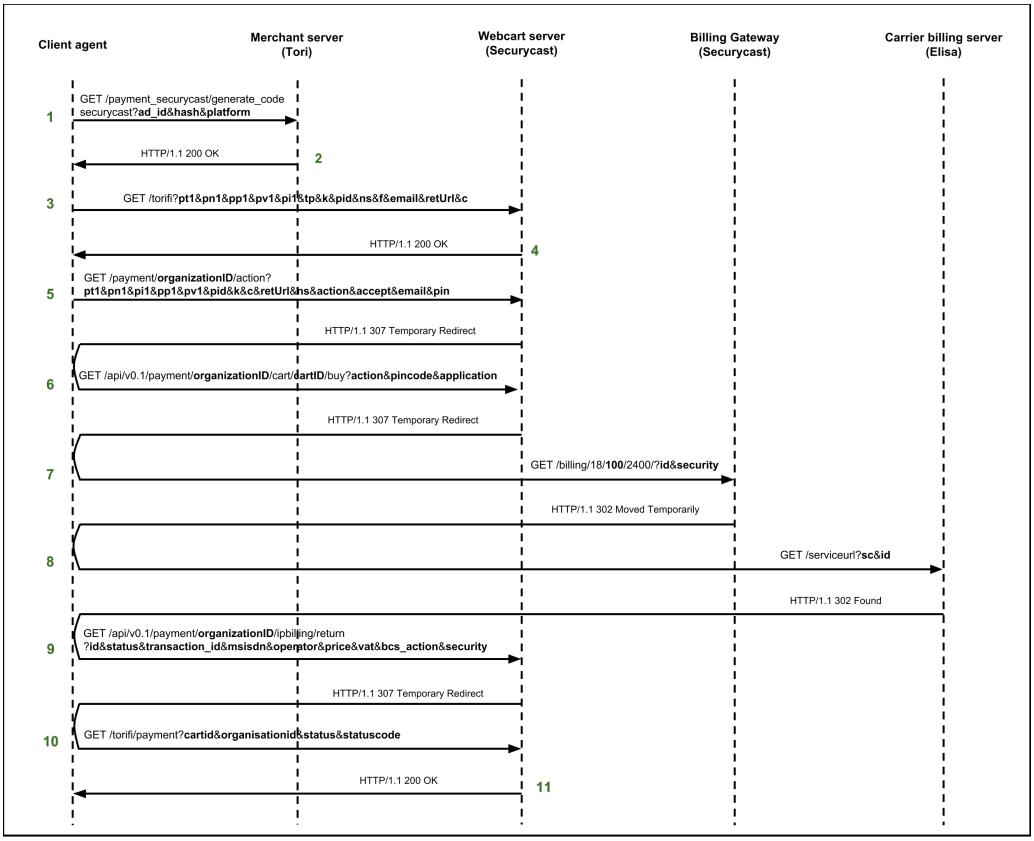

Fig. 3.5: Steps of the payment process.

Parties Involved

Below is the list of parties observed during the payment process.

- **Client:** The web browser used by any user to access services from Tori.
- **Tori:** Web-based marketplace.
- **Securycast:** Mobile Payment Service Provider. Handles payments between the merchant and the mobile network operator.
- **Elisa:** Mobile Network Operator, providing the client with internet connection and the *Mobiilimaksu* payment method.

Transaction Flow The steps to request and pay for a service at Tori page are listed below, starting from the moment the consumer has posted an ad and is offered to upgrade it, to the moment he receives a confirmation of the payment.

1. The consumer, connected to Tori page via his mobile network, clicks to see the available upgrades for his ad. The browser sends the request to Tori server, passing three parameters through a GET request:
   - *Advertisement for which the upgrade is being requested (ad_id)*
   - *40-character hash (hash)*
   - *Client platform (platform)*
2. Tori server generates the links to the Securycast webcart server for the different services (i.e., upgrades) available, including the following parameters.
   - *Product type (pt1)*
   - *Product name (pn1)*
   - *Product price (pp1)*
   - *VAT (pv1)*
   - *Ad ID (pi1)*
   - *Total price (tp)*
   - *Price and currency (k)*
   - *Product ID (pid)*
   - *Payment Methods (f)*
   - *Return URL (retUrl)*
   - *128-character checksum (c)*

   The server responses to the request and the browser displays the available services.
3. User selects the service and clicks to proceed to checkout. The browser is redirected to the webcart server of the payment service provider, Securycast, with the parameters defined in step 2.
4. Securycast responds with a form including all available payment methods defined by the merchant.
5. User confirms the *Mobiilimaksu* payment method. A script in the webcart server generates a Cart ID for the organization (Tori). A GET request is sent to the webcart server with the parameters:
   - *Product type (pt1)*

- *Product name (pn1)*
- *Advertisement ID (pi1)*
- *Product price (pp1)*
- *VAT (pv1)*
- *Product ID (pid)*
- *Price and currency (k)*
- *Total price(tp)*
- *128-character checksum (c)*
- *Return URL (retUrl)*
- *Payment Method selected (action)*
- *Indicator of purchase acceptance (accept)*
- *Email (email). Empty parameter.*
- *PIN code (pin). Empty parameter.*

6. Client is redirected to the payment platform server for Tori and the specific Cart ID with a GET request containing the parameters:
   - *Payment method selected (action)*
   - *PIN code (pincode). Empty parameter.*
   - *Application from which the request is redirected (application=WEBCART)*
7. Client is redirected to the mobile billing gateway server of Securycast with the parameters:
   - *Cart ID (id)*
   - *40-character hash (security)*
8. Client is redirecting to the Elisa billing server with the parameters:
   - *Source Code(sc)*
   - *Transaction ID (id)*
9. Client is redirected back to the payment platform server with the details of the transaction:
   - *Cart ID (id)*
   - *Status of transaction (status)*

- *Transaction ID (transaction_id)*
- *Phone number (msisdn)*
- *Mobile Operator ID(operator)*
- *Price in cents (price)*
- *VAT(vat)*
- *Billing action (bcs_action)*
- *40-character hash (security)*

10. Client is redirected back to the webcart server with the status of the transaction and additional parameters for reference.
    - *Cart ID (cartid)*
    - *Tori ID (organisationid)*
    - *Status of transaction (status)*
    - *Status code of transaction (statuscode)*
11. From the webcart server, the client receives the confirmation message for the transaction if the process has completed successfully, or otherwise, an error message.

Additional parties and steps may be present at the time of the payment processing; however, those remained invisible to the client band and reverse engineering. It is not possible to describe, in detail, what communication exists between the merchant, the payment service provider and the operator. It can be assumed that at least the payment service provider and operator exchange information to create and process valid transactions.

PayiQ Mobile Ticketing

PayiQ is a Finnish company offering mobile ticketing solutions. The mobile application PayiQ Tickets, developed by PayiQ, is available to end users to buy mobile tickets. To study the way PayiQ provides mobile ticketing services and offers *Mobiilimaksu* as a payment method, the application was tested on a Nexus 5 phone connected to the Elisa network.

Process

To use the application, the user must register by providing personal information including name, family name, phone number and e-mail. Once registered, the user accesses the application and creates a PIN code (4 digits), which is thereafter used to authorise purchases. Subsequently, the user accesses the ticket shop. The types of tickets available to purchase depend on the location. In Finland, the current available tickets are for public transportation.

To buy a ticket, the user selects the city (Fig. 3.6), the ticket type (Fig. 3.7) and the zone (Fig. 3.8).

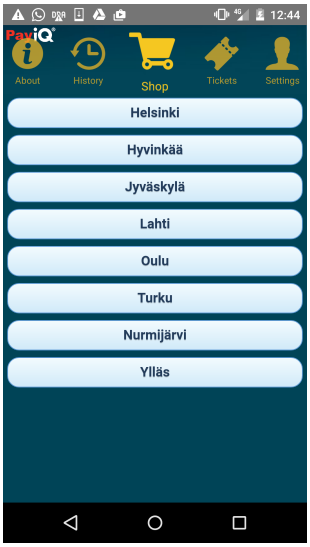

Fig. 3.6: Cities.

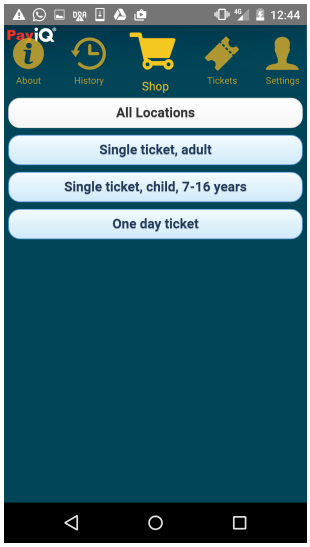

Fig. 3.7: Ticket type.

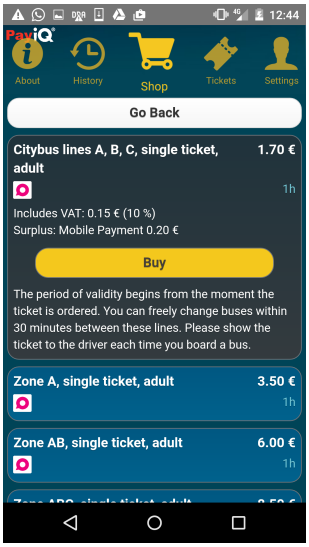

Fig. 3.8: Zones available.

By clicking the "Buy" button, the user confirms the purchase and is presented with a screen to select the payment method and to enter the PIN code to authorise the payment (Fig. 3.9).

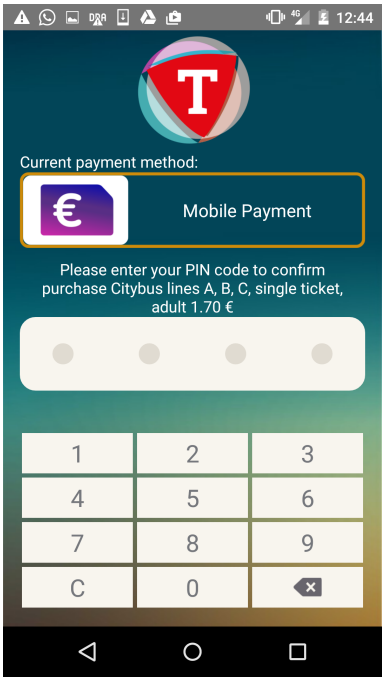

Fig. 3.9: Authorising the payment.

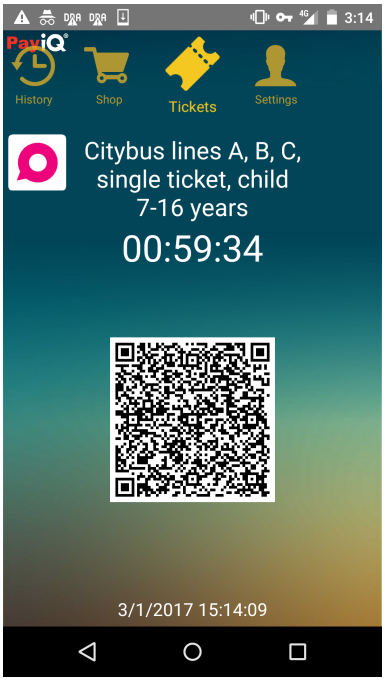

Fig. 3.10: Digital ticket.

If the PIN is correct, the order is processed. If the payment is successful, the user receives the mobile ticket in the form of QR code (Fig. 3.10).

During the analysis, it was observed that the application works only when the mobile device is connected to a MNO network. If a Wi-Fi connection is active, a message prompts the user to connect to a mobile network.

Protocol

To understand the entire protocol carried out during the ticket purchase, the traffic between the application and servers was recorded using a TLS proxy. The Protocol described below is the result of the analysis of the traffic captured. The diagram in Fig. 3.11 shows the parties involved and the steps executed to complete the ticket purchase.

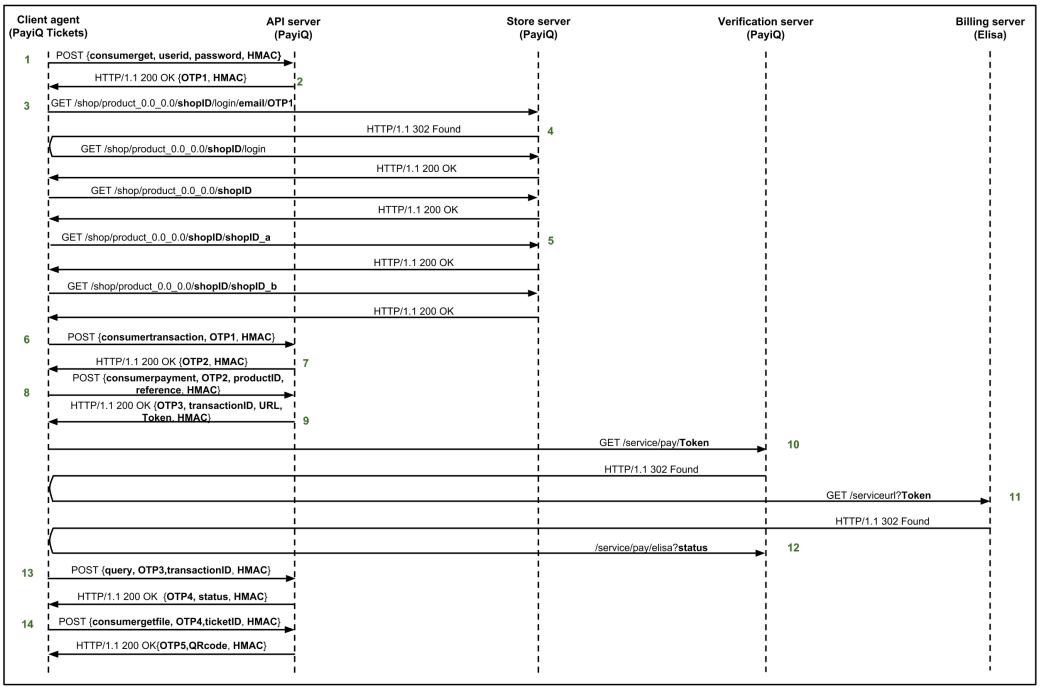

Fig. 3.11: Steps of the payment process.

Parties Involved

Below is the list of parties observed during the acquisition and payment of tickets in the PayiQ Tickets application.

- **Client:** The mobile application used to acquire a ticket for public transportation.
- **PayiQ:** Merchant, providing mobile tickets through an HTTP API.
- **Elisa:** Mobile Network Operator, providing access to internet and *Mobiilimaksu* payment method.

Transaction Flow

The steps to acquire tickets from the application are listed below, starting from the moment the consumer logs in the application to the moment he receives the electronic ticket.

1. The user enters his credentials in the application, the application sends a request *consumerget* to the PayiQ API server to retrieve information about the user. The parameters sent in the request include:
   - *Shop ID (shop)*

   - *User name (login_identity)*
   - *User password (login_password)*
   - *Device details (identity_plugindata)*
   - *String of 40 random characters (random)*
   - *Version (version)*
   - *Request type (command)*
   - *128-character HMAC (hash)*

2. If the credentials are correct, PayiQ API server returns the user information in addition to a one-time password (40-character OTP). In the application, the user is asked to create a PIN code to authorise payments.

3. The application sends a request to the PayiQ Consumer server to grant access to the shop, passing along the OTP and additional parameters:

   - *Shop ID*
   - *Email*
   - *OTP*

4. If the OTP is correct, the access request is successful and the PayiQ Consumer server retrieves the list of available cities to purchase transportation tickets.

5. The user selects the city, type and zone of the ticket desired and finally clicks the "Buy" button in the application.

6. The application sends a *consumertransaction* request to the PayiQ API server with the OTP received in the first request and the additional parameters:

   - *Shop ID (shop)*
   - *User name (login_identity)*
   - *OTP (login_password)*
   - *Location of the device (location)*
   - *Device details (identity_plugindata)*
   - *String of 40 random characters (random)*
   - *Version (version)*

- *Request type (command)*
- *128-character HMAC (hash)*

7. The PayiQ API server returns the transaction history of the user and a new OTP as part of the response.

8. The application sends a *consumerpayment* request to the PayiQ API server with the latest OTP received and the parameters:

   - *Shop ID (shop)*
   - *User name (login_identity)*
   - *OTP (login_password)*
   - *Location of the device (location)*
   - *Device details (identity_plugindata)*
   - *Type of account (account_type)*
   - *String of 40 random characters (random)*
   - *Reference number (reference)*
   - *Account number (account_number)*
   - *Product details (id,count)*
   - *Version (version)*
   - *Request type (command)*
   - *128-character HMAC (hash)*

9. The PayiQ API server returns the information to process the payment and a new OTP:

   - *128 Hex-character HMAC (hash)*
   - *Transaction ID (id)*
   - *OTP*
   - *Action Pending (pending: operator verify)*
   - *String of 40 random characters (random)*
   - *Reference number (reference)*
   - *Status of the transaction (status)*
   - *Timestamp*
   - *URL to the verification server (url/token)*

The token is a 40-character string, unique for each transaction to verify the transaction.

10. The application redirects to the URL received in previous request (verification server).

11. The verification server redirects to Elisa server with the verification code (token).

12. Elisa server returns the status of the transaction back to the verification server.

13. The application sends a *query* request to the PayiQ API server with the OTP received in the last request, the transaction ID and additional parameters:

    - *Shop ID (shop)*
    - *User name (login_identity)*
    - *OTP (login_password)*
    - *Device details (identity_plugindata)*
    - *Transaction ID (id)*
    - *String of 40 random characters (random)*
    - *Version (version)*
    - *Request type (command)*
    - *128-character HMAC (hash)*

    In return, the API server sends the status of the transaction and a new OTP.

14. If the transaction was accepted, the application sends a *consumergetfile* request to the API server with the OTP received in the previous requests, and the ID of the public transport provider for whom the ticket is bought for. In return, a QR code is received to serve as the electronic ticket, completing with this the procedure.

Like the Tori and Securycast case study, additional parties and steps may participate in the payment transaction. Since those remained invisible to the client, it is not possible for us to describe what communication exists between the actual merchants (i.e., the public transport providers), PayiQ and the operator. It is assumed that PayiQ and the operator exchange information to create and processed valid transactions.

Adversary Model

With the systems characterized, it is possible identify the threats and vulnerabilities. As Burns from the SANS Institute states in his paper [3], before proceeding with the threat identification, it is convenient to analyse the system as an adversary. Therefore, it is appropriate to define attack scenarios in the context of computer security and to reason about the possible motives behind malicious parties, i.e., adversaries, to act against mobile payment systems.

According to the Oxford English Dictionary, an attack is "an attempt to disrupt a computer system, network, etc., by gaining unauthorised access or control" [29]. In the computer security field [18][10][1], additional factors are considered to define and classify attacks such as the tools utilize for its completion, the vulnerabilities exploited and the impact on the target. The Adversaries, for their part, may be classified based on different criteria:

- **Location:** An adversary can be an insider or outsider, i.e., an entity known and authorised to participate in the protocol or an unidentified external entity with no explicit rights to take part in the process.
- **Organization:** The adversary may be a single entity or a group of coordinated entities.
- **Level of knowledge and resources:** Depending on the level of knowledge and the resources, in terms of funding and tools; the adversary may be an expert with sophisticated and specialized resources or a neophyte with access only to commercial or free tools.
- **Activeness:** The adversaries may vary in how actively they interfere with the target system. An active advisory, concerns about reading, modifying and injecting data. On the other hand, a passive advisory only concerns about eavesdropping the communication channels to gather information without causing alterations in the system, process or data.

Regardless of the type of adversary, to plot an attack and to succeed, it is necessary to identify the attack surface, i.e., all the elements in the system (e.g., individuals, organizations, hardware, software and communication channels), establish the goals and targets, and to recognize potential weaknesses that enable the attack.

In the systems studied in this thesis the main goals for any party to act maliciously include:

- Receive the service without being charged by the carrier.
- Receive a payment without providing any product or service.
- Excessive charging for the service provided.
- Obtain sensitive information.

The targets may be any of the participant entities listed for each system. However, for this thesis, the MNO network was left out of scope as an attacker.

Identifying the Threats

The next phase for the security analysis is to identify the threats by analysing the flow of the data throughout the system and to locate entry and exit points where critical security processes occur. Different approaches and models exist to assist in the identification of threats [25]. The NIST Institute suggest the use of the STRIDE [7] model. The model is designed to identify the types of attacks an application is vulnerable to. This is accomplished by analysing the properties opposite to the ones desired in a system.

The table 3.1 summarizes the list of threats identified after applying the STRIDE model to the system comprised by Tori and Securycast. The same is depicted in table 3.2 for the PayiQ ticketing application. The column *Threat Identifier* shows the ID to identify the threat, the column *Threat* names the threat identified while the column *Property Violated* lists which security requirement is compromised by the threat. The column *Path*, describes the possible path to find the threat and discover if the systems are vulnerable to certain threat.

[7] STRIDE, an acronym derived from six threat categories: **S**poofing identity, **T**ampering with data, **R**epudiation, **I**nformation disclosure, **D**enial of service, **E**levation of privilege

Table 3.1: STRIDE model applied to the Tori-Securycast System

Threat Identifier	**Threat**	**Property Violated**	**Path**
TS.t01	Spoofing the consumer.	Authentication	Impersonate someone by using their mobile network.
TS.t02	Spoofing the client (browser agent)	Authentication	Fake a mobile browser agent. Configure a SSL proxy.
TS.t03	Spoofing Tori server.	Authentication	Mock the Tori web page.
TS.t04	Spoofing the payment gateway.	Authentication	Impersonate the gateway server.
TS.t05	Spoofing the MNO billing server.	Authentication	Impersonate the billing server.
TS.t06	Spoofing the transaction.	Authentication	Reutilize an authentic transaction. Trigger unauthorised transactions.
TS.t07	Spoofing the messages.	Authentication Freshness	Reuse old messages.
TS.t08	Tampering the Payment data sent from Tori.	Integrity	Modify the price set by Tori.
TS.t09	Tampering the payment data.	Integrity	Modify the price or account details in the requests processed by the Securycast server.
TS.t10	Tampering the payment status.	Integrity	Modify the status of a transaction.
TS.t11	Repudiate the transactions.	Non repudiation	Request for services without signing up.
TS.t12	Information Disclosure.	Confidentiality	Intercept the communication by setting a proxy.
TS.t13	Denial of Service.	Availability	Flood the servers with fake requests. *Does not help achieve any of the main goals.
TS.t14	Elevation of Privilege.	Authorisation	Capture payments from other customers of Securycast.

Table 3.2: STRIDE model applied to the PayiQ System

Threat Identifier	**Threat**	**Property Violated**	**Path**
PT.t01	Spoofing the consumer.	Authentication	Take over a legitimate user account.
PT.t02	Spoofing the client (Application).	Authentication	Create a fictitious application. Configure a SSL proxy.
PT.t03	Spoofing the merchant server.	Authentication	Impersonate PayiQ servers.
PT.t04	Spoofing the MNO billing server.	Authentication	Impersonate the billing server.
PT.t05	Spoofing the network.	Authentication	Spoof a mobile network connection in the application.
PT.t06	Spoofing the messages.	Authentication	Reutilize authentic messages or craft new messages to simulate authentic ones.
PT.t07	Tampering the payment data.	Integrity	Modify the requests to the servers.
PT.t08	Tampering the payment status.	Integrity	Modify the status of a transaction from failed to successful.
PT.t09	Repudiate the transactions.	Non-repudiation	Provide false personal identifiable Information.
PT.t10	Information Disclosure.	Confidentiality	Intercept the communication by setting a proxy.
PT.t11	Denial of Service.	Availability	Block an account. *Does not help achieve the goals.
PT.t12	Elevation of Privilege.	Authorisation	Obtain the NIP to authorise payments.

Testing the Threats

This section presents the test scenarios that were defined to accomplish one of the goals defined in the adversary model while testing if the selected case studies are vulnerable to the threats identified in the previous section.

Tori with Securycast

Man in the Middle

TS.a01 **Goal:** Obtain information about the payment process by eavesdropping the communication between the client and the Tori.

TS.a01 **Related Threat ID:** TS.t02, TS.t12.

TS.a01 **Description:** At the consumer end, configure a TLS proxy in the web browser to intercept and decipher the communication between the browser and the merchant server.

TS.a01 **Results:** Achieved. The attack was successful, the servers did not refuse the connection, it was possible to decipher the messages sent between the client, Tori and other parties involved. Nonetheless, the attack requires explicit consent in the browser for to trust in the Man in the Middle and is mainly useful for reverse engineering.

TS.a02 **Goal:** Act as the Securycast gateway to request payments to Elisa on behalf Tori.

TS.a02 **Related Threat ID:** TS.t04, TS.t05.

TS.a02 **Description:** Redirect traffic intended to the gateway to a fake server to divert payments. Create a fake transaction ID to send a payment request to the carrier billing server.

TS.a02 **Results:** Failed. The transaction ID sent in the step 8 cannot be spoofed. The carrier billing server sends an error if an invalid ID is sent. Additionally, according to one of the Finnish MNO, the service provider should indicate its domain prior to being able to request charges [27]. Although, no evidence or documentation was found about Elisa to confirm this.

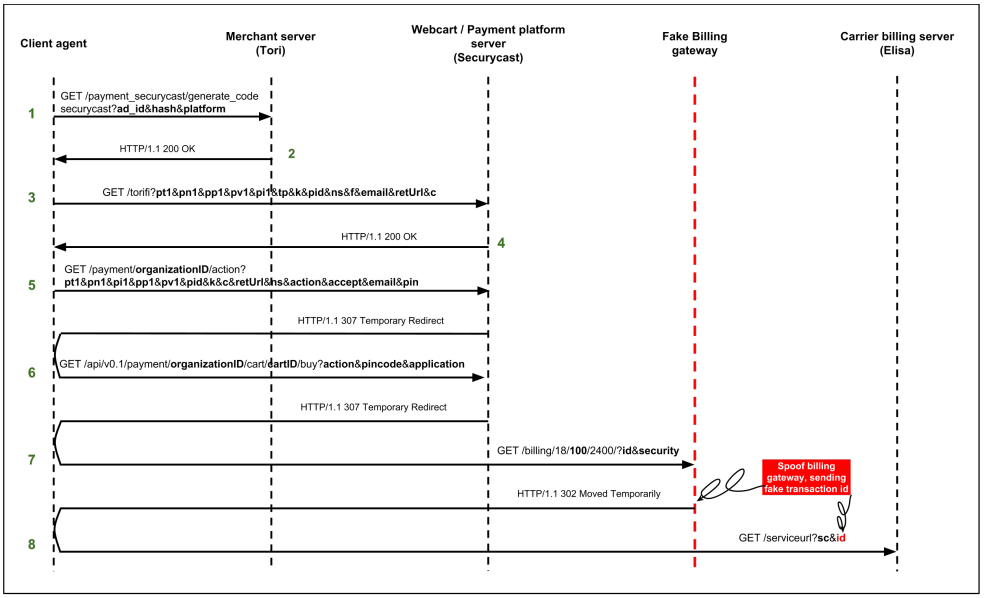

Fig. 3.12: Requesting a payment to the carrier billing server.

Impersonation

TS.a03 **Goal:** Acquire advertisement upgrades from Tori paid by someone else by impersonating a Finnish mobile operator customer to access *Mobiilimaksu* services, misusing their shared mobile Internet connection.

TS.a03 **Related Threat ID:** TS. t01.TS.t011.

TS.a03 **Description:** To impersonate a mobile subscriber, it is necessary to find an individual holding a subscription with a Finnish mobile operator. The individual should share his mobile connection. Then, in the tethered device, purchase an upgrade in the Tori page using the *Mobiilimaksu* payment method.

TS.a03 **Result:** Achieved. The mobile network operator identifies the payee automatically [8] and charges accordingly. Furthermore, the user is never prompted for a confirmation or authorization of the purchase after selecting the mobile payment option. Therefore, as malicious user, is possible to take advantage of tethering services to acquire Tori upgrades for free, as the owner of the subscription is the one billed.

[8] Elisa's notes about *Mobiilimaksu*, elisa.fi/asiakaspalvelu/aihe/matkapuhelinliittymat/ohje/mobiilimaksaminen

Replay

TS.a04 **Goal:** Receive an upgrade from Tori without effecting the payment.

TS.a04 **Related Threat ID:** TS.t05, TS.t06, TS.t07, TS.t10.

TS.a04 **Description:** Bypass the payment by avoiding the redirection to the mobile operator server, instead, redirecting the client to the Securycast server with an old successful *transactionID* for the new cart.

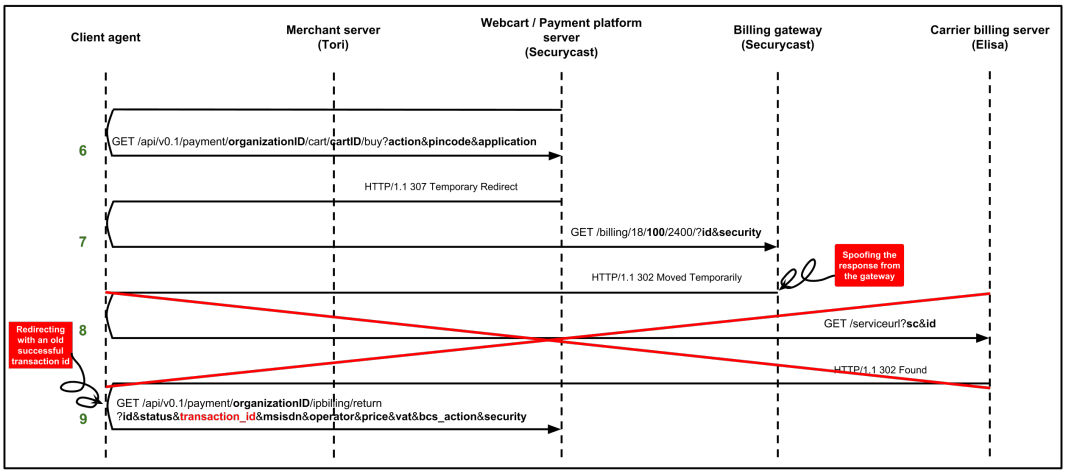

Fig. 3.13: Skipping redirection to the MNO server.

TS.a04 **Results:** Failed. A general error thrown by the Securycast server.

Forgery

TS.a05 **Goal:** Execute payment transactions without the explicit consent of the consumer.

TS.a05 **Related Threat ID:** TS.t06.

TS.a05 **Description:** The merchant adds JavaScript code in the advertisement page to trick customers. Then, when the customer clicks on a strategic place of the page, the merchant performs a cross site request forgery attack sending a payment transaction request to the webcart server, bypassing the webcart form to checkout and the confirmation from the customer. Skipping every step before step 5 of the data flow as shown in figure 3.14.

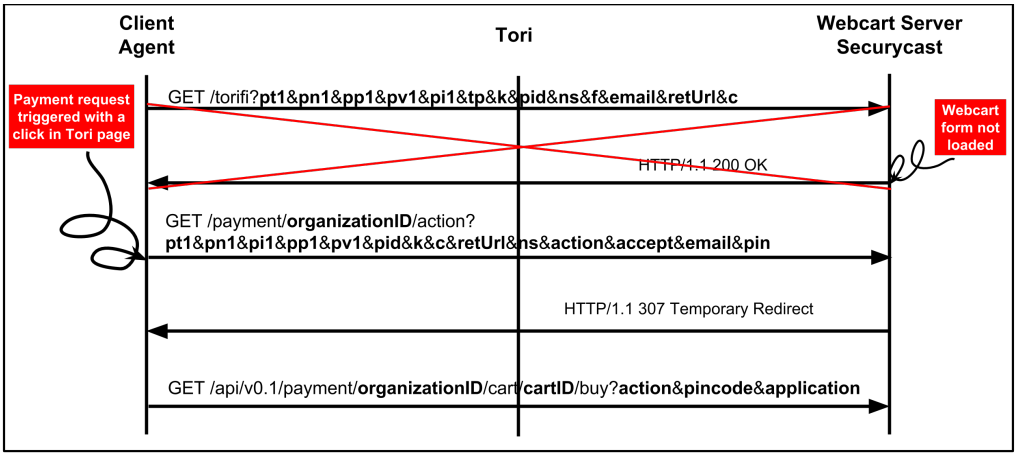

Fig. 3.14: CSRF attack possible from the merchant to the webcart server.

TS.a05 **Results:** Achieved. The attack was successful. The webcart server does not identify the request comes directly from the merchant page and not the webcart form page. The payment is completed successfully.

TS.a06 **Goal:** Receive payments on behalf the merchant by forging Tori webpage.

TS.a06 **Related Threat ID:** TS.t03.

TS.a06 **Description:** Cloning Tori webpage to appear to the consumers as the real one, offering fake services while receiving mobile payments.

TS.a06 **Results:** Failed. Tori server is authenticated with a digital certificate; thus, it is not sufficient to clone the page to deceive consumers, the forged page requires a valid certificate for consumers to trust in it. Furthermore, to process mobile payments, it will be necessary to impersonate Tori towards Securycast.

Tampering

TS.a07 **Goal:** Receive an upgrade from the Tori without effecting the payment by tampering, as a consumer, the product details set by the merchant.

TS.a07 **Related Threat ID:** TS.t02, TS.t08.

TS.a07 **Description:** When the user selects the product, he is redirected to the webcart server with the product details in the request. The attack consists of modifying the request to the webcart server and setting a

new value to the product price (pp1=0.00) to avoid paying the amount established by Tori.

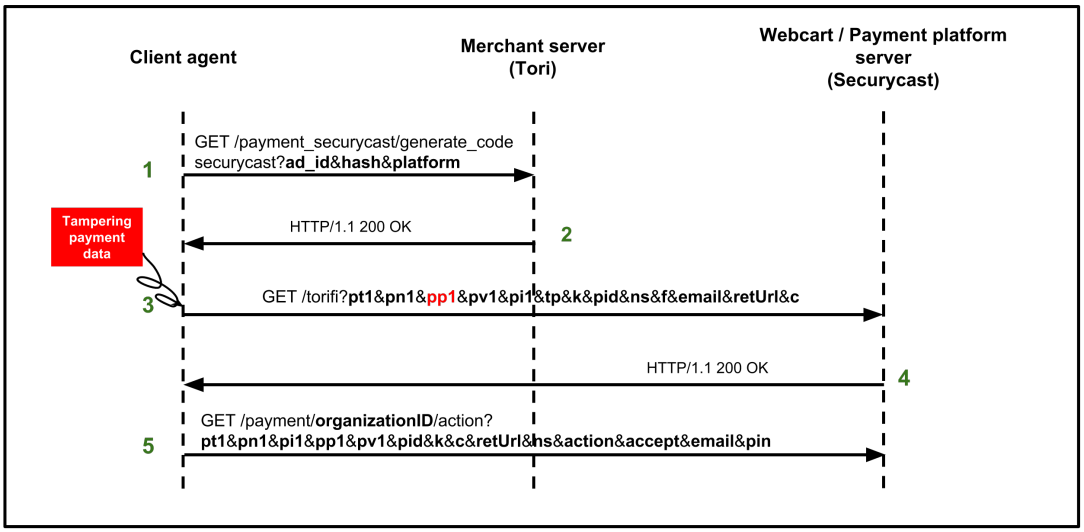

Fig. 3.15: Tampering payment details in the payment request.

TS.a07 **Results:** Failed. It was observed that the server responded with an error code and redirects the browser back to the advertisement page. It can be inferred that the webcart server checks the integrity of the data using code c

TS.a08 **Goal:** Receive an upgrade from Tori without effecting the payment by tampering the data in the webcart form received from the webcart server of the payment service provider.

TS.a08 **Related Threat ID:** TS.t02, TS.t09.

TS.a08 **Description:** Tampering the data at different point of the transaction flow, i.e., the form in the response from Securycast. Setting the value of the product price (pp1) to 0.00 before confirming the payment.

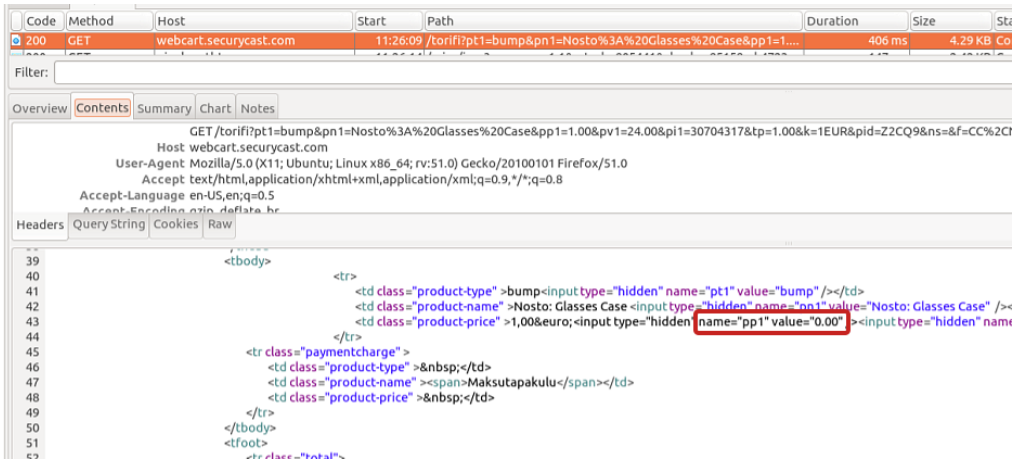

Fig. 3.16: Tampering the webcart form received from Securycast.

TS.a08 **Results:** Failed. The payment process continues after altering the form in the response from Securycast. However, Elisa rejects the transaction, redirecting the user to the webcart server with an error message.

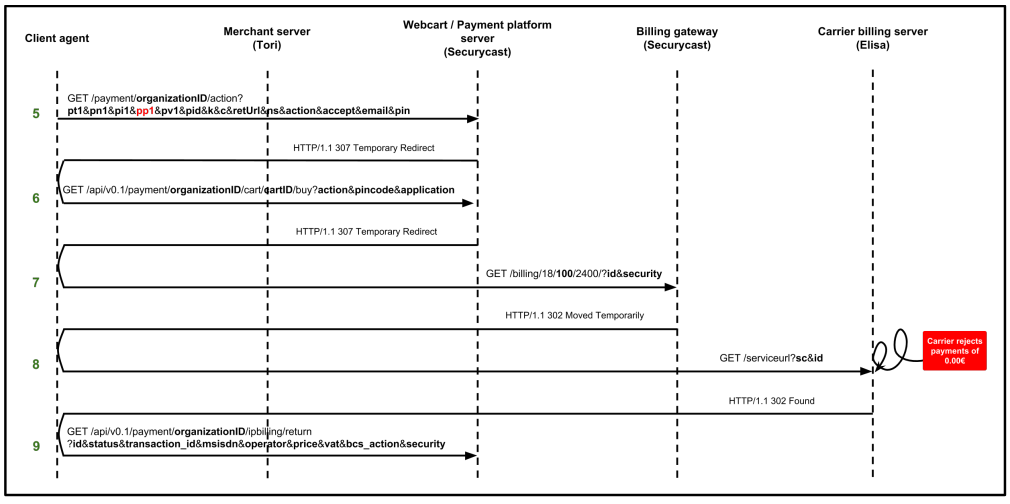

Fig. 3.17: Part of the protocol where the MNO rejects a payment for 0 €.

TS.a09 **Goal:** Receive an upgrade from Tori paying a minimum amount by tampering the response from the webcart server at the consumer end, i.e., the browser.

TS.a09 **Related Threat ID:** TS.t02,TS.t09.

TS.a09 **Description:** As a consumer, tamper the data at different point of the transaction flow, i.e., the form in the response from Securycast. Setting the value of price pp1 to 0.10 before confirming the payment.

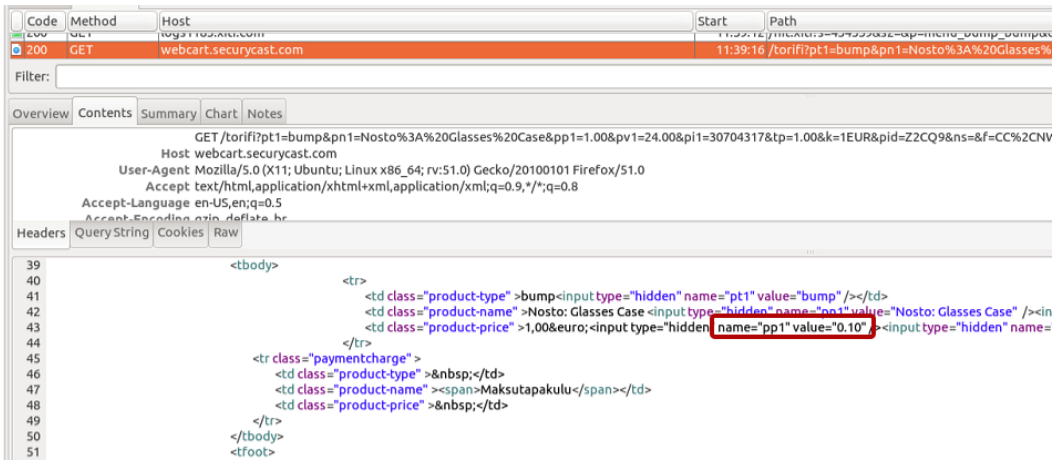

Fig. 3.18: Tampering the webcart form received from Securycast.

TS.a09 **Results:** Success. The payment process continues after altering the form in the response from Securycast. The transaction is completed and the consumer receives a receipt for the purchase.

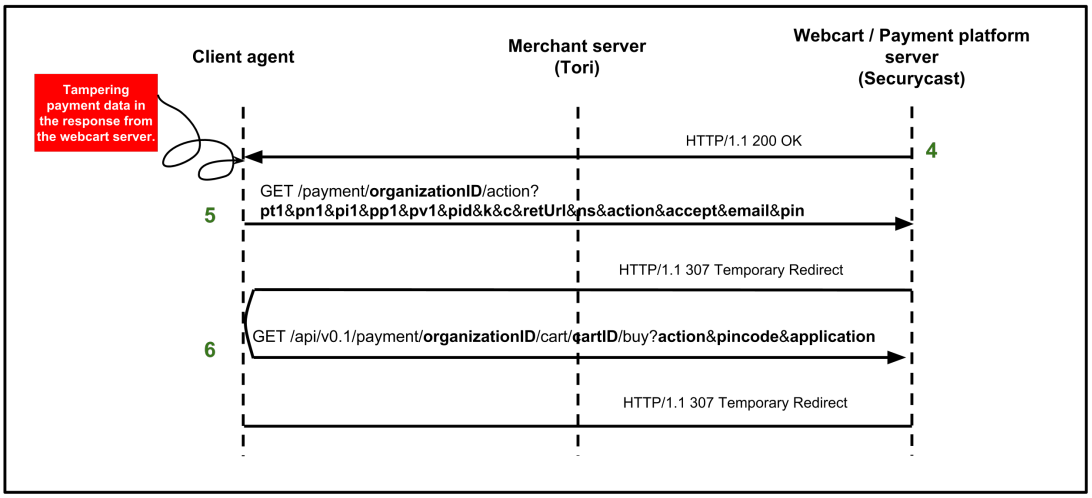

Fig. 3.19: Part of the protocol where tampering attack is successful.

TS.a10 **Goal:** Receive an upgrade from Tori without effecting the payment by tampering the requests from the browser, at the consumer end, to the servers.

TS.a10 **Related Threat ID:** TS.t02, TS.t09.

TS.a10 **Description:** Tampering the data at different point of the transaction flow, i.e., after selecting the payment method and confirm. Tampering the request to the Securycast payment server.

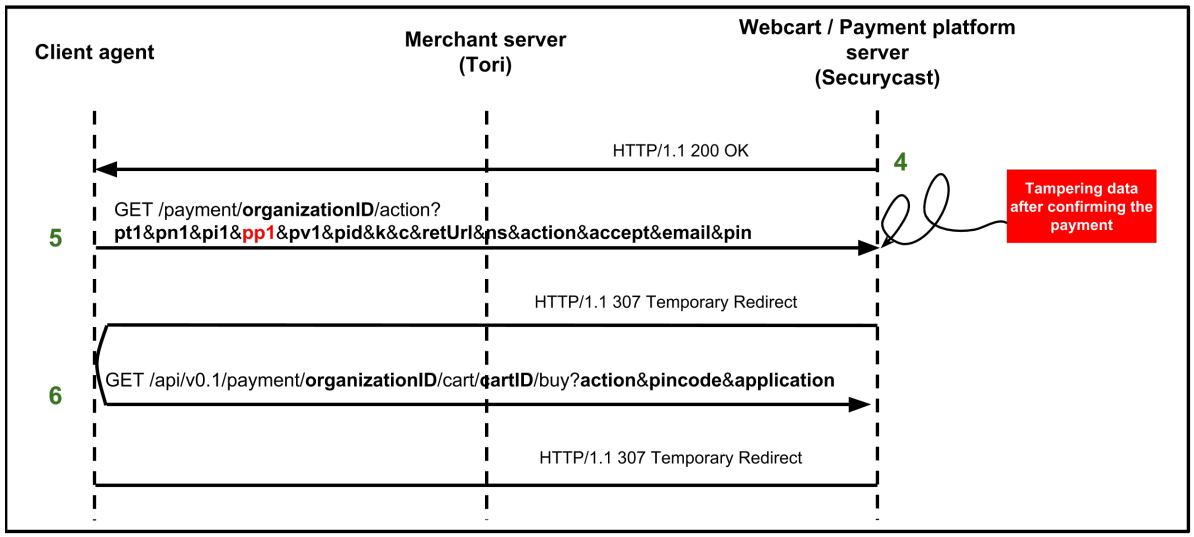

Fig. 3.20: Tampered data accepted by the webcart server.

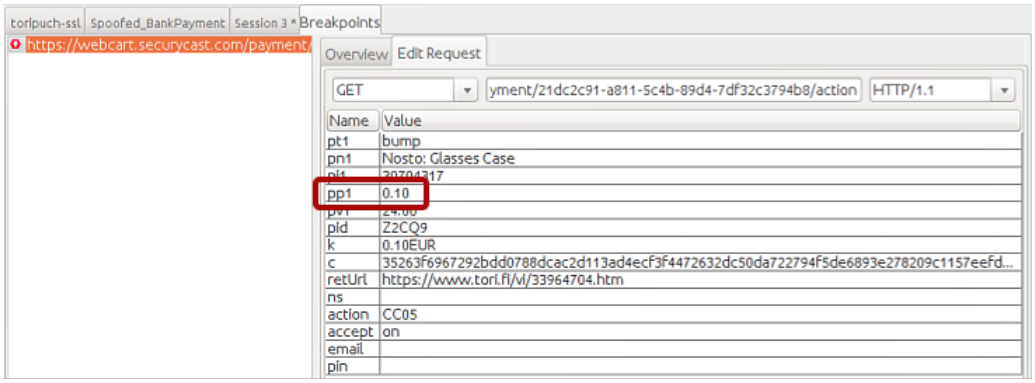

Fig. 3.21: Tampering the request to the webcart server.

TS.a10 **Results:** Failed. The payment process continues after altering the request to Securycast. Consumer receives a receipt for the purchase, however, Tori does not provide the upgrade, even though user is charged.

After detecting where in the protocol tampering the data results in a successful attack (step 5), additional attacks were conducted to validate if the same will happen with other payment methods.

TS.a11 **Goal:** Receive an upgrade from Tori paying a minimum amount with credit card by tampering the requests from the server to the browser.

TS.a11 **Related Threat ID:** TS.t02, TS.t09.

TS.a11 **Description:** Tampering the form in the response from Securycast. Setting the value of price pp1 to 0.10 and confirm the payment by credit card.

TS.a11 **Results:** Success. The payment process continues after altering the webcart form received from Securycast. The browser is redirected to the card payment processing server with the new price, transaction is completed and the customer receives a receipt for the purchase.

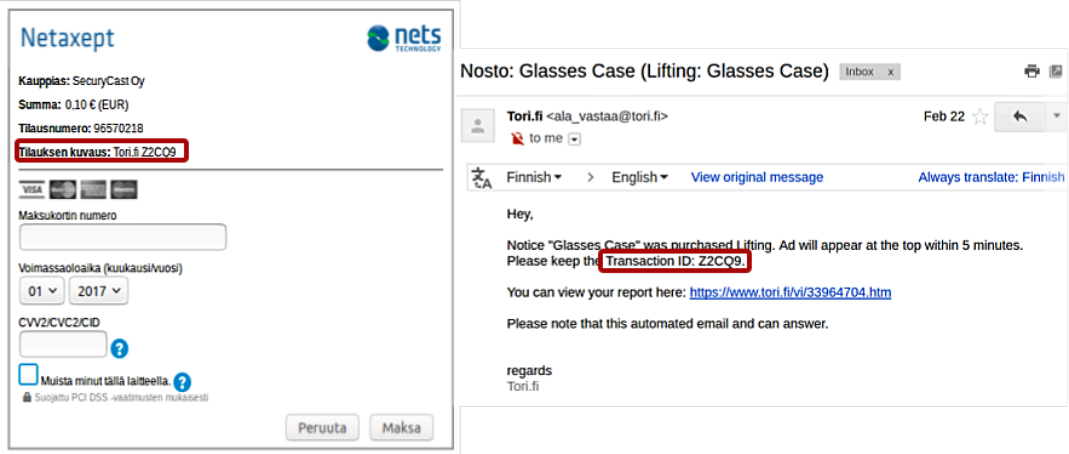

Fig. 3.22: 2€ value product acquired for 0.10 cents.

HTML Injection

TS.a12 **Goal:** Receive an upgrade from Tori paying a minimum amount by injecting HTML code.

TS.a12 **Related Threat ID:** TS.t09.

TS.a12 **Description:** With the breach in the protocol detected, utilize unsophisticated, unpaid tools to tamper the data, i.e., the web browser development tools. Injecting HTML code to replace the information in

the Webcart form and set the value of pp1 (price) to the one desired (0.10).

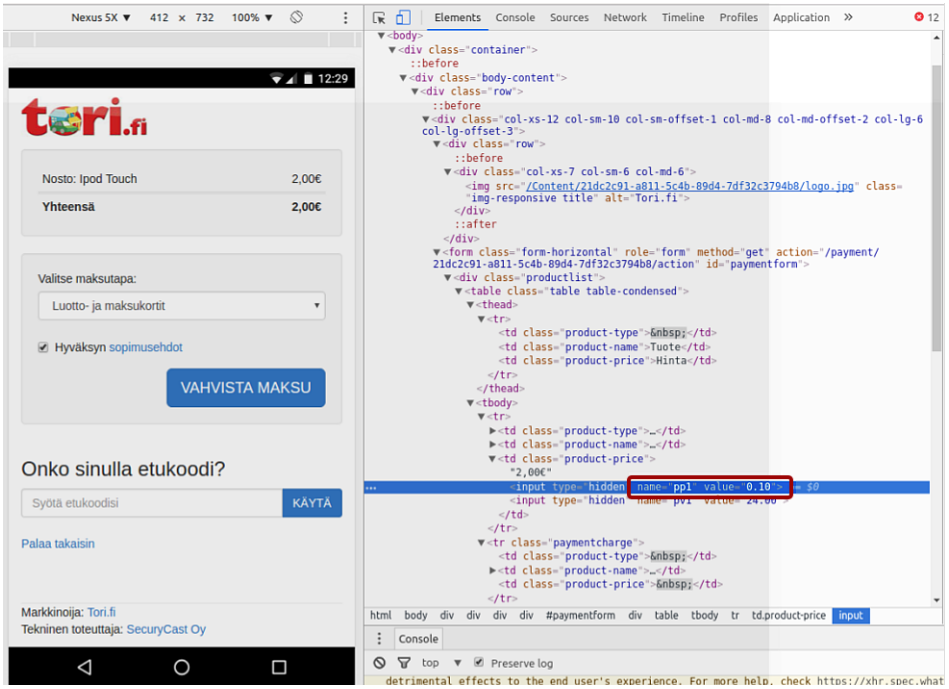

Fig. 3.23: New value set to the product price.

TS.a12 **Results:** Achieved. Payment is requested with the price set and confirmation of product is received. Since the *Mobiilimaksu* payment is processed without further authorisation required, the payment method was change to credit card to make it evident the new price was accepted and the payment transaction was processed.

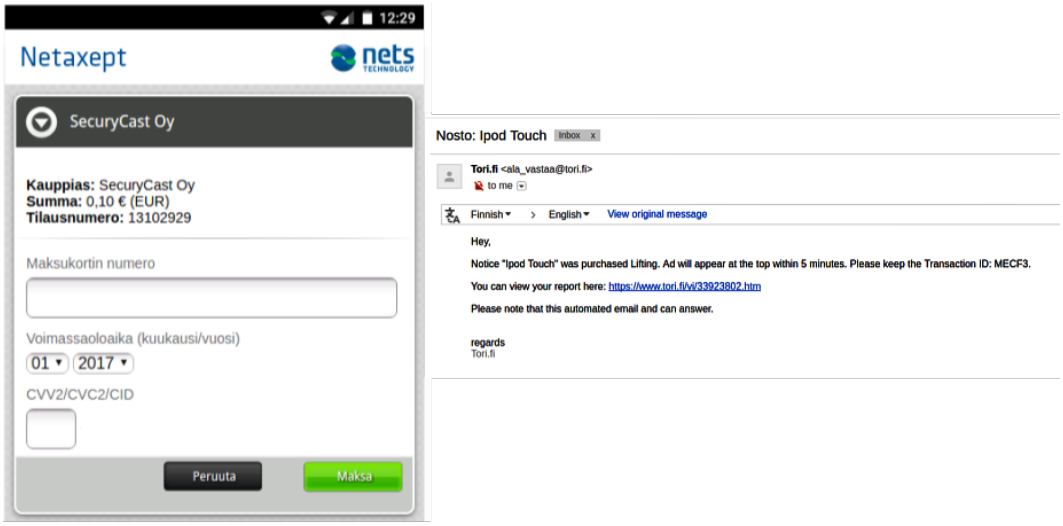

Fig. 3.24: New price to pay by card.

Others with Securycast

After identifying a possible design weakness in the way Securycast processes transactions, it was important to test if this was a particular case of the payment processing for the merchant Tori, or if it was a general weakness in the payment services offered by Securycast. Therefore, different merchants using Securycast services were required. Securycast, in its web page, publishes a list of partners who are, by now, potentially vulnerable. To avoid harm, a charity organization, Nena Paiva, was selected from the list[9]. Then, the traffic was analysed during the donation transaction to define the attack scenario below described.

S.a01 **Goal:** Validate if tampering the payment data is successful in a different customer of Securycast.

S.a01 **Related Threat ID:** TS.t06.

S.a01 **Description:** Nena Paiva allows a minimum donation of 10 Euro. The attack consists of tampering the message to Securycast and donate only 1€.

[9]Securycast Partners www.securycast.com/en/References

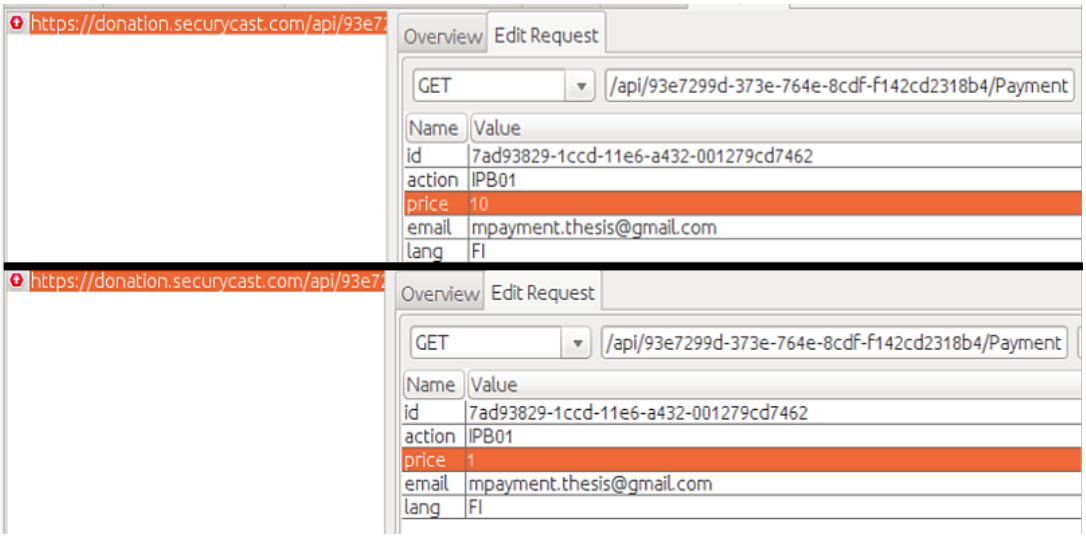

Fig. 3.25: Tampering donation details in the request to Securycast.

S.a01 **Results:** Achieved. It was possible to change the amount of donation to 1 Euro. Securycast does not validate the integrity of the messages from the browser. It was also noticed that, in the donation page, there is no verification code sent from the donation server to Securycast.

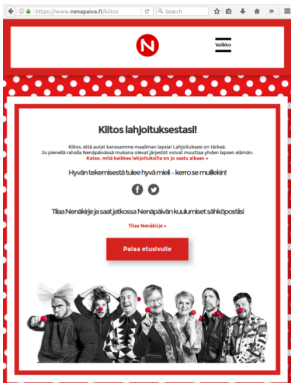

Fig. 3.26: Donation of 1€ successful.

S.a02 **Goal:** Capture payments from different customers of Securycast.

S.a02 **Related Threat ID:** TS.t14.

S.a02 **Description:** Utilize a valid *organizationID* provided by Securycast to deviate transactions by modifying the value in the requests to Securycast.

S.a02 **Results:** Failed. The *cart ID* created by Securycast is linked to the *organization ID*, thus, the transaction pertained to certain *cart ID* cannot be processed for a different organization that the one that originated it.

PayiQ Mobile Ticketing

After using the PayiQ Tickets application and analysing the messages exchanged during a ticket purchase, it was possible to define the attack scenarios to achieve the goals listed in the adversary model. We tested the security mechanisms implemented by PayiQ to protect the payment transactions as follows.

Man in the Middle

PT.a01 **Goal:** Obtain information about the payment process by eavesdropping the communication between the client and the PayiQ servers.

PT.a01 **Related Threat ID:** PT.t02, PT.t03, PT.t04.

PT.a01 **Description:** The communication between the application and the servers occurs over TLS. The servers utilize digital certificates to authenticate themselves, the weak point to achieve the goal resides in the client. The attack consists of installing a TLS proxy in the phone where the PayiQ Tickets resides to decipher the messages.

PT.a01 **Results:** Achieved. The attack was successful, the PayiQ servers did not refuse the connection, however, it requires root permissions, therefore, does not occur unwittingly to the consumer. For this thesis, the attack was useful for reverse engineering.

Impersonation

PT.a02 **Goal:** Purchase tickets via the application without effecting the payment by impersonating a legitimate user holding a mobile subscription.

PT.a02 **Related Threat ID:** PT.t1.

PT.a02 **Description:** To impersonate a user, it is necessary to obtain their credentials. This can be done applying social engineering. Then, with the username and password of a legitimate user, access the application (in a different phone) to buy a ticket.

PT.a02 **Results:** Failed. The payment failed. A message was received indicating the phone number of the logged user was different from the billing phone number, i.e., the one detected in the connection (Fig. 3.27).

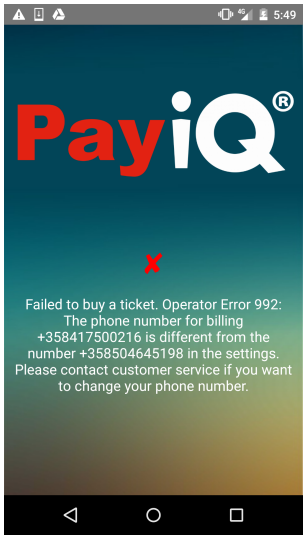

Fig. 3.27: Operator Error

Replay

PT.a03 **Goal:** Reproduce the behaviour of the PayiQ Tickets application on a personal computer to obtain a one-time password to access the store.

PT.a03 **Related Threat ID:** PT.t01, PT.t06, PT.t10.

PT.a03 **Description:** From the protocol illustrated before (Fig. 3.11), it is inferred that each request to the PayiQ API server requires a one-time password (OTP) except from the first request which is the point where a replay attack could succeed. The attack consists of sending the previously recorded message for a *consumerget* request to the API server.

PT.a03 **Results:** Achieved. The request was successful, the API server returns the information about the user and a new OTP (Fig. 3.28). It was observed that the same replayed message can be used to obtain any number of OTPs. No freshness is required for the request to be processed successfully.

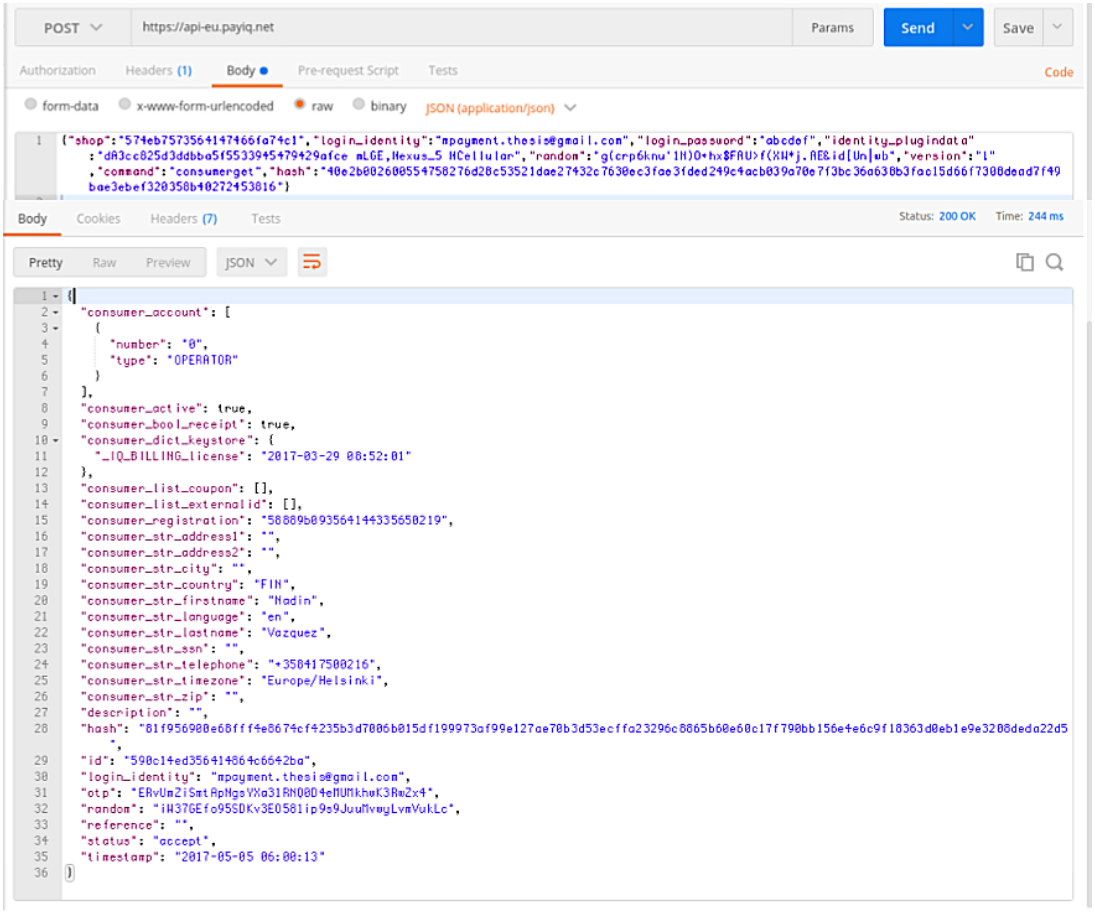

Fig. 3.28: Replay attack response.

Forgery

PT.a04 **Goal:** Acquire tickets forcing someone else to pay by forging a mobile connection and misusing a shared mobile Internet connection.

PT.a04 **Related Threat ID:** TS.t05.

PT.a04 **Description:** Connect to the mobile network of a Finnish MNO subscription holder via Wi-Fi. Then, in the tethered device, purchase tickets through the application using the *Mobiilimaksu* payment method.

PT.a04 **Results:** Failed. The application requires a mobile connection to function properly. Utilizing the application while being connected to a Wi-Fi network produces an error message.

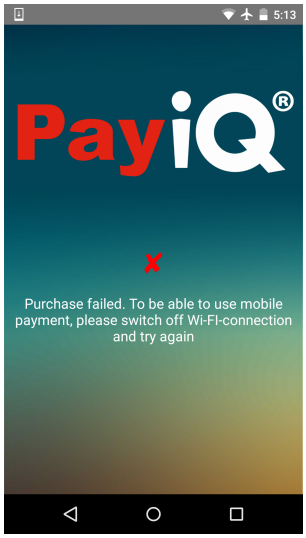

Fig. 3.29: Application verifies the connection type.

PT.a05 **Goal:** With the OTP received in the attack PT.a03, continue the attempt to purchase tickets by reproducing the application behaviour on a PC tethered to a device with mobile Internet connection. This will avoid the network verification of the application while still taking advantage of the *Mobiilimaksu* payment method in shared mobile networks.

PT.a05 **Related Threat ID:** PT.t02.

PT.a05 **Description:** It was observed that without an OTP or using an old one in the step 3 of the protocol, the Store server would return an error message. Thus, the attack consists of mimicking the application by forging the application agent in a desktop browser. Then, make the GET request in step 3 of the protocol with the OTP received in the attack PT.a03.

PT.a05 **Results:** Failed. The request was successful and the access to the store was granted. In the store, it was possible to select the ticket. Nonetheless, the process failed when clicking the "buy" button. Instead of receiving the form to enter the PIN and authorise the purchase an error message was displayed (Fig. 3.30), terminating the purchase process there because we were unable to emulate the interactive step of the application in the PC.

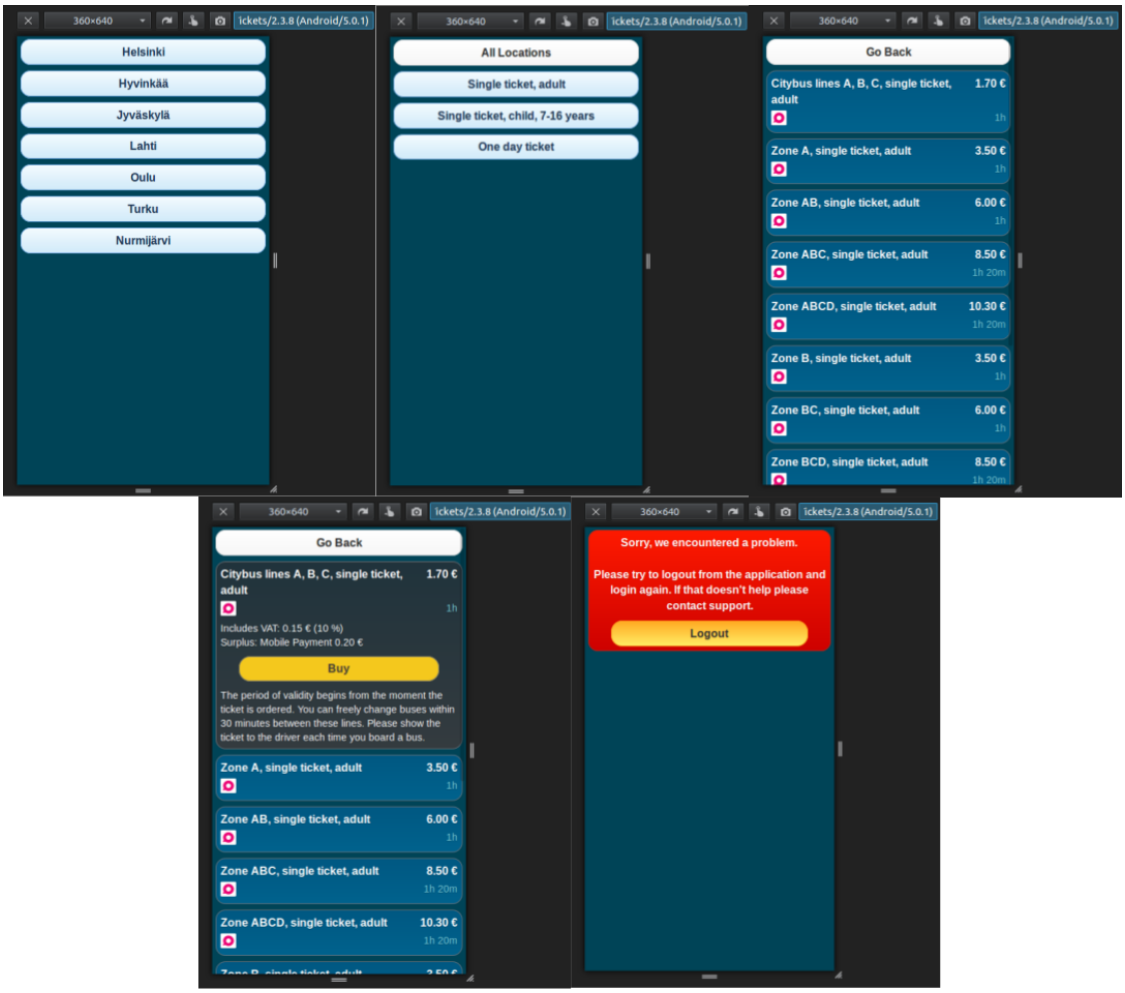

Fig. 3.30: Browsing the store via desktop browser.

Tampering

PT.a06 **Goal:** Skip the requests to the Store server for product selection and the PIN entry, which we were unable to emulate on the PC. Now attempting to acquire tickets by sending the request with the product details directly to the API server, utilizing an OTP received from the replay attack.

PT.a06 **Related Threat ID:** PT.t07.

PT.a06 **Description:** A different approach to find weaknesses in the PayiQ protocol is to tamper a request to the API server providing a new OTP to bypass the authentication process. The attack consists of crafting a

new *consumerpayment* request (step 8 of the protocol) utilizing a new OTP to try to proceed with the ticket order.

PT.a06 **Results:** Failed. No error is received about the login data, OTP is valid. Nonetheless, a different error is produced as seen in figure 3.31.

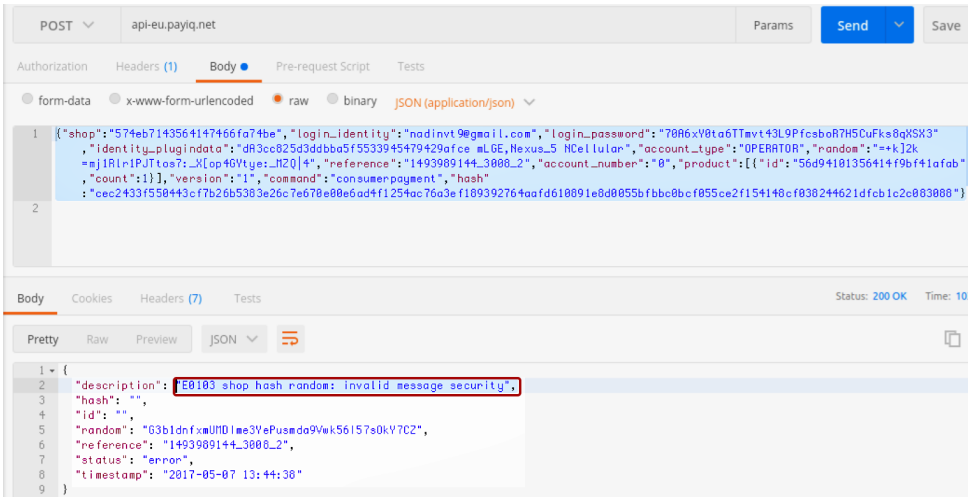

Fig. 3.31: Error in the message integrity validation.

Privilege Elevation

PT.a07 **Goal:** Acquire tickets with stolen or borrowed phone or SIM card of a mobile subscriber.

PT.a07 **Related Threat ID:** PT.t01, PT.t12.

PT.a07 **Description:** With the phone and the credentials of a legitimate user (previously gathered applying social engineering), access the application and purchase tickets.

PT.a07 **Results:** Achieved. Authorising a purchase requires a PIN, only known by the user. However, a new PIN is created every time the user logs in. Therefore, by logging out and logging in again, the attacker can generate a new PIN to authorise purchases. Additionally, registering a new account with a phone number previously registered was successful. Thus, to pay for tickets, it is possible to create a new account using the phone number of a legitimate user, as long as, the attacker has access to the SIM card of the legitimate user.

Reverse Engineering

PT.a08 **Goal:** Gather additional information about the application to reproduce its behaviour on a PC to finally succeed at acquiring tickets without any use of the PayiQ Tickets application by entirely emulating the requests from the application. Using a legitimate username and password to PayiQ but misusing a shared mobile Internet connection.

PT.a08 **Related Threat ID:** PT.t01, PT.t06.

PT.a08 **Description:** From the transaction flow it was observed that each message exchanged between the API server and the application has a hash. The hash value seems to be utilized as a HMAC to validate the integrity of each message. Thus, the attack consists of performing reverse engineering to the installation file of the application to discover the function used to create the HMAC. Subsequently, craft new messages with valid hash values for the server to process them as authentic. The attack requires the following phases:

1. The first phase consists of finding the function to create the HMAC and determine the required input to generate a valid integrity code for each type of message in the transaction flow.
2. The second phase consists of identifying valid values for each parameter in the request message (e.g., device details, reference number, account number, product id).
3. Next step is to craft the messages and send them to the servers to get the values necessary to proceed with the transaction (i.e., OTP, transaction ID, token).
4. Identify the manner to bypass the authorization and pass the verification towards the MNO billing server for the transaction to be accepted.
5. If the transaction is accepted, the last step is to open the application and find if the ticket can be retrieved for proper display.

PT.a08 **Results:** Achieved. The function to produce the HMAC and the key, were found and replicated to produce valid HMACs for new messages. New POST requests to the API Server were crafted and successfully processed by the API server. A transaction ID was created to acquire a ticket. The API server sent the token to charge the MNO. With

the token, it was possible to reach the verification server [10], avoiding the PIN entry. It was noticed that the connection to the verification server must be through a mobile network for the transaction to be accepted. With the tested operator, no matching phone numbers between the used SIM and the one entered during the account registration are required to complete the payment. After validating the successful transaction with a *query* request, the application was opened and the ticket bought, completely outside the real application, was retrieved .

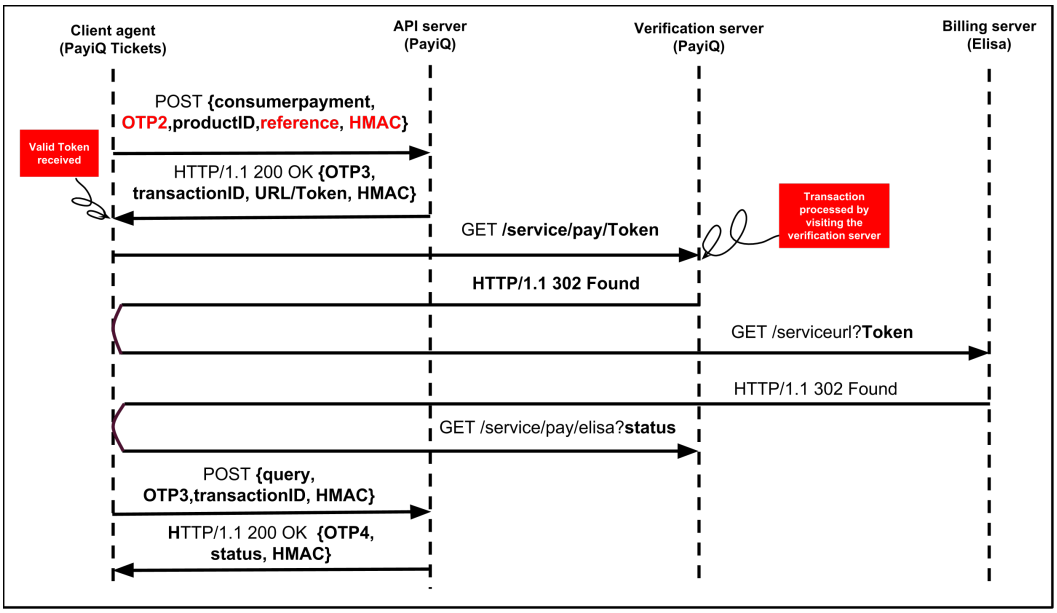

Fig. 3.32: Spoofing the application to bypass its security mechanisms.

Mitigation

This section concludes the security analysis with the controls that could be implemented by the systems under study to mitigate the threats for which an attack was successful.

[10]The attack succeeds if the Internet connection is provided by Elisa. It is important to notice that the verification server will return a mismatch phone number error if the request is done through the Telia (Sonera) network.

Tori with Securycast

The system comprised by Tori and Securycast was the most vulnerable one. The table 3.3, below, presents the summary of the attacks performed along with the indicator of whether the threats related to certain attack were mitigated or not by the developers.

Table 3.3: Summary of the attacks and mitigation status.

Attack Identifier	Attack Type	Threat Related	Mitigated
TS.a01	Man in the Middle	TS.t02, TS.t12	NO
TS.a02	Man in the Middle	TS.t04, TS.t05	YES
TS.a03	Impersonation	TS.t01	NO
TS.a04	Replay	TS.t05, TS.t06, TS.t07	YES
TS.a05	Cross Site Request Forgery	TS.t06	NO
TS.a06	Forgery	TS.t03	YES
TS.a07	Tampering	TS.t02, TS.t08	YES
TS.a08	Tampering	TS.t02, TS.t09	YES
TS.a09	Tampering	TS.t02, TS.t09	NO
TS.a10	Tampering	TS.t02, TS.t09	NO
TS.a11	Tampering	TS.t02, TS.t09	NO
TS.a12	HTML Injection	TS.t09	NO
S.a01	Others	TS.t09	NO
S.a02	Others	TS.t14	YES

A mitigation is proposed below for those cases where no countermeasures were found to prevent an attack from occurring.

TS.m01 **Problem:** It is possible to impersonate the client to the server and vice versa.

TS.m01 **Related Attack:** TS.a01.

TS.m01 **Related Threat ID:** TS.t02, TS.t12.

TS.m01 **Countermeasure:** The problem is mitigated with the use of TLS connections. However, the authenticity of the client is not verified by the servers, which leaves the possibility to eavesdrop the traffic at the client end. The attack described in previous section was accomplished because explicit trust was given to a forged certificate for the proxy acting as the Man in the Middle, the consumer is aware, at all times, about the presence of the intermediary.

TS.m02 **Problem:** *Mobiilimaksu* payment method is accepted if a mobile connection is detected, no authorization is required.

TS.m02 **Related Attack ID:** TS.a03.

TS.m02 **Related Threat ID:** TS.t01.

TS.m02 **Countermeasure:** Provide means for a mobile subscriber to confirm and authorise each payment transaction. This can be accomplished by assigning a randomly generated authorization code to the subscribers.

TS.m03 **Problem:** The merchant can bypass the consumer confirmation to the payment gateway, triggering unauthorised payments.

TS.m03 **Related Attack ID:** TS.a05.

TS.m03 **Related Threat ID:** TS.t06.

TS.m03 **Countermeasure:** The payment service provider, Securycast, should enforce the correct flow of the transaction from the moment the merchant triggers the request to the webcart server. This could be done using session tokens, provided by the server. The webcart server generates a session token and sends it along with the webcart form, the next request sent, confirming the payment, should include the token previously generated by the server.

TS.m04 **Problem:** Information about payment and product details is easily identifiable.

TS.m04 **Related Attack ID:** TS.a09, TS.a10, TS.a11, TS.a12.

TS.m04 **Related Threat ID:** TS.t02, TS.t09.

TS.m04 **Countermeasure:** Replace critical data with a representation of the same, preventing its identification, hence, modifiability. This can be accomplished by applying tokenization to the payment details.

TS.m05 **Problem:** It is possible to make unauthorised changes to the payment details without being detected by the payment service provider.

TS.m05 **Related Attack ID:** TS.a09, TS.a10, TS.a11, TS.a12.

TS.m05 **Related Threat ID:** TS.t02, TS.t09.

TS.m05 **Countermeasure:** Provide means to verify the integrity of each message transmitting critical data for the transaction process. To achieve this, a checksum/HMAC code must be generated along with the messages to be transmitted in the sending point and verified at the receiving point.

PayiQ Mobile Ticketing

The application developed by PayiQ resulted resilient to most of the threats. The table 3.4, below, presents the summary of the attacks performed along with the indicator of whether the threats related to certain attack were mitigated or not by the developers.

Table 3.4: Summary of the attacks and mitigation status.

Attack Identifier	Attack Type	Threat Related	Mitigated
PT.a01	Man in the Middle	PT.t02	NO
PT.a02	Impersonation	PT.t01	YES
PT.a03	Replay	PT.t01, PT.t06, PT.t10	NO
PT.a04	Forgery	PT.t05	YES
PT.a05	Forgery	PT.t02	YES
PT.a06	Tampering	PT.t07	YES
PT.a07	Privilege Elevation	PT.t01, PT.t12	NO
PT.a08	Reverse Engineering	PT.t01, PT.t06	NO

Similarly, to the Tory with Securycast case study, a mitigation is proposed below for those cases where no countermeasures were found to prevent an attack from occurring in the PayiQ Tickets application.

PT.m01 **Problem:** It is possible to impersonate the client to the server and vice versa.

PT.m01 **Related Attack:** PT.a01.

PT.m01 **Related Threat ID:** PT.t02.

PT.m01 **Countermeasure:** The problem is mitigated with the use of TLS connections. However, the authenticity of the client is not verified by the server which leaves open the possibility to eavesdrop the traffic at the client end. The attack described in previous section was accomplished because explicit trust was given to a forged certificate for the application acting as the Man in the Middle, the consumer is aware, at all

times, about the presence of the intermediary. As far as what PayiQ concerns, they could validate the integrity of the downloaded application and platform by making use of trusted computing technology for remote attestation [11] of each client loading its application. However, attesting each client with this method may not be very realistic solution.

PT.m02 **Problem:** It is possible to receive one-time passwords (OTP) from the servers by re-utilizing an authentic message.

PT.m02 **Related Attack:** PT.a03.

PT.m02 **Related Threat ID:** PT.t01, PT.06,PT.10.

PT.m02 **Countermeasure:** Provide freshness to each message exchanged during the transaction process to ensure new requests are served, exclusively. This is accomplished by adding a timestamp or nonce to each message.

PT.m03 **Problem:** It is possible to change the authorization PIN by creating a new session with the PayiQ servers.

PT.m03 **Related Attack:** PT.a07.

PT.m03 **Related Threat ID:** PT.t01, PT.t12.

PT.m03 **Countermeasure:** Generate a random PIN that will last more than a session to prevent it being easily guessable and changeable.

PT.m04 **Problem:** It is possible to bypass the authorization and authentication performed by the application and acquire tickets using the mobile connection of an Elisa subscriber.

PT.m04 **Related Attack:** PT.a08.

PT.m04 **Related Threat ID:** PT.t01, PT.t06.

PT.m04 **Countermeasure:** Provide means for a mobile subscriber to confirm and authorise each payment transaction directly to the MNO billing server, rather that leaving the authentication and authorisation tasks to the merchants and payments processors. This can be accomplished by

[11] Remote attestation: Method where a verifying system (server) determines if a remote system (client) is trustworthy by analysing evidence of its state, e.g., a hash of the software that has been loaded on the client [17].

assigning a randomly generated authorization code to the subscribers. Additionally, the verification server should always verify the phone number of the account in the request, matches the phone number to bill for the transaction, preventing the misuse of shared mobile networks. Registering each instance of the application, binding it to the phone number of the subscriber and creating a unique shared secret for each instance with the API server would make the task of spoofing the requests to the API more difficult for attackers.

Chapter 4

Evaluation

This chapter presents an evaluation of the systems studied regarding their success or failure to meet the security requirements of payment solutions.

Identification

Tori with Securycast Although it is unnecessary to register to Tori to enjoy its services, it has an option to sign up as a user. To identify a user, Tori requires a valid email address, also used to activate the account. Securycast, for its part, serves from a gateway in charge of detecting the IP Address of the incoming connection, the operator, and the phone number (MSISDN) to identify consumers/subscribers. The gateway succeeds at identifying the subscriber to request the charge to the MNO as seen in the figure below.

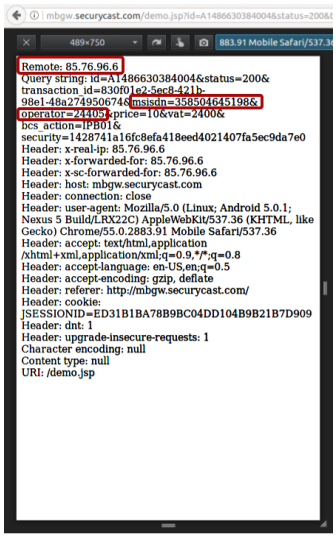

Fig. 4.1: Payment gateway response in the Elisa network

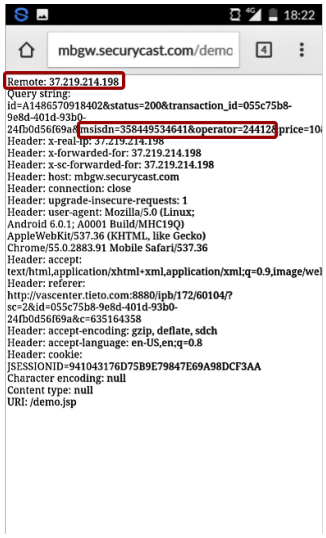

Fig. 4.2: Payment gateway response in the DNA network

PayiQ Mobile Ticketing By creating a user account in the PayiQ application, the consumer is identified with an email that serves as the user ID, and a phone number. Both remained linked from the registration onwards. The data cannot be modified by the owner of the account but only by PayiQ administrators.

Authentication

Tori with Securycast If registered to Tori, users must provide a password of a minimum of 5 characters long, no level of complexity is required (e.g., a mix of alphanumeric characters, upper and lower cases, inclusion of special characters), therefore, it is not considered as a strong authenticator [11].

PayiQ Mobile Ticketing PayiQ uses a single factor to authenticate its users to the application. The authenticator is a password which comprises only 6 characters without any level of complexity. Such password represents a weak point in the system. Advantageously, authentication occurs for each request coming from the application to the PayiQ servers with a HMAC.

Authorisation

Tori with Securycast A single click is required to confirm and execute the payment in the webcart form. No mechanism was found to bind the confirmation from the consumer or to authorise the payment. Tethering poses a security risk to subscribers. Since subscribers have no means to prevent unwanted purchases, transactions may occur unwittingly to them until receiving the mobile bill. Furthermore, Tori is capable of forging payment confirmations to the webcart server.

PayiQ Mobile Ticketing PayiQ implements authorization controls to ensure the appropriate user confirms the purchases. In the application, the PIN code set by the user is entered to authorise the payment, after the selection of the payment method. The PIN is verified by the application and never leaves the mobile device. However, it is set by the user, opening the possibility of being guessable and changeable as it lasts only for the session the user is logged to the application. If the user logs out or logs into a different device, the session is lost and, when the user access again, a new PIN must be set.

Integrity

Tori with Securycast Both parties, Tori and Securycast, attempt to provide integrity to their data by adding a checksum such as HMAC to the requests coming from the client agent. However, they fail at merging their solutions for integrity. Tori protects the payment data and product details sent to the webcart server, in the step 3 of the protocol (Fig. 3.5), by passing the parameter **c**, a 128 Hex-character checksum. The webcart server verifies the integrity of the data before sending the form for checkout in step 4. The server sends, in response, the form containing the data provided by Tori. However, before creating the cart ID in step 5 and proceeding with the payment, Securycast does not validate for a second time the integrity of the data, allowing the process to continue with tampered data. In further steps of the protocol, the presence of a 40 Hex-character HMAC is observed under the name "security", protecting the data exchanged between Securycast and the MNO Elisa.

PayiQ Mobile Ticketing PayiQ succeeds at providing integrity to the data exchange between the application and its servers by adding a 128 Hex-character HMAC to each message in a POST request to the API server.

Confidentiality

Tori with Securycast Regarding the confidentiality of data, a greater part of the protocol in the Tori-Securycast transactions occurs over TLS which ensures data is traversing the internet through secured, ciphered, channels within authenticated parties. The one server leaking data is the gateway server, exposing the mobile number of the subscriber.

PayiQ Mobile Ticketing Concerning the confidentiality of data, the communication between the application and the servers occurs through a secured channel implementing TLS. Additional confidentiality is provided with the use of Tokens to communicate information about the transaction to the MNO.

Audit mechanisms and non-repudiation

Tori with Securycast As far the investigation concerned, Tori page does not provide ample audit support to users, starting from the fact that the entire universe of its users is not registered and continuing to the observation that the activity of registered users does not remain for more than 3 months. It is unknown if there are logs between Securycast and Tori about the payment transactions and the information they may include.

PayiQ Mobile Ticketing Compared to Tori, PayiQ is in a better position to provide audit support and guarantee non-repudiation because its users pass through a registration process providing the means to identify them and link their actions to them through a unique identifier. Additionally, the application keeps logs of all the transactions, both successful and failed ones.

Chapter 5

Discussion

In an ideal world, the mobile payment solutions presented in this thesis would have appropriate mechanisms to fulfil the security requirements listed in the chapter 2. However, providing security is a task that requires exhaustive analysis and more effort than just utilizing cryptographic mechanisms such as hash functions and ciphered channels. This chapter presents a discussion of the outcome of the security analysis.

At a glance, the entire payment process implemented by Tori and Securycast seemed rather simple: User accepts to pay for an upgrade with a single click, subsequently, the mobile operator charges the user. Nonetheless, the observed security in the Tori web page was questionable, especially by noticing the following:

- Using Tori services to publish an advertisement does not require any type of registration, which makes it difficult to track the actions of a particular user.
- Neither Tori nor Securycast verify the client agent (i.e., Tori mobile application, mobile or web browser) used to access its services to determine whether the *Mobiilimaksu* payment method should be available or not.

After completing the security analysis, it was clear that the lack of authentication controls or the implementation of weak ones does not expose sensitive data about the customer or subscriber when executing a payment transaction. However, neglecting the main security services required by mobile payments make Tori an ideal platform for scams. Whether an unregistered user publishes a counterfeit advertisement to attract honest buyers, leaving little trace for prosecution, or a malicious user accesses the account of honest sellers to mislead their buyers, Tori seems to be risky platform for trading, even

though, its purpose is only to connect sellers and buyers. Most importantly, the way Securycast serves the webcart form and processes payments poses a security breach, allowing Tori, if it became malicious, to triggered payment request without the approval and confirmation of the customer. This raises a concern about payment service providers trusting the merchants to participate in the execution of payment transactions.

While we have no reasons to suspect any criminal activity by Tori or other known merchants, the trust model is not as robust as it could be. The observations mentioned above, represent an opportunity for Tori and Securycast to improve their services. Providing appropriate identification, authentication and strengthening the correct transaction flow would increase the level of security to protect customers and their payment transactions.

On the other hand, a significantly different protocol was found using the PayiQ Tickets application, not only because PayiQ carries out the entire payment process acting as both the merchant and the payment service provider but also because the mobile application enables more security mechanisms than web applications.

To access PayiQ services, a registration must be completed, additionally, the users must provide a PIN code to authorise the payment. These are two additional steps in the process of acquiring digital products, in comparison to single-click solutions. Nonetheless, we still found areas of opportunity in the way PayiQ offers electronic tickets:

- From the installation file of the application, it was possible to find the key and function that generates the HMAC for each message to the API server. It is possible that additional obfuscation is required in the code of the application to make it less intelligible, reducing the probability of spoofed requests to the API. It is also advisable that a unique key (shared secret) is created for each instance of the application to authenticate requests to the API server.

- The transaction flow is enforced with the use of one-time passwords generated by the API server. However, the field used for this parameter in the same as the one for the login password of the user. Having two different valid values for the same parameter make it vulnerable to spoofing, breaking the correct flow of the transaction.

Additionally, fraud is an important aspect to prevent in financial transactions. However, the user accounts used in Tori and PayiQ Tickets to perform the different tests, were never blocked or disabled by any of the service providers, even though unusual transactions were executed with them. Furthermore, it was observed that payment service providers processed the same

transaction more than once, which makes them prone to fraudulent payment transactions. Regarding the participation of the mobile operator Elisa, the mechanism they used to protect direct carrier billing payments remain unknown, as no technical information is publicly available. The only controls observed by the MNO is a limit of 150 € per month per mobile subscription destined to the direct carrier billing payment *Mobiilimaksu* and the option to disabled this payment method in the account settings of the subscribers.

Overall, the direct carrier billing method seemed to be strictly focused on identifying the mobile subscription to accept payments. However, additional mechanism should be in place to verify the authenticity of the payment request from the consumer including the intended payee and amount. It is still questionable who should be the owner of that task, the payment service provider or the mobile operator. If the type to payments allowed for direct carrier billing were to change from micro payments to larger amounts of money, it may be worth for either one of the payment service provider or the mobile operator to invest in providing better mechanisms to validate the authenticity of payment requests and provide means for the consumer or subscriber to authorise payments, sacrificing the simplicity of single-click transactions for better security.

Chapter 6

Conclusion

The security analysis of two case studies of direct carrier billing show that despite the development of security mechanisms to protect electronic payment transactions, service providers face difficulties to offer a secure solution. None of the payment solutions studied offer services completely immune to security threats. The major vulnerabilities found were:

1. If the merchant takes part in the payment transaction, it can forge the payment confirmation from the user to appear legitimate to the carrier billing server.

2. Wi-Fi tethering can be misused to complete payments on behalf of the mobile subscriber account.

3. It is possible to bypass the security of a mobile application.

PayiQ was found to provide better security to its users and merchants who are not required to develop mobile or web applications to offer their products to consumers, unlike Securycast. However, this comes at the cost of less flexibility as there is no separate merchant in the system and it is still possible to misuse it services.

Securycast, for its part, offers a solution similar to Stride and PayPal, which is a webcart (checkout) page that handles electronic payments. However, it lacks the use of tokenization to send data to the payment gateway (as Stripe does) or to tokenize transaction details set by the merchant (as PayPal does). This, in addition to the integrity validation that should perform when sending or receiving sensitive transaction data.

The transactions that both studied payment service providers handle remain in the category of micro payments, which reduces considerably the

financial risks in each transaction. However, in a system accessible by thousands of users, malicious users may leverage the ability to tamper the process without being detected, increasing the damage considerably.

Overall, Tori, Securycast and PayiQ seem to have an idea about the appropriate mechanisms to secure mobile payments. However, overlooked sensitive entry and exit points in the flow of the transaction may jeopardise the entire security chain implemented to protect the payment transactions. From the security analysis, it is concluded that:

1. In the basis of carrier billing, tethering is like sharing the wallet. Mobile subscribers sharing their connection are vulnerable to receiving charges for unwanted payments executed through their connection.

2. Payment service providers should not trust the merchants. Malicious merchants are in position to request for payment transactions by tricking honest consumers.

3. The conjunction of registered users and mobile applications can offer more security than unregistered users and web applications for payments, at the cost of less flexibility in the business model.

4. In-app security controls are not sufficient for payment transactions. Controls should be implemented in the servers as well.

5. The simplicity of single-click transactions pose risks for consumers.

For now, it is recommended for consumers or subscribers to disable direct carrier billing payment method from their mobile subscription or to pay special attention to whom they share their mobile connection as well as their browsing activity to avoid malicious merchant sites.

Bibliography

[1] Marie Baker. Striving for effective cyber workforce development. page 26, 2016.

[2] The World Bank. *Mobile Cellular Subscriptions*. The World Bank Group, 1960-2015. data.worldbank.org/indicator/IT.CEL.SETS.

[3] Steven F. Burns. *Threat Modeling: A Process To Ensure Application Security*. SANS Institute InfoSec Reading Room, 2005. www.sans.org/reading-room/whitepapers/securecode/threat-modeling-process-ensure-application-security-1646.

[4] M. Carbonell, J. Torres, D. Suarez, J. M. Sierra, and J. Tellez. Secure e-payment protocol with new involved entities. In *2008 International Symposium on Collaborative Technologies and Systems*, pages 103–111, May 2008.

[5] Matt Collins. *Best Payment Gateways Reviewed and Compared*. Ecommerce Platforms, March 2017. ecommerce-platforms.com/ecommerce-selling-advice/choose-payment-gateway-ecommerce-store.

[6] Telia Community. *DIMOCO Europe GMBH*. Telia Company, Novemeber 2014. yhteiso.telia.fi/t5/forums/v3_1/forumtopicpage/board-id/asiakas/highlight/true/page/1/thread-id/1086.

[7] T. Dahlberg, N. Mallat, and A. Öörni. Trust enhanced technology acceptance model-consumer acceptance of mobile payment solutions: Tentative evidence. *Stockholm Mobility Roundtable*, 22:23, 2003.

[8] Fortumo. *Direct Carrier Billing in 2016: Global Market Report*. Fortumo, January 2017. fortumo.com/blog/direct-carrier-billing-in-2016-global-market-report-by-fortumo.

[9] A. Ghezzi, F. Renga, R. Balocco, and P. Pescetto. Mobile payment applications: offer state of the art in the italian market. *info*, 12(5):3–22, 2010.

[10] Dieter Gollmann. *Computer Security*. Wiley, 2011.

[11] P. Grassi, J. Fenton, E. Newton, R. Perlner, A. Regenscheid, W. Burr, J. Richer, N. Lefkovitz, J. Danker, Y. Choong, K. Greene, and M. Theofanos. *Digital Identity Guidelines*. National Institute of Standards and Technology, 2017. pages.nist.gov/800-63-3/sp800-63b.html.

[12] H. Harb, H. Farahat, and M. Ezz. Securesmspay: Secure sms mobile payment model. In *2008 2nd International Conference on Anti-counterfeiting, Security and Identification*, pages 11–17, Aug 2008.

[13] P. Hartman, J.P. Bezos, K. Shel, and J. Spiegel. Method and system for placing a purchase order via a communications network, 1999.

[14] H. Ho, S. Fong, and Z. Yan. User acceptance testing of mobile payment in various scenarios. In *2008 IEEE International Conference on e-Business Engineering*, pages 341–348, Oct 2008.

[15] Apple Inc. *iOS Security-White Paper*. Apple Inc.., March 2017. support.apple.com/en-us/HT207143.

[16] Weidong Kou. *Payment Technologies for E-Commerce*. Springer Science & Business Media, March 2013.

[17] Kubilay Ahmet Küçük, Andrew Paverd, Andrew Martin, N. Asokan, Andrew Simpson, and Robin Ankele. Exploring the use of intel sgx for secure many-party applications. In *Proceedings of the 1st Workshop on System Software for Trusted Execution*, SysTEX '16, pages 5:1–5:6, New York, NY, USA, 2016. ACM.

[18] V. Kumar, J. Srivastava, and A. Lazarevic. *Managing Cyber Threats: Issues, Approaches, and Challenges*. Massive Computing. Springer, 2010.

[19] L. Mainetti, L. Patrono, and R. Vergallo. Ida-pay: An innovative micropayment system based on nfc technology for android mobile devices. In *SoftCOM 2012, 20th International Conference on Software, Telecommunications and Computer Networks*, pages 1–6, Sept 2012.

[20] Niina Mallat. Exploring consumer adoption of mobile payments – a qualitative study. *The Journal of Strategic Information Systems*, 16(4):413 – 432, 2007.

[21] Mozido. *Mobile payments then & now.* Mozido Inc., June 2014. www.mozido.com/mobile-payments-now/.

[22] Ripoff Report. *Complaint Review: DIMOCO, GmbH.* Ripoff Report, November 2015. www.ripoffreport.com/reports/dimoco-gmbh/brunn-am-gebrige-a-2345/dimoco-gmbh-dimoco-americas-also-dimoco-eu-country-locations-serves-as-the-interface-f-1267823.

[23] A. Sarajlic and D. Omerasevic. Access channels in m-commerce services. In *2007 29th International Conference on Information Technology Interfaces*, pages 507–512, June 2007.

[24] Michael Schmidt. Consistent M-commerce security on top of GSM-based data protocols, a security analysis, 2002.

[25] Adam Shostack. *Threat Modelling.* John Wiley And Sons, Incorporated, Somerset, UNITED STATES, 2014.

[26] Edwin Smith. *Meet the top 20 hottest payment companies.* Raconteur Media Ltd., March 2015. www.raconteur.net/technology/meet-the-hottest-payment-companies.

[27] Telia Sonera. *MSISDN Enrichment.* TeliaSonera Finland Oyj, April 2014. www.telia.fi/dam/jcr:0006ef83-c2b9-43db-a743-78870a5687a7/msisdn_enrichment.pdf.

[28] M. A. Tehrani, A. A. Amidian, J. Muhammadi, and H. Rabiee. A survey of system platforms for mobile payment. In *2010 International Conference on Management of e-Commerce and e-Government*, pages 376–381, Oct 2010.

[29] Oxford University. *Oxford English Dictionary.* Oxford University Press, third edition, December 2015. www.oed.com.

[30] Viestintävirasto. *Telecommunication Markets in the Nordic and Baltic Countries.* Viestintävirasto, Finnish Communications Regulatory Authority, June 2016. www.viestintavirasto.fi.