UNIVERSIDADE DE LISBOA

FACULDADE DE CIÊNCIAS

DEPARTAMENTO DE INFORMÁTICA

SECURING USSD IN MOBILE FINANCIAL TRANSACTIONS

(A PRACTICAL PROPOSAL FOR M-FINANCE)

Paula Margarida Mendonça da Silva Cravo

MESTRADO EM SEGURANÇA INFORMÁTICA

Dezembro 2011

UNIVERSIDADE DE LISBOA

FACULDADE DE CIÊNCIAS

DEPARTAMENTO DE INFORMÁTICA

SECURING USSD IN MOBILE FINANCIAL TRANSACTIONS

(A PRACTICAL PROPOSAL FOR M-FINANCE)

Paula Margarida Mendonça da Silva Cravo

Tese orientada pelo Professor Jason Hong
e co-orientada por Professor Dr. Marcelo Pasin

MESTRADO EM SEGURANÇA INFORMÁTICA

Dezembro 2011

Acknowledgments

This work would not be possible if it wasn't for the help of a lot of people, and I would like to refer some of the most important persons that supported me throughout this Master thesis and during the MSc graduation work.

First of all, I must refer Professor Jason Hong, from CMU, who was very attentive, questioning me about my decisions and guiding me through studies paths that came out to be this work. Without his help, I would still be immersed in document analysis, searching for a possible solution for my quest. Thank you very much, Professor Hong.

Secondly, I would like to thank PT and PT Inovação, who promoted this work through CMU Portugal program and gave me all the necessary tools and support to do it. It was a marvelous experience to be able to execute part of this study in the United States and an opportunity to meet different realities.

I would also like to thank all my MSc colleagues: André Cruz, Carlos Lopes Pereira, Nuno Antunes, Nuno Medeiros, Paulo Ferreira, Rui Martins and Vitor Leitão. They made this 16 months feel less terrifying, although they were themselves also buried in lots of work. Thank you, guys! We'll keep in touch. I also want to thank all the teachers and TA's, especially those who were closer to me, at FCUL, in Lisbon:

Professor António Casimiro, Professor Paulo Veríssimo, Professor José Rufino, Professor Nuno Neves, Professor Marcelo Pasin, Professor Alysson Bessani, and Professor Miguel Correia, but also João Craveiro and, of course, Tiago Carvalho.

My thank you goes also to my colleagues at PT Inovação: Luís Cortesão, José Bonet, Manuel Aguiar, Jorge Gonçalves, António Santos, Raúl Costa, Joel Ferreira and José Eduardo Rocha, Cristina Cabral, Lara Brito (thank you for everything, Lara!), Hugo Cabral, João Bica Osório, and many, many, many others! To all of them, thank you for your friendship!

Of course, I cannot forget my family, who sacrificed themselves for me during these 13 months living 250 km away from home, and another 3 months in another continent. They took care of my house, finances and cats, drove me back and forth and were irreplaceable. Thanks, Mom, Olga, Pedro and Sara.

Finally, my last ‘obrigado’ goes to Silkshadow, Fernando Almeida's rock band. Their debut album was my soundtrack for these final weeks, and gave me the speed and energy I was already missing. Good work, guys!

To my family, for all the support.

Resumo

1. Introdução

O trabalho que se apresenta propõe uma solução de segurança para um aspeto do sistema de mobile-Finance da Portugal Telecom (PT). A proponente, PT Inovação, sugeriu que este estudo se centrasse na segurança do canal de acesso USSD da sua plataforma de gestão de transações financeiras, Financial Transaction Manager (FTM). A Figure 1 mostra a arquitetura funcional do FTM.

Os participantes nas transações analisadas são:

- O Utilizador: usará a plataforma FTM para depositar e levantar dinheiro na sua conta do sistema, efetuar pagamentos, compras e transferências, etc.
- O Agente: representa uma instituição e tem a função de registar Utilizadores e aceitar depósitos e levantamentos destes.
- O FTM: coordena e controla todas as transações, regista Agentes e Utilizadores, valida e consolida os movimentos, etc.

Sendo o Agente essencial na conversão de valores monetários entre o FTM e o mundo real, recebendo e entregando dinheiro ao Utilizador, considera-se que é um participante suscetível de ser tentado a executar ou a sofrer ações maliciosas. Por esse motivo a nossa proposta centra-se essencialmente na segurança das comunicações USSD entre o Agente e o FTM.

O protocolo USSD é um protocolo que permite enviar mensagens de texto no canal de sinalização GSM, tal como o SMS, mas contrariamente a este não é armazenado até ser enviado. Encontra-se disponível em todos os equipamentos GSM, é bidirecional e de tempo-real, e apresenta tempos de resposta curtos relativamente ao SMS. É por isso um forte candidato como canal de acesso para algumas soluções de m-Finance fornecidas pela PT, mas o facto de não oferecer segurança constituiu um problema.

Este trabalho faz um estudo do modelo de ataque do protocolo financeiro já existente, propõe uma solução que garante segurança nas comunicações USSD entre o Agente e o FTM e sugere medidas que aumentam o grau de segurança do sistema em geral.

Para tal, partimos dos seguintes pressupostos:

- O Agente já se encontra registado na plataforma FTM.
- O Utilizador regista-se apenas com um Agente.
- O Utilizador não tem conhecimento prévio sobre o Agente antes de se registar com ele.

O protocolo já existente na plataforma define alguns passos que os participantes devem seguir e que se encontram descritos em Figure 3, Figure 4 e Figure 5.

2. Modelo de Ataque

O modelo de ataque para o protocolo de comunicações analisado pressupõe que o Agente tem um dispositivo móvel com alguma capacidade de processamento, se encontra certificado pelo FTM e que as comunicações entre o Agente e o FTM são encriptadas. Os ataques possíveis podem ser classificados como:

- Ataques ao equipamento móvel: inclui perda ou roubo do dispositivo, presença de aplicações maliciosas no dispositivo, acesso ao dispositivo por outros canais não referenciados e não autorizados, utilizador malicioso, clonagem do cartão SIM.
- Ataques à rede: incluindo a personificação do operador de rede, personificação do FTM e personificação do Agente.
- Ataques ao protocolo e aos algoritmos criptográficos: incluindo ataques de repetição ou reinjecção, MSISDN spoofing, ataques de homem-no-meio, ataques em conluio e ataques criptográficos.
- Ataques ao operador da rede: inclui perda, atraso ou corrupção de mensagens, inundação de mensagens (flooding) e modificação de mensagens.

A mitigação de todos os ataques referidos pode não ser possível e o nível de segurança alcançado dependerá da capacidade de processamento do equipamento móvel.

Do lado do Utilizador a mitigação de alguns dos ataques tem que ter em consideração o tipo de equipamento que este possui, e que classificamos do seguinte modo:

- Categoria 1: não é capaz de executar operações criptográficas ou não permite aplicações SIM.
- Categoria 2: permite aplicações SIM e possui alguma capacidade de processamento.
- Categoria 3: smartphones ou outros, programáveis e personalizáveis, com componentes TCB.

Os ataques de repetição ou reinjeção podem ser evitados através da introdução de valores aleatórios ou de utilização única nas mensagens. Os ataques do tipo homem-no-meio podem ser mitigados através de confirmações fora-da-banda ou multicaminho e os ataques de negação de serviço pela exaustão de recursos do FTM podem ser prevenidos através de puzzles ou desafios ao Agente ou Utilizador, exigindo interação humana.

A confidencialidade das transações entre o Utilizador e o FTM não está completamente resolvida uma vez que não impomos ao Utilizador a necessidade de encriptar a informação. As comunicações USSD serão seguras desde que encriptadas com chaves de curta duração e tal não é possível em equipamentos de categoria 1. No entanto, mesmo nestes equipamentos é possível garantir algum grau de segurança pelo uso de desafios, puzzles, e/ou confirmações fora-de-banda ou com multicaminho. Propomos o uso de confirmações fora-de-banda baseadas em segredos conhecidos apenas do FTM e do interveniente (Agente ou Utilizador),

como um código secreto ou uma matriz de códigos. O FTM deverá forçar o interveniente a mudar estes códigos periodicamente.

No caso do Agente, propomos encriptar sempre as comunicações com o FTM usando chaves de curta duração que são geradas apenas se o Agente conseguir enviar corretamente um código secreto, conhecido só dele e do FTM. Este procedimento evitará que um atacante consiga personificar o Agente. Um Agente malicioso só conseguirá ter sucesso relativo e receber o dinheiro do Utilizador, não o registando no FTM, se simultaneamente for capaz de executar um ataque de homem no meio intercetando e modificando a mensagem de confirmação que o FTM envia ao Utilizador. Será eventualmente detetado pelo Utilizador por comparação de saldos da conta e este pode apresentar queixa do Agente desonesto.

3. Protocolo

O protocolo de segurança proposto, USSL/UTLS, baseia-se em SSL/TLS tendo sido modificado para poder ser transmitido numa mensagem USSD. O tamanho máximo da mensagem USSD é 182 bytes com um formato apresentado na Figure 6.

Partindo do pressuposto de que o FTM tem um conjunto de chaves pública/privada, e que o Agente conhece a chave pública do FTM, o Agente inicia uma sessão USSL/UTLS usando um handshake semelhante ao do SSL/TLS. O formato das tramas SSL/TLS está representado na Figure 7 e Figure 8, e calcula o tamanho em bytes de cada uma das mensagens trocadas. Este estudo está traduzido nas tabelas Table 2, Table 3, Table 4, Table 5, Table 6, Table 7 e Table 8.

Após a fase de autenticação USSL/UTLS são geradas as chaves MAC e de encriptação para o FTM e Agente. A segunda fase do processo requer que o Agente confirme a sua identificação através de um código pessoal e está representada na Table 9.

Propomos a adição de alguns passos ao processo de registo do Utilizador: o Agente entrega ao Utilizador um envelope selado com um identificador exterior, contendo um código secreto ou um conjunto de códigos secretos, informa o FTM do identificador entregue e o FTM informa o Utilizador desse facto. Este envelope contém o código, ou códigos secretos partilhados apenas entre o Utilizador e o FTM e que serão usados para confirmar operações.

O procedimento para depósito de uma quantia pelo Utilizador não se altera, mas o procedimento para levantamento deve ser modificado, evitando que o Utilizador indique o seu PIN na primeira mensagem, e confirmando a operação de levantamento com a introdução do código secreto partilhado com o FTM.

4. Análise de Desempenho e Discussão

O desempenho do sistema está essencialmente dependente das capacidades do dispositivo móvel, do servidor FTM, do número de mensagens trocadas, do tempo de transmissão USSD e do tempo de processamento USSD.

Não é possível reduzir o número de mensagens de handshake sem comprometer a segurança, mas é possível limitar o número de vezes que as chaves temporárias são criadas associando-as a um temporizador que as invalida ao fim de algum tempo.

O timeout das mensagens USSD também pode ser ajustado para cada mensagem usando um valor mínimo possível. Na fase de autenticação do Agente, com um timeout de 10 segundos por mensagem, teremos um máximo de 100 segundos para concluir a autenticação.

Existem limitações na implementação de protocolos de segurança em dispositivos de categoria 1 e embora o protocolo proposto coloque o máximo de esforço no FTM, algumas operações têm que ser executadas no dispositivo.

A nossa proposta, por se basear no protocolo SSL/TLS, permite fazer alterações às bibliotecas criptográficas usadas e inclusivamente suportar várias versões do protocolo, possibilitando a coexistência de diferentes níveis de segurança.

5. Conclusões

O esquema de segurança sobre o canal de acesso USSD do sistema m-Finance da PT apresenta várias vulnerabilidades. Este trabalho propõe algumas modificações de forma a garantir a segurança das comunicações, especialmente entre os Agentes e o FTM. Sugerimos o uso de mecanismos que aumentam a segurança de forma geral, bem como a adaptação do protocolo SSL/TLS de forma a ser possível usá-lo nas transmissões USSD. A esse protocolo chamámos USSL/UTLS. Estamos convictos que a nossa proposta para o uso de USSL/UTLS é viável em equipamentos de categoria 2 e 3, e que aumentará a segurança nas comunicações entre o Agente e a plataforma FTM.

Na generalidade, a nossa proposta requer um mínimo de intervenção por parte do utilizador e apresenta medidas de segurança adicionais para autenticação do

Agente. A solução que advogamos assegura a confidencialidade, integridade e autenticidade das mensagens trocadas entre o Agente e o FTM. A combinação de PIN e código de confirmação do Agente e a chave pública do FTM asseguram igualmente o não-repúdio entre estes dois intervenientes.

Palavras-chave: USSD, SSL/TLS, m-finance, confidentiality, authenticity

Abstract

This work analyses an existing mobile-finance scheme at Portuguese PT Inovação, targeting users that do not have a bank account, and using the USSD communication channel to process financial transactions between three parties: the User, an Agent that represents, or acts on behalf of, an institution, but not necessarily a bank or a financial one, and the Financial Transaction Manager (FTM) that manages the Agent network, the Users and the transactions made.

We start by analyzing USSD communications: by itself it is not a secure communications channel, but it is available at every GSM device, allows for instant messaging services and is inter-operable, i.e. is not telecom dependent. Besides, it can run on commodity mobile phones, and requires practically no software download. From the user point of view, it resembles a normal text message and requires no special communications contract with the telecom operator other than the one that allows for sending text messages. It presents some security issues, namely, no authentication, no confidentiality, no integrity. We demonstrate that these issues can be solved through the use of end-to-end secure protocols on top of USSD in addition to other security mechanisms.

PT Inovação's m-finance scheme already implements a set of operations and financial transactions. We analyze the system's threat model and we propose a solution that will protect a specific communication path, namely, between the Agent and the FTM. We suggest the implementation of SSL/TLS over USSD, a lightweight version that we call USSL/UTLS. We demonstrate that it is feasible to implement such security mechanism on a USSD communication channel, and that it provides end-to-end security over the network communication path, at least if the devices present some processing capabilities. We propose some possible implementation paths, and conduct a brief performance analysis.

Keywords: USSD, SSL/TLS, m-finance, confidentiality, authenticity

Table of Contents

List of Tables

List of Figures and Illustrations

Glossary

2G	second generation of mobile cellular phones and mobile telecommunications standards.
3G	third generation of mobile cellular phones and mobile telecommunications standards.
CAMEL	**Customized Applications for Mobile Enhanced Logic**: a GSM 2+ feature that makes world wide support of Operator Specific Services possible.
ETSI	**European Telecommunications Standards Institute**: independent, non-profit, standardization organization in the telecommunications industry in Europe.
FTM	**Financial Transaction Manager**: the manager responsible for the control of financial transactions in PT Inovação's mobile finance system.
GSM	**Global System for Mobile Communications**: a standard set developed by ETSI to describe technologies for 2G digital cellular networks
HMAC	**Hash-based Message Authentication Code**: a mechanism for message authentication using cryptographic hash functions.
IMSI	**International Mobile Subscriber Identity**: a unique 15-digt code stored in the mobile phone SIM card, used to identify an individual user on a GSM network.
IN	**Intelligent Network**: a standard network architecture intended for fixed as well as mobile telecom networks, allowing operators to provide value-added services in besides standard telecom services.
MAC	**Message Authentication Code**: a piece of information used to authenticate a message.
MD5	**Message-Digest Algorithm**: an algorithm based on cryptographic hash functions.
MSISDN	**Mobile Station International Subscriber Directory Number**: a number uniquely identifying a subscription in a GSM or UMTS mobile network, corresponding to a phone number of the SIM card.
OTA	**Over the Air**: a technology used to communicate with, download applications to, and manage a SIM card without being connected physically to the card

PIN **Personal Identification Number**: a secret numeric password shared between a user and a system that can be used to authenticate the user to the system.

PT **Portugal Telecom, S.A.**: a telecommunications provider present in many countries, that provides a full range of fixed and mobile telecommunications services.

RSA **Rivest, Shamir and Adleman public key cryptographic algorithm**: an algorithm for public-key encryption.

SHA-1 **Secure Hash Algorithm 1**: and algorithm for computing a condensed representation of a message or a data file.

SIM **Subscriber Identification Module**: an integrated circuit that securely stores the IMSI and the related key used to identify and authenticate subscriber on mobile telephony devices. It is kept on a removable SIM card that can be transferred between different mobile devices.

SMS **Short Message Service**: a text messaging service component of phone, web or mobile communications systems that uses a standardized communications protocol that allow the exchange of short text messages between devices.

SMS-C **Short Message Service Center**: a network element responsible for handling the SMS operations of a wireless network.

SSL/TLS **Secure Sockets Layer/Transport Layer Security protocols**: cryptographic protocols that provide communication security over the internet, designed to prevent eavesdropping and tampering of the information.

STK **SIM Tool Kit**: a standard of the GSM system that enables the SIM to initiate actions which can be used for various value-added services.

TTP **Trusted Third Party**: A security authority, or its agent, trusted by other entities with respect to security related activities.

UICC **Universal Integrated Circuit Card**: the smart card used in mobile terminals in GSM and UMTS networks, which ensures the integrity and security of all kinds of data.

UMTS **Universal Mobile Telecommunications System**: third generation mobile cellular technology for networks based on the GSM standard.

USAT **USIM Application Toolkit**: the equivalent of STK for 3G networks.

USIM **Universal Subscriber Identity Module**: a software application for UMTS mobile phones that runs on a UICC, which is inserted in a 3G mobile phone.

USSD **Unstructured Supplementary Services Data**: a protocol built into the GSM standard used by GSM cellular phones to communicate with the service provider's servers. It is available to all devices since 2G GSM.

WAP **Wireless Application Protocol**: an open, global specification that allows mobile users with wireless devices to easily access information over a mobile wireless network.

Chapter 1

Introduction

Securing mobile-Finance (m-Finance), the topic of this work, was proposed by PT Inovação, the R&D branch of the Portuguese communications operator Portugal Telecom, PT.

In this chapter, we present the motivation for the work, and our main goals and achievements. At the end of the chapter, we present the organization of the rest of the document.

1.1 Motivation

Mobile-finance (e.g., mobile-payment systems, contactless or remote, branchless banking models, airtime transfers, etc) introduces several security issues that need to be urgently addressed by all the actors in this new and thriving business area.

Remote payments and remittances, deposits via ATM, the operator's financial transaction platform, and the interface with banks and other financial institutions deal with many security requirements concerning the type of communication (USSD, SMS, etc), the infrastructure responsible for processing and storing financial transactions, and the communication between each participant.

In the case of emerging economies there is a special interest in domestic remittances and international remittances through financial brokers (like Western

Union, Belgacom ICS, etc). Usually the communication is done in IPSec or SSL, but when considering the case of emerging economies, like Brazil or Morocco, other types may need to be available. In this type of countries economy is growing very rapidly but communications infrastructure is not yet fully developed and many of its citizens have a low literacy level and low incomes. Modern services that are based on a good telecommunications infrastructure, like banking, remote commerce, etc, are not evenly spread throughout the country, and some regions may be more developed than others. The necessity for other types of secure communications comes from the existence of a large quantity of individuals with low-literacy, no bank account, no access to top Internet technologies (like value-added services, data services or broadband services), but owning mobile devices, although not necessarily top technology devices.

With this work, PT intends to have analyzed the security mechanisms needed for the some usage scenarios on the m-Finance system, having in mind the constantly changing technological platforms and usability needs on the markets PT intend to address. Therefore PT needs to have one or more prototypes developed to test the different security solution approaches available for several cultural and economic development realities. Such realities include developed countries, like most European countries, in which potential users have easy access to internet technologies and top mobile devices, as well as emerging economies, like Brazil, Morocco and others, where many potential users have limited financial capacities, little or no access to internet technologies, and yet own a personal, although limited, mobile phone. These different solutions approaches should allow for the exploration of several types of secure communication channels that can be used for a financial transaction.

The company already supports a mobile financial platform, the Financial Transaction Manager (FTM), briefly described in the following Figure 1 depicturing its functional architecture. The FTM platform is part of PT largest m-Finance system that is meant to provide m-Finance services to clients: banks, financial institutions, brokers, commerce retailers, etc.

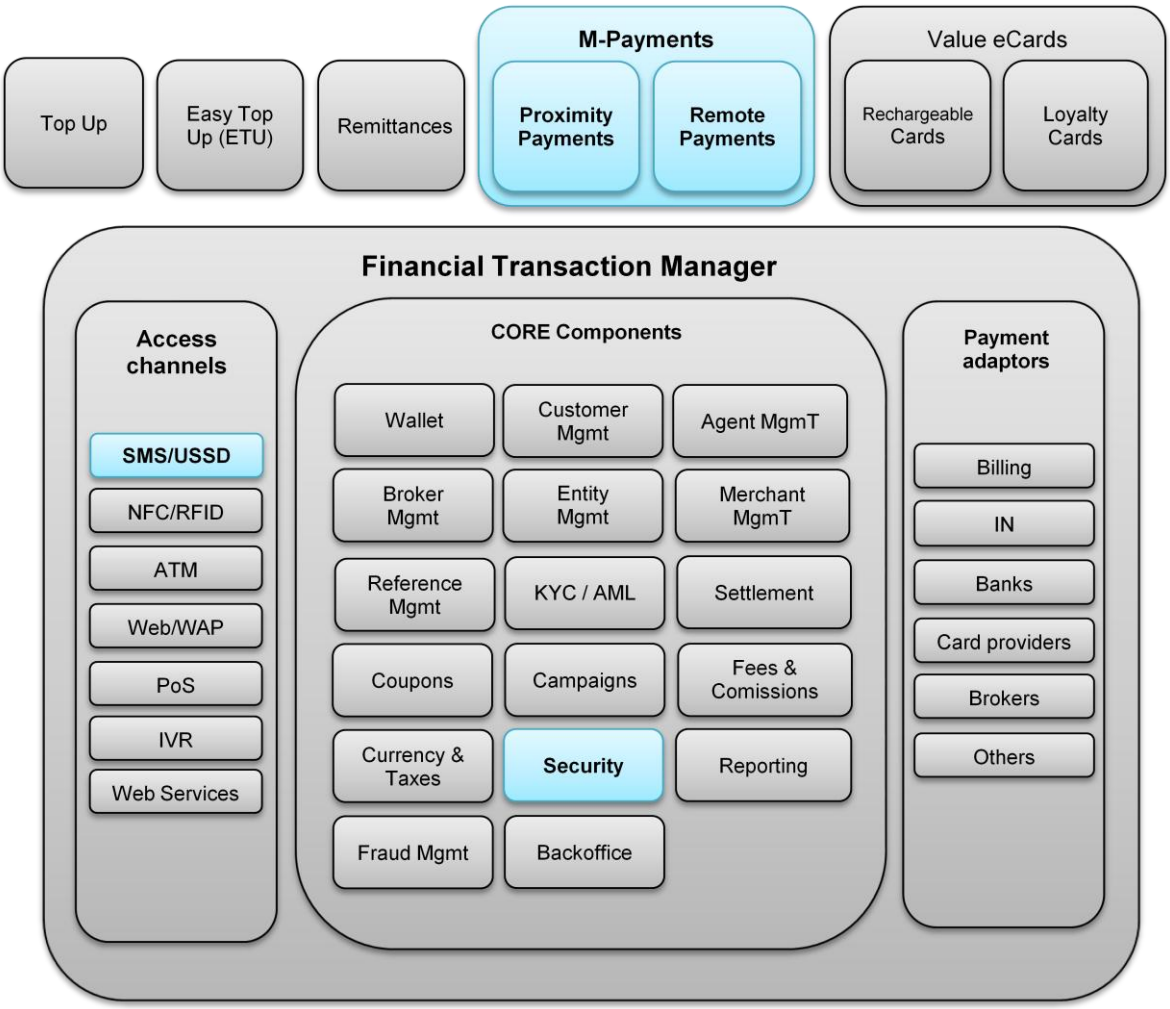

Figure 1 – FTM platform functional architecture: possible communications channels and system core components. Highlighted are the SMS/USSD access channel and the security core component covered by this work.

This architecture is composed of three main blocks, responsible for providing the access channels, payment adaptors and the system core components. The focus of this work is the Security Core Component and the SMS/USSD[1] access channel.

The existing m-Finance platform already establishes a number of participants involved in several types of transactions. These transactions include the ones the User can execute if he wishes to use the e-money stored in his m-Finance account to do some financial operations, like

- pay for some products or services at a m-Finance valid merchant;
- transfer some amount from his m-Finance account to his bank account;
- transfer some amount from his m-Finance account to another m-Finance user;
- recharge his m-Finance account with some amount by instructing a transfer from his bank account to his m-Finance account;
- recharge his m-Finance account with some amount by going to a m-Finance Agent or broker in order to convert some cash into m-Finance account;
- convert some amount stored in his m-Finance account into current coins and banknotes.

The m-Finance system presents some properties. Namely, it is a complement to traditional payment methods, it is flexible, convenient and ubiquitous. It is also a facilitator for a diversity of m-commerce services, like m-ticketing, m-retail, m-banking, etc. Although this payment system can be connected to bank account payments, one of its advantages is that this is not a requirement, and the system can

[1] Although the access channel includes SMS, we will not cover that technology in this work.

be used by the segment with no bank accounts as well. This is achieved through the use of Agents, certified by the FTM platform, that take the necessary steps to convert money to or from e-money.

An Agent is a person that represents, or acts on behalf of, an institution (but not necessarily a bank or a financial one), registering Users to the system, receiving money from a User when he commands a deposit to his system account or delivering money to the User when he executes a withdraw operation, among others. From time to time, Agents will deliver the received money to the institution they represent, or receive more money from it. Agents can be a network operator authorized dealer, other retailers like petrol stations, distributors supermarkets, etc, selected banks and other micro-Finance institutions, and basically anyone who the m-finance system agrees that can act on its behalf.

Agents, like Users, can contact the FTM platform through a mobile device using any of the available access channels. In this work we assume that the chosen channel is USSD. This is not an uncommon choice because USSD is possible in every SMS-capable GSM device (virtually, all GSM devices) and requires no special communications contract with the telecom operator other than the one that allows for sending text messages, which makes it less expensive than other access channels requiring explicit data communication, like Web/WAP.

On the User side, the co-existence of different mobile devices generations is a problem we will have to address. We categorize the User's devices into 3 categories:

- Category 1: Legacy devices: devices are not able to execute cryptographic functions due to their limited processing capabilities. We also include in this group those devices that do not support SIM applications;
- Category 2: Medium capable devices: in this category we include devices that have some processing capabilities and that allow for SIM application install
- Category 3: Top devices: in this category we included smartphones and similar devices. They allow for custom application installations and they provide some type of TCB components.

We will analyze later the minimum requirements for the Agent's device in order to assure a certain security level.

The role of this Agent is depicted in Figure 2, where we can see the steps taken by a User that whishes to deposit or withdraw a certain amount. In this figure we can also see the role of other participants, like the bank and the Intelligent Network (IN) platform, the platform that is responsible for providing the financial service.

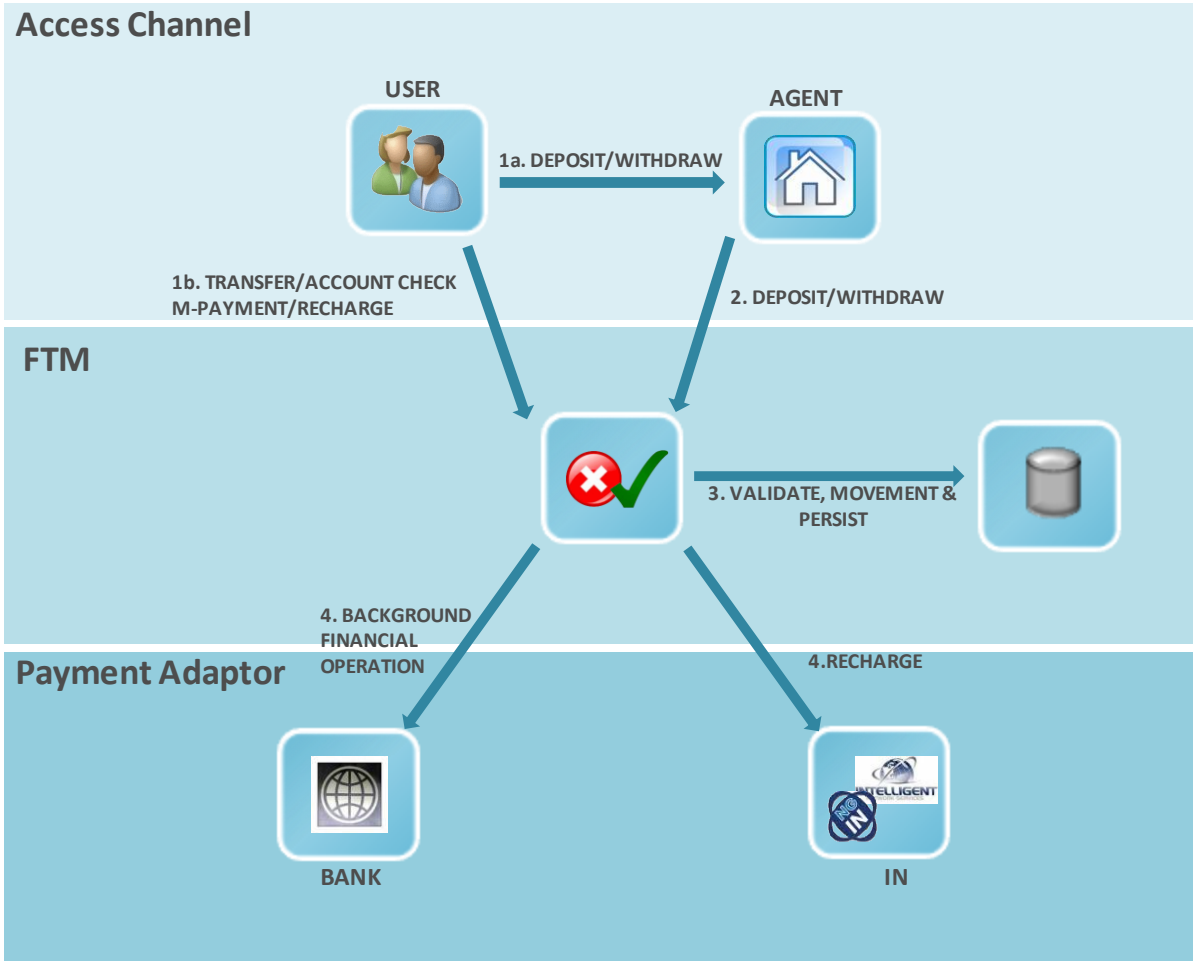

Figure 2 – m-Finance: Remittances communications sequence. The User can either directly contact the FTM to transfer, pay or recharge his mobile phone account or go to an Agent to deposit or withdraw money. In this case, the Agent requires a deposit/withdraw to the FTM in the User account and the FTM validates and registers the movement at its database and informs both Agent and User of its success or failure. Depending on the operation the User commanded, FTM will sinchronize the financial monetary operations with a bank, broker or other, or will invoke the IN platform to process recharges in the User's account, for instance.

In Step 1a, if the User wants to withdraw or deposit some money, he goes to an Agent and gives him the required information to process the transaction. Next, in Step 2, the Agent requires deposits or withdraws to the FTM in the User account. Optionally, the User can contact the FTM directly (through a preferred access channel) to transfer, pay or top-up his mobile phone account, not involving an Agent in this process. This is depicted in Step 1b, in Figure 2. The FTM then takes Step 3,

to validate data and to register events and movements into its database. The User or Agent is informed of the success or failure of this operation. Finally, FTM takes the appropriated Step 4. In this case, the FTM may synchronize financial monetary operations with a bank or an international broker, or may invoke IN to process a recharge done with e-money.

The role of the Agent is a key one, most especially to those without bank accounts. This is a major feature to certain economies, where bank agencies are not available at every corner and most people do not necessarily have a bank account, but most certainly possess a mobile phone. The Agent is the key to convert money into e-money and vice-versa. Once the User has an m-Finance account, he is able to transfer e-money to or from another User, and buy services and products that would otherwise be unavailable.

All operations may be executed through an available access channel, as depicted in Figure 1. PT Inovação proposed that this study would be done on the USSD communication channel.

Presently, all GSM mobile phones have SMS/USSD capabilities but unlike SMS, USSD [1][2][3] is a session-oriented service and offers a real-time connection during a session, which allows for bi-directional exchange. The USSD session is defined as the sequence of messages being exchanged between the two parties. The real-time session is initiated between the mobile user and the USSD application platform when the service is invoked, allowing data to be sent back and forth between the mobile user and the USSD application platform until the service is completed.

Besides this property, USSD does not need to store and forward a message, which means it does not need a SMS-Center (SMS-C) to store any arriving message until being able to send it to the destiny or give up due to message timeout, the procedure that SMS follows. Hence USSD is much faster than a SMS message. One implication is that the device needs to be on for the messages to be delivered and read. This makes USSD more responsive than SMS, enforced by the fact that USSD messages present smaller timeouts. In case of failure in delivering a USSD message, the sender will be notified of the fact.

Another interesting property is that USSD Phase 2 (and later) [2][3][4] supports both device-initiated (push) operations as well as network-initiated (pull) operations. Comparing the communications model, SMS messages usually are sent between two peers, where the sender actually sends the message to the SMS-C, indicating that it is intended to a destiny peer. The SMS-C is responsible for storing the message and forward it to the destiny as soon as possible, and deleting it after it was successfully sent or its timeout expired. USSD messages may be sent to the network operator's USSD application platform where they trigger some kind of action, like executing some task or getting some information from the system concerning the sender (e.g. getting information on the pre-paid mobile account balance) or are sent by the USSD application to the mobile phone with information to the phone's owner (for instance, alerting for the need to recharge a pre-paid service in order to continue to use it without service interruption), but are not stored. They are either almost immediately delivered/executed or not, and the sender is informed of the success or failure of it.

USSD is supported by WAP, SIM Application Toolkit and CAMEL.A USSD message is limited to a maximum of 182 alphanumeric characters. Typically, the message starts with an asterisk (*) and terminates with the number sign (#). Between these two delimiters there are some digits and/or alphanumeric characters, which may be separated by asterisks (*), to identify the action to be executed and the necessary parameters, when applicable.

The user can directly create a USSD message by simply typing the proper alphanumeric sequence and press “call” to send the message, after which it will receive an answer from the USSD application. Another form of starting an USSD session is for the USSD service on the network operator’s USSD application server to contact the user so that he will have to introduce some values (again, alphanumeric characters) and answer back. In both cases, the USSD application may even offer some menus to the user and drive him through several options in order to provide a more complex interaction. The response from the application server usually takes a few seconds.

The availability characteristics of USSD, as well as its universality – it is present in all GSM mobile phones – make it an interest candidate for a possible communications channel, and this work will study the possibility of implementing secure communications on a USSD channel.

A USSD message will put as much load to the network as a SMS message, but SMS presents higher message timeouts that are not suitable for a communication scheme where we have message flows with several steps. To its favor, USSD messages flow are aggregated in sessions and can be controlled. As for USSD service pricing, it is

a contract agreement between the network operator and USSD service application provider that defines the cost of each USSD services access.

The main aspects to be studied are highlighted in Figure 1: the security of mobile payments using USSD access channel as a communication channel.

1.2 Goals

Having in mind that PT's m-Finance services partially target mobile phone users of emerging economies without bank accounts we can list three main challenges for this work:

1 - Find a protocol that is able be used on a USSD channel. This includes coping with USSD message size limitations but also we have to deal with performance limitations.

2 - Present some solutions suitable for mobile devices with reduced computational performance and capabilities. This can only be achieved with some limitations due to the computational demands of the cryptographic and hash calculations. In the case of category 3 devices (smartphones and similar), this is no big challenge, but we do not expect every service user to own such a device. Instead, we expect users to possess category 2 devices, and maybe some of them will have older, category 1 devices. We will see later how this will impact on our proposal.

3 - Present a solution that is usable and requires minimum user intervention, but at the same time offers some assurance that the person using the application is the expected one, and is not being impersonated.

As referred above, the role of the Agent is a key one. He is one possible participant responsible to convert money to e-money and vice-versa. His importance is even greater to Users with no bank accounts that will have limited options to process this conversion. One of the goals of this study is to assure a certain security level in Step 1a and Step 2 of Figure 2. The fact that the Agent deals with money directly imposes as a requirement a certain degree of confidence during these steps.

Finally, the presented solution has to be able to keep track of the operations executed and who ordered them, in order to provide audit information in case the system is tampered.

1.3 Contributions

This work will contribute with a proposal for secure communications over a USSD channel. The way we intend to assure security properties is by developing a lightweight SSL/TLS protocol, capable of being transmitted over a communications channel with limited bandwidth as well as limited message length. Our target is the population segment that do not own a bank account, do not have access to broadband internet access services and do not possess a high literacy level, but do possess a mobile device, possibly a top device (a smartphone, for example) but most certainly a lower level device, with smaller computational capabilities. We will see next how this will affect our proposal.

Because this is not an isolated or theoretic analysis of a security solution for the USSD channel, but rather is intended to make use of it in its m-finance service, we

will also analyze the existing protocol and will propose security improvements whenever necessary.

1.4 Document Structure

This document is organized in the following way: in Chapter 2 present some related work to our solution; in Chapter 3 we describe the existing protocol and in Chapter 4 we present a threat model; Chapter 5 describes our solution and Chapter 6 presents a brief survey of the implementation aspects; in Chapter 7 we conduct a performance analysis of the protocol while in Chapter 8 we present some more discussion and analysis of the solution we proposed; Chapter 9 is dedicated to some hints for future work and finally, Chapter 10 is dedicated to a summary and conclusion.

Chapter 2

Related Work

There is plenty of literature related to m-finance and to USSD value-added applications on mobile transactions. In fact, almost every country and or operator has a customized solution for m-finance. Next we present only a few of the related works we found out to be more relevant to our own work.

In [6], Kumar, Martin and O'Neill describe the India experience in m-commerce, focusing on "specific India ethnographic characteristics". Their work is a survey of a possible implementation of m-commerce services supported in SMS, like paying for day-to-day services or goods, for instance, buying food or a bus ticket, or cash transfer between people, and taking advantages of the widespread penetration of mobile phones. Like this present work, they target the poorest section of the population, most of them without bank accounts and illiterate. However, the India specific environment presents to the authors some increased challenges, like the inhabitant's habit of sharing sensitive information of their devices, for instance PIN numbers. Even so, the study focus on the importance of the number of human interaction required to complete a transaction, the system's swift notifications to both participants of the transaction being held as well the both parties correct identification. The study also refers to the importance of m-Finance for financial inclusion of the populations of less developed areas. These aspects are also important to our work, and we will take them in account for our proposal.

Also in [8], Karunanayake, De Zoysa and Muftic, present a study referring the major benefits of m-commerce, specially focused on the no-bank-account and low-literacy segment of rural areas of developing countries. By the time they presented their proposal, it was being implemented in a rural bank in Sri-Lanka. This solution is also implemented over secure SMS and uses random values and PIN numbers as well as symmetric key cryptography. Although SMS is also transmitted on GSM's signaling channel, it presents some other characteristics that are unique to this technology, like the need to store the messages in a SMS-Center (SMS-C) until being able to forward them to the destiny.

McKitterick and Dowling [9] present a brief resume of some mobile payment technologies. Although this document is a study from 2003 and does not cover more recent technologies nor the newest trend on smartphones and their capabilities, it describes several technologies that are still at use. More interestingly, the authors present the case of the Spanish MobyPay, a service that used USSD to process financial transactions. This service was associated to credit or debit cards and thus was targeted to users with bank accounts. This is not our case, as we do not want to impose the need for a user's bank account association in our solution.

More recently, Panjwani [12] presented a study on branchless banking systems for developing countries, demonstrating that most of the services offered rely largely on network-layer services for securing transactions and that they are not end-to-end secure. The author a model that builds security at the application layer and provide recommendations for solutions based on his model. His proposal is not specific to any kind of communication channel and bases security in PIN combined with one-time passwords and digital signatures.

In [13], the authors explore user authentication schemes for m-banking that are present in the developing world. They analyze an authentication scheme based on PINs and printed codebooks for authenticating users that is used by EKO, an Indian mobile banking service provider, and propose an improvement to that scheme that is a variant of one-time pad. However, their proposal requires the user to modify his PIN (by adding a printed nonce) before a transaction is conducted. Although this study is meant to propose a solution for a weaker schemed used by the previously referred company, it is interesting in the sense that the result is a one-time user signature that is used to authenticate the user to the EKO, and is sent in the clear in a USSD message.

Other literature can be found referring the practical implementations of solutions for m-finance, of which M-PESA in Kenya is an example. Most of these implementations are based on SMS communications and only provide minimal security, usually via PINs. When we researched related work on secure USSD communications, we confirmed the diversity of solutions available, but either they do not provide all the security properties we are aiming (namely authenticity, confidentiality, integrity, non-repudiation) or they use other access channels, like WAP that provides its own security mechanisms.

Chapter 3

Description

As stated before, in this work we will only address the case where a User interacts with an Agent, the representative of the institution that supports the m-Finance service and acts on its behalf, like depicted in Figure 2. We make some assumptions for this interaction:

- The Agent is hard-registered to the FTM platform, which means that someone at the backoffice introduced into the system all the necessary data to validate an Agent. This registration is assumed to be correct and is outside the scope of this thesis.
- The User is registered with an Agent he chooses and becomes associated with that Agent. After the User is registered to a certain Agent, he cannot go to another Agent. One of the reasons to limit the number of Agents to one per User is to simply the sending of notification messages in the transaction. More details will be provided when message flow is described;
- The User has no previous knowledge of the Agent, i.e., the User has no guarantees that he can trust an Agent he has not previously met or acknowledged. This means that the User is not able to tell in the first meeting if the Agent is a legitimate certified Agent or someone impersonating an Agent. Yet, we will show that the User will have some certainty degree after the first meeting with the Agent. This assurance is important because the User will have to trust the Agent in order to execute

some operations that involve the User's money. This also means that the User does not possess any special information, like certificate keys or signature verification keys before meeting an Agent for the first time.

The User needs to take some steps in order to be registered in the existing PT's m-Finance system and to execute some operations on the FTM platform.

PT Inovação proposed some operations for the normal usage of the m-Finance system. Next we describe three of these basic operations, decomposing its steps: User registry and activation, User deposit and User withdrawal. These operations are already implemented in the FTM platform and our next description does not take into account security aspects but only the basic information flow between the participants. Our security analysis will be done considering these three basic operations. In Chapter 5 we will propose some security modifications to the exiting basic operations and in Chapter 8 we will further analyze other security aspects that are not covered by the suggested modifications.

3.1 User Registry and Activation

The actual existing procedure that a User must undergo in order to be registered with an Agent is described in the following steps:

1. Presentation: User physically goes to an Agent and presents personal ID documents (ID Card, SSN/NIF, etc) and contacts (mobile phone number, email, etc). He shows also his device unique Mobile Subscriber Integrated Services Digital Network Number (MSISDN).

2. Registration: Agent sends a message to FTM with the User name and MSISDN.
3. Registration Acknowledge: FTM validates message and in case of success replies to the Agent:

```
User <MSISDN> was successfully registered.
```

4. Activation Request: FTM sends message to the User:

```
You have been successfully registered with PIN <PIN>. Please
confirm service activation sending the message "ACT <PIN>" to
number <LA>.
```

5. Activation: User must send the message to the number indicated with the PIN he received.
6. Activation Acknowledge: FTM validates the message and, if successful, replies to User:

```
Service activation was successful. Current PIN - <PIN>. For
security reasons, change your PIN as soon as possible.
```

Figure 3 depicts this message flow. The dashed line in the Registration step depicts a physical interaction, i.e., both parties are physically near each other, in the same room. The dotted line in the Activation step represents a text SMS message the User sends to the FTM to activate his account.

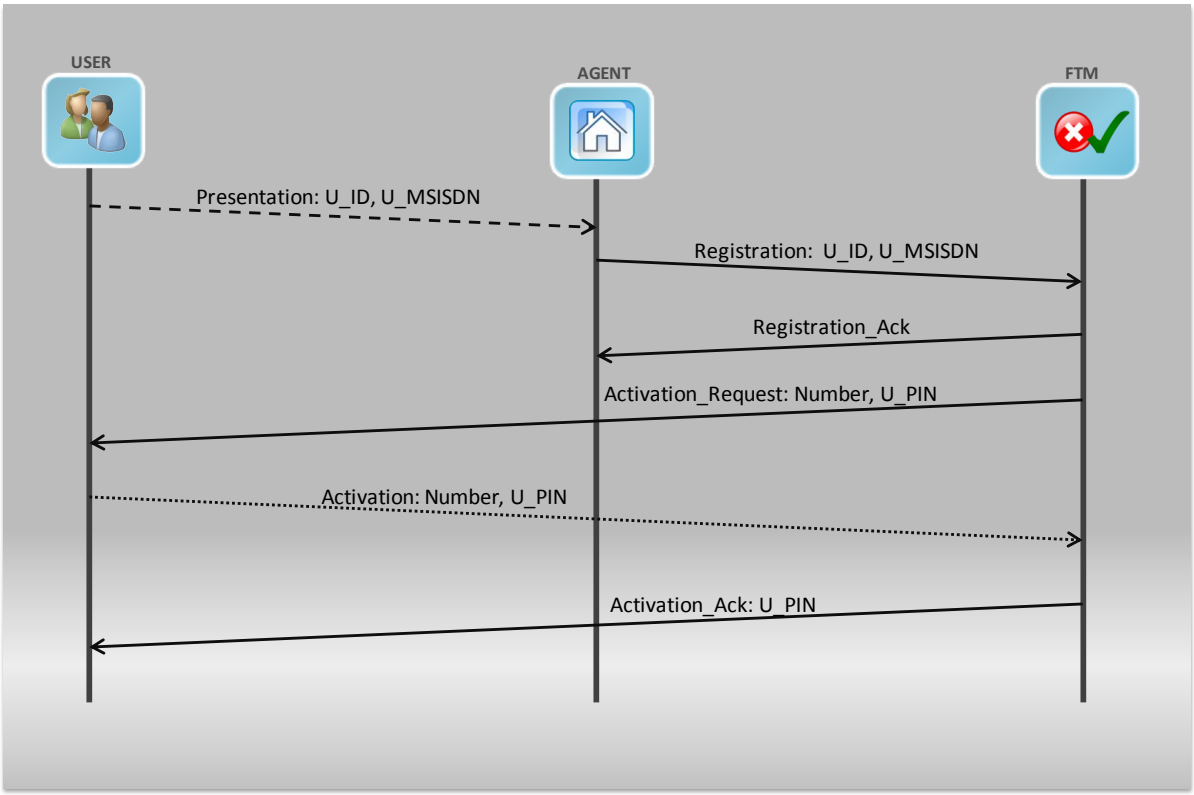

Figure 3 – User Registry and Activation. 1: User goes to the Agent and presents his identification and device MSISDN. 2: Agent sends a message to FTM with user ID and MSISDN; 3: FTM confirms User registration to Agent; 4: FTM sends User a message confirming his registration and indicating his PIN to confirm service registration and activation; 5: User sends a message to FTM confirming his registration; 6: FTM confirms User activation.

Note that any User can be registered at only one Agent, and it is the Agent's responsibility to provide the User's information to the FTM. The FTM is responsible for verifying that the message received came from a registered Agent and reject it if it does not. With this scheme, the fact that the User has no prior knowledge about the Agent's authenticity is minimized, although not resolved. The User will detect if the Agent is correctly registered with the FTM because in case it is not, the User will never receive the FTM Activation Request, during step 4. This does not prevent an adversary from eavesdropping the Agent's communications and obtain his MSISDN,

allowing for the Agent's impersonation. We will detail this type of attacks in the next chapter.

3.2 User Deposit

A User is able to deposit some money at his m-Finance account. He will deliver the money (coins, banknotes, checks or other valid form) to the Agent that informs the FTM of the amount received and the User's identification. The money is then converted to the same amount of e-money and credited to the User's account. The Agent will later deliver the money received to the institution that is supporting the m-Finance service but that procedure is not the scope of this work. The User's deposit transaction involves the following steps:

1. Presentation: User physically goes to the Agent and hands out the amount of money.
2. Deposit Request: Agent sends a message to the FTM, with the User MSISDN and the amount.
3. Deposit Acknowledge: FTM validates the message and, if correct, replies to the Agent:

   ```
   The <currency> <amount> was deposited successfully on the client
   <MSISDN> account.
   ```

4. Deposit Information: FTM sends a message to the User:

   ```
   <currency> <amount> were deposited in your account.
   ```

This flow is shown in Figure 4. Again, the dotted line denotes a physical proximity of both parties, able to interact with each other.

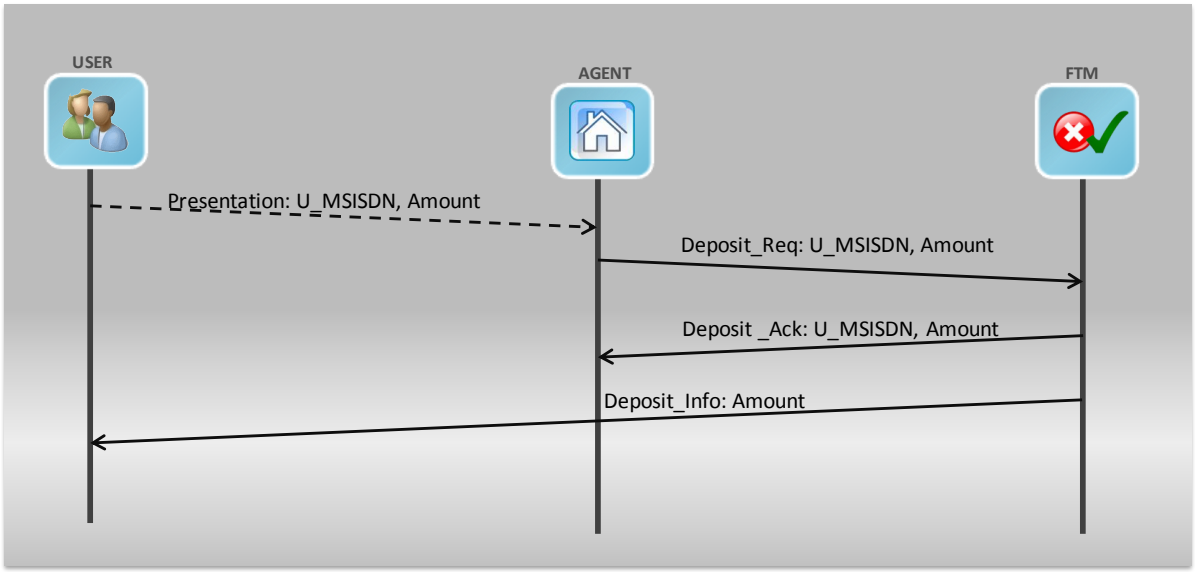

Figure 4 – User Deposit. 1: User goes to the Agent, presents him his MSISDN and the amount of money he wants to deposit at his account. 2: Agent receives the amount of money and sends a message to the FTM identifying the User and indicating the amount of money to deposit. 3: FTM notifies the Agent of the operation's success. 4: FTM notifies the User that an amount of money was deposited in his account.

Again, there are some vulnerabilities in this operation, for instance, it is not protected against a mistyping error of the User's MSISDN. This aspect will be discussed in Chapter 8.

3.3 User Withdrawal

Withdrawal operations are also possible. The User is able to reconvert the e-money at his m-Finance account into day-to-day money (coins and banknotes). The process

involves the User withdrawal request to the FTM and, in case that is possible, the FTM will issue a virtual voucher with a unique identifier that the User must present at his Agent in order to obtain the corresponding amount of money. The virtual voucher will identify the User and the amount withdrawn when presented to the Agent. The Agent himself has no knowledge of the amount of money, but he will send the voucher's unique identifier to the FTM, and the FTM will verify the voucher's validity and authorize the Agent to handout the money to the User. Notice that once the voucher is issued, it means the User has at least that amount in his account. Besides, the referred amount will be captivated until the withdrawal operation is completed or the voucher's validity is reached. The withdrawal procedure comprises the following steps:

1. User Withdrawal Request: User sends a message to the FTM, indicating his PIN and the amount to withdraw.
2. Withdrawal Authorization: FTM validates the message and, if successful, generates a voucher code, identifying specifically the withdraw of the amount requested, and sends a message to the User:

   ```
   Your request was successful. To withdraw <currency> <amount> from
   your account go to the Agent and show the code <voucher code>.
   ```

3. Presentation: User goes to the Agent and presents the voucher code.
4. Agent Withdrawal Request: Agent sends a message to FTM, with his own PIN and the voucher code.
5. Withdrawal Verification: FTM validates the message and, if successful, replies to the Agent:

```
Voucher <voucher code> was successfully withdrawn. Deliver
<currency><amount> to the client.
```

6. Withdrawal Information: FTM sends message to the User:

```
Agent will deliver you <currency><amount> that were withdrawn from
your account.
```

7. Withdrawal: Agent gives the requested amount to the User.

This is shown in Figure 5. Again, the dotted lines depict a physical interaction between the User and the Agent, where they are near each other. In the first physical interaction, the User shows the voucher code to the Agent, while in the last case, the Agent hands out the requested amount of money to the User.

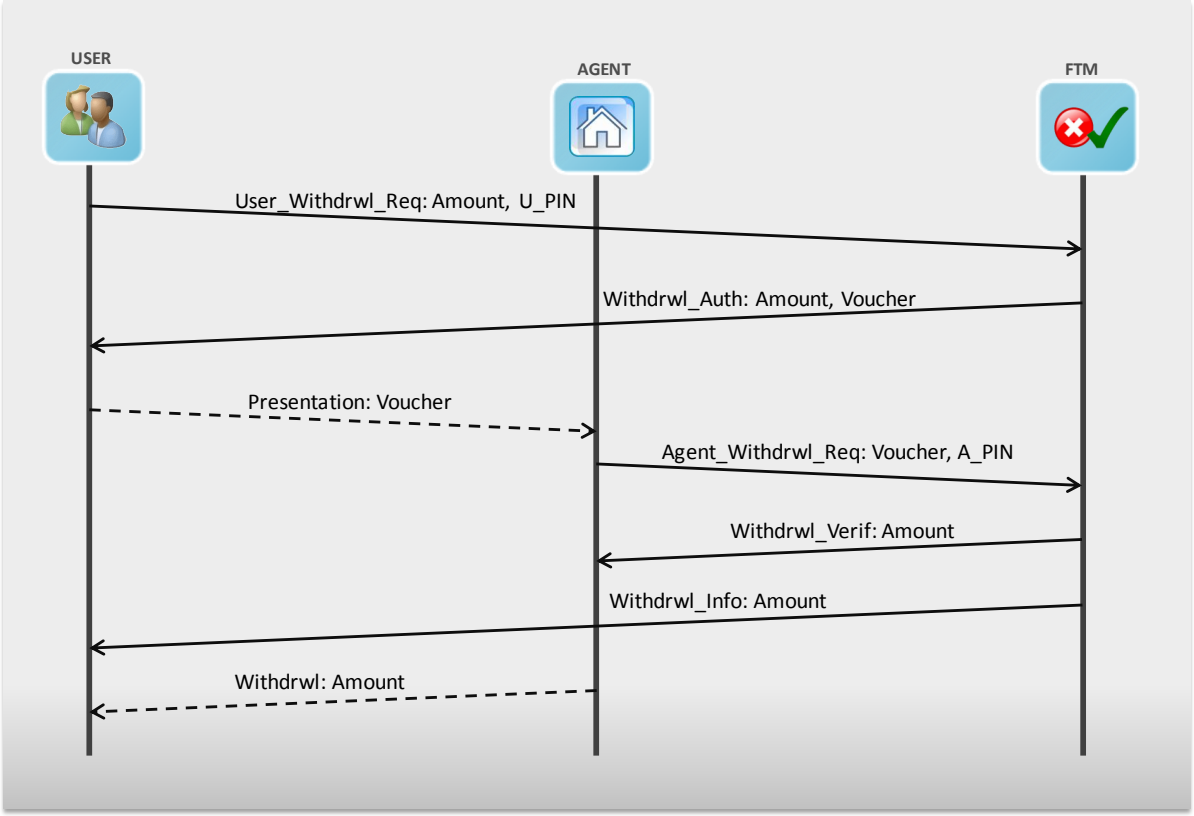

Figure 5 – User Withdrawal. 1: User sends to FTM a message requesting the withdrawal of a certain amount and indicating his PIN. 2: FTM validates the User and verifies if withdrawal is possible. In case so, FTM sends a virtual voucher code to the User. 3: User goes physically to the Agent's and shows him the voucher, requesting the money. 4: Agent sends the voucher to the FTM along with his own PIN. 5: FTM verifies the Agent's PIN and confirms the validity of the voucher. FTM sends a message to the Agent authorizing him to deliver the amount associated to the voucher code to the User. 6: The FTM sends a message to the User informing him of the withdrawal authorization from his account. 7: Agent hands out the amount of money to the User.

Again, there are some vulnerabilities in this procedure that will be discussed in Chapter 8, namely, disputes between the Agent and the User.

Chapter 4

Threat Model

In this chapter we will analyze the threat model. As we are proposing a protocol similar to SSL/TLS, but using a USSD channel, we need to analyze the USSD threat model.

There are several studies already on USSD communications. As stated before, because USSD is a GSM service and transfer of information occurs over signaling GSM channels, security characteristics for USSD are narrowly tied with GSM security. In [11], the authors present a thorough analysis of GSM vulnerabilities, as well as their proposed solutions for them. They refer to unilateral authentication and Man-in-the-Middle attacks, SIM card cloning, flaws in the GSM algorithms used, vulnerability to DoS attacks, the absence of integrity protection and vulnerabilities to replay attacks, among others. USSD is also mentioned because messages are not encrypted and secured in the GSM backbone. Their solutions range from secure algorithm implementations and secure ciphering algorithms, encryption of the GSM backbone traffic and end-to-end security. They describe some possible means of implementing this end-to-end security, pointing that most of them will require improved device processing capabilities.

Sarajlic and Omerasevic [5] present a study of some available m-Commerce access channels, including USSD. As they noted, while GSM intends to provide user authentication and encryption of communication in the radio interface, this does not apply to the signaling channels, and USSD is transmitted unencrypted. Besides,

attacks to GSM encrypted communications have been made, as was demonstrated by Barkan, Biham and Keller in [7], so we cannot rely on GSM's security properties.

Panjwani [12] presents some vulnerabilities of m-finance protocols for branchless banking systems. He identifies internal system vulnerabilities, addressing the possibility of insider attacks to the m-finance protocols. These insider attacks need to be mitigated as well as GSM vulnerabilities and the author considers external threats as well as internal threats to the protocol.

We present here a USSD threat model according to our system architecture and communications model. We make some assumptions and impose some requirements in our system, listed below:

1. We assume that the Agent has a mobile device with increased computational and processing capabilities. This is needed if we want to assure that the Agent is able to encrypt/decrypt all communication with the FTM.
2. We impose that the Agent is to be certified by the FTM so that we can be sure the Agent in not being impersonated and, at the same time, have non-repudiation properties. However, we propose that this certification is done out-of-band, in a multi-path security confirmation, not requiring a Certification Authority or a standard public/private key pair. In fact, we will not use a public key to certify the Agent to the FTM. With a certified Agent we can be assured that the FTM is not communicating with an impersonation version, and we can call him responsible for the transactions. We will use temporary keys to attest that we are dealing with the correct Agent.

3. We require that the communication between the Agent and the FTM must be encrypted. This requirement provides end-to-end encryption between the Agent and the FTM.

We identified five possible sources of vulnerabilities: mobile device attacks, network attacks, cryptographic and protocol attacks, operator attacks and social engineering attacks. We will describe in the next sections.

4.1 Mobile Device Attacks

Mobile devices present some vulnerabilities that cannot be ignored for the application we intend. The vulnerabilities list includes:

- Device stolen or lost. This is a very common situation. Devices are very small and portable, so it will be easy for someone to misplace it. Besides, modern devices are expensive and may be used as a status symbol, so they will attract thieves will a high probability. The case of the User's lost or stolen device may be critical if another person is able to access the device history list of previously dialed numbers. If the USSD messages are sent in a full format, i.e., the user includes all the needed information in the message sent, then it is possible to retrieve sensitive information out of the history list, most especially in the case of the withdrawal operation, where the User needs to send his PIN. The Agent's lost or stolen device is also very critical because the FTM only accepts Agent commands from registered Agents, and detects them through their MSISDN. The whole Agent – FTM communication is

currently only based on the Agent's correct MSISDN and no further Agent validation is required.

- Presence of malicious applications. Modern mobile devices present a high degree of customization, where users can decide what type of applications they want to install in their device. This presents some problems due to the higher possibility of installing malicious software, able to collect sensitive information from the device, or take control of it by making calls and or sending messages without the user's knowledge. The malicious application may be able to read all the previous USSD communications with the FTM system and replay the messages or impersonate the User or the Agent.
- Backdoor access to the device. This is somewhat related to the previous. A malicious application that intends to collect sensitive information most certainly will try to access communication ports out of the owner's knowledge, in order to transmit that information. On the other hand, other ports may be used to access and control the device even if there is not any kind of malicious application installed. This may happen, for example, if the user uses other wireless channels, like Bluetooth or WiFi, not properly protected. An attacker may access the device through those unprotected channels and take control of it.
- Malicious user. The owner of the device may try to tamper the protocol or the application. This is another type of vulnerability, where the owner of the device is not trustable. The User or the Agent may try to modify the messages transmitted in order to profit from it.
- SIM card cloning. If the system stores sensitive information in the SIM card and an attacker is able to access that information, we may end with duplicated

Agents or Users in the system. We will see that this will not happen in our proposal because sensitive information is never stored in the Agent's or User's devices. Although no sensitive information is stored, the Agent's device SIM card or memory must be protected as it will be were the Agent application is installed and the adversary may try to change or modify it in order to decrease its security level.

4.2 Network Attacks

Attacks to the network are also possible and not very difficult, as mentioned by Paik [4]. Some of the attacks are described below.

- Network operator impersonation. As described in [4], an attacker is able to impersonate the network operator, leading a valid user to think he is using a legitimate operator, but instead his conversations and data are being eavesdropped. In the worse case, the adversary may conduct a Men-in-the-Middle attack, by altering the contents of the data being transmitted both sides.
- FTM impersonation. An attacker may also be able to impersonate the FTM where he can alter the amounts he receives from the User, or the values he has to deliver in. For example, when the User wants to withdraw some money, the attacker impersonating the FTM may act towards the User and the Agent as a legitimate FTM, but internally divert part of the User's credit to his own account. This is possible because it is the FTM's responsibility to synchronize the Users accounts at the m-Finance system and the banks,

merchants, IN systems or others. The attacker might conduct a major attack, where he is able to get a large amount of money from this attack, or might opt to conduct a low profile attack, where he diverts small amounts that may be unnoticed for a long period of time and only be detected when the User complains about the balance difference.

- Agent impersonation. The attacker may also try to impersonate an Agent. One possible way to do so is through Agent's MSISDN spoofing. Currently, the Agent is identified to the FTM by his MSISDN, but an adversary my start by eavesdrop the communications, collect important and sensitive information about the Agent, and then spoof his MSISDN.
- Malicious Agent. n this case a malicious Agent will be easily detected. This will happen because the FTM will always acknowledge the values of the transaction being processed with the User directly. If the attacker is trying to deceive the User, making him believe that he is depositing a higher amount of money, but instead the dishonest Agent is getting some of it to himself, the FTM will acknowledge the User about the exact amount that was deposited, as is shown in 3.2, during step 4. If the malicious Agent is trying to deceive the User during withdrawal, the User will also detect it because the FTM's voucher will identify the amount of money to withdraw from the User's account, and the User will receive a message informing him of that amount, as can be seen at 3.3, in step 6.

4.3 Cryptography and Protocols Attacks

Cryptography and protocol attacks are also possible, like for example:

- Replay attacks. If the messages exchanged are not properly protected by timestamps, random values or nonces, an eavesdropper may collect the information being transmitted and replay it to the system.
- MSISDN spoofing. Part of the protocol is supported in the User being identified by the Agent and the FTM platform by its MSISDN. Yet, this number can be spoofed and an adversary eavesdropping the communication between the User and the FTM, for instance, can read his PIN and spoof his MSISDN, thus impersonating the User to the FTM system. This will allow the adversary to start financial transactions on behalf of the User.
- Man-in-the-Middle attacks. As stated before, USSD messages are sent in plain text, so a Man-in-the-Middle attack is possible, where the adversary intercepts and modifies the USSD communications between the User and the FTM system. Such an attack was described in [4].
- Colluding attacks. We showed before that a dishonest Agent will be detected by the User, but if this attack is combined with a Man-in-the-Middle attack, intercepting and modifying the messages sent between the User and the FTM, then the malicious Agent attack is more certain to succeed when the User tries to deposit money. The malicious Agent must be able to send erroneous information about the amount to deposit, and at the same time a Man-in-the-Middle attack should be perform in order to intercept and modify the message sent by the FTM to the User. This way, the User will not be able

to detect that the FTM system received less money than what he really handed out to the Agent.

- Cryptographic attacks. Sensitive and critical information should be encrypted, but if we use weak cryptographic algorithms, the users may get a false feeling of security and while their information is vulnerable and may be tampered with. This may lead to fraudulent financial transactions, amongst other problems.

4.4 Operator Attacks

Operator attacks generally configure DoS attacks because the participants (Users, Agent and FTM) are not able to complete the transactions. Depending on the state of the transaction completion, this may be disruptive to the system. For example, the user commands a withdrawal operation but never receives the voucher from the FTM system. He may be tempted to try again a withdrawal transaction, and so on, until his funds are not enough. Although the money never really leaves his account until he presents the voucher to the Agent, this may cause perturbations to the service. The attacks may include:

- Message sending or receiving loss. The operator is not able to deliver the USSD messages due to attacks to the GSM infrastructure, for example.
- Message corruption. An adversary is able to corrupt part of the USSD messages sent, and prevent financial transactions from being completed, or the User's registration to an Agent.

- Message delay. An attacker is able to introduce delays in the message flow. Because of USSD characteristics, mainly session oriented and expiration time, message delay may cause transactions to abort due to timeout.
- Message flood. Another possible attack is to initiate many USSD sessions until the operator or the FTM system is no longer able to process them.
- Message tamper. And attacker can tamper with USSD plain text communications, but with the FTM finance model he would have to do that both between the Agent-FTM and User-FTM communications at the same time and he would also have to ensure that that the USSD communications he is attacking would not timed-out. Although difficult to conduct, this attack is possible and we will try to solve it with our proposal.

4.5 Social Engineering Attacks

Social engineering attacks are also possible. Deceiving the User or the Agent in such a way that he gives away critical information without noticing, and performing this attack without needing much knowledge on technology, but rather on psychology makes this kind of attacks very effective, so we have to consider them. These may include:

- Phishing. The attacker sends messages to the victim leading him to disclose sensitive information. For instance, the attacker sends a fake message asking the victim to send a response message to a certain number (the attacker's) with the victim's PIN. The message does not need to be similar to the ones sent by the FTM. It only have to be convincing. This is particularly important

because the User does receive such a message to activate his system registration, so an attacker will only have to be near and observe when someone is being registered, get his MSISDN, which includes the User's mobile phone number, and send the fake message in the right moment. The User will then respond to the fake message with his PIN, and after that, the attacker either stills the User's device, or impersonates the User through MSISDN spoofing. Another possible phishing attack can be executed when the attacker sends a message to the victim stating, for example, "We have problems with your account, please send your PIN to number <xxx> to validate your account". Again, the following step will be MSISDN spoofing or device stealing.

- Shoulder Surfing. In this case, the attacker will be observing the victim, User or Agent, in order to obtain important information, like the PIN. Latter, we will be able to use that information for his profit in colluding with another attack, like MSISDN spoofing (that will allow for Agent or User impersonation) or simply by stealing the victim's device.

4.6 Attack Mitigation

Mitigation of all the attacks described may not be possible. The security level that we can achieve highly depends on the devices processing capabilities and it is not possible to impose to every Agent or User of the FTM platform to purchase a mobile phone device with better security requirements. Some of the targets of this type of service are users without bank accounts and we must assume that not everyone will

possess a secure device. Because in our proposal we focus on the Agent – FTM secure communications, mainly because the Agent is a person responsible for receiving and delivering money to the Users, and hence subject to be tempted, we impose that the Agent's device must have minimum processing characteristics in order to be able to encrypt, decrypt and compute MACs for the USSD messages.

Even if we assume that all mobile devices have processing advanced capabilities, the underlying network is still vulnerable to attacks: USSD service may be unavailable due to a DoS attack, or the operator's GSM network may be attacked. These attacks may not aim the FTM service, but it will be affected anyway, because is it based on the mobile network infrastructure.

Replay Attacks. To prevent from replay attacks, transactions should include a random or a nonce, but this requires that the mobile device is able to produce random values, or that the mobile owner has the means to use a nonce generator of some sort.

Man-in-the-Middle Attacks. Multi-path and out-of-band confirmations are also viable solutions to prevent Man-in-the-Middle attacks as well as replay attacks. These can take the form of an extra pass-code or a code from a code matrix only known to the participants and never stored or computed by the device. This solution is also advantageous for User's devices with low processing capabilities. Although an eavesdropper can listen to the communications, if the codes are modified periodically, or ideally, never repeated, there is little direct advantage to this. If we use a matrix code, the FTM platform can keep track of the codes already used and not ask for them again. When all the codes are exhausted, the FTM system could send a message to User, asking him to renew his matrix codes at an Agent's, or to

the Agent asking him to go to the operator's facilities in order to obtain a new set of codes. This implementation can easily be done from the FTM platform side and although the communication between the User and the FTM system should be protected, the use of nonces from a matrix code card will be an inexpensive implementation to prevent this type of attack, even if encryption is not possible.

Challenge, Puzzles, Secret-Code Confirmations. The FTM platform can also implement an extra security measure of asking for another type of secure confirmation if the cumulative value of the transactions exceeds a certain value, or the number of transactions is higher than a certain threshold. This extra security measure should be implemented both at the User and the Agent.

Incomplete Transactions. The problem of incomplete transactions either due to unintentional or malicious network service disruption (which may be classified as DoS), or to the User's responsibility, for instance, not enough battery or out-of-cover zone, must be solved at the FTM side. A timed-out USSD session must conduct the FTM service to abort the transaction and the FTM should guarantee transactions atomicity, i.e., either every step described is executed correctly or all steps fail and the account status should rollback to the beginning of the failed transaction. The User will need to start a new transaction if this happens. This is not the scope of this work and we will not address this any further.

FTM's Resource Exhaustion. It is possible to implement measures that prevent DoS attacks due to the FTM's resource exhaustion. One possible solution is that for every time the FTM receives a message, it will reply with a challenge that the User needs to answer in order to continue. The challenge may be as simple as asking the value for a random arithmetic operation, but the USSD session will terminate if the

answer is not sent, or if it is wrong. This will have the advantage of requiring human interaction and hence prevent automatic attacks to the FTM service. We have not considered an implementation in our security proposal. Although it is a simple solution, it adds more steps to the transaction and the impact on performance needs to be evaluated.

User's Communication Security. We classified the User's devices into three categories, according to their processing capabilities:

- Category 1: Legacy devices: devices are not able to execute cryptographic functions due to their limited processing capabilities. We also include in this group those devices that do not support SIM applications;
- Category 2: Medium capable devices: in this category we include devices that have some processing capabilities and that allow for SIM application install
- Category 3: Top devices: in this category we included smartphones and similar devices. They allow for custom application installations and they provide some type of TCB components.

The level of User communications security will depend on the device category he owns. Medium capable devices will allow for the use of STK to develop secure USSD applications. Besides, it will also allow the operator to perform application OTA updates.

If the User loses his device or changes it, his account that is associated with the previous device MSISDN will need to be re-associated to the new device. This procedure will not be addressed in this work. If the User's device is stolen, then an adversary may have access to sensitive information kept in the phone so we do not

propose a solution supported in secret keys stored in the User's device, independently of the device's category. Instead, we propose a multi-path, out-of-band transaction confirmation, for instance, based on a code matrix personal card that will prevent a robber from using any kind of secret information stored inside the device, unless he also gets the card.

User Confidentiality. One problem that is not completely resolved in our proposal is transaction confidentiality. This is especially true on the User's side, mostly because in order to protect User's communications we need to be able to encrypt and decrypt what is being sent, which is not possible for category 1 devices. For other categories, a STK application may be used to execute this extra security measure, but it will require that the User shares a set of secret keys with the FTM system. In this proposal we considered only the Agent-FTM communications protection, which is encrypted, but the same mechanism can be extended to the User-FTM communication depending on his device category. In spite of this, User-FTM confidentiality should be considered for User safety. An attacker may be able to eavesdrop the USSD communications channel to know when the User carries money with him, and conduct a physical attack to rob the money.

Encryption. From the network side, USSD communications will be secured when encrypted with short term secret keys. As referred before, this is not completely possible for User's devices of category 1, but we assume that the Agent's device is of a higher category, and hence will allow for the generation of session keys, as we propose in the next chapter. If it is not possible to encrypt communications, then a certain level of security may be achieved through challenges or multi-path confirmations. In our proposal we use multi-path confirmations based on secrets that

only the FTM and the player (the Agent or the User) know about, like a pass code or a coordinate from a code matrix. To increase the level of security, the system must force the Agent or User to change the codes periodically, or after a certain number of utilizations. Even if the codes have to be transmitted in the clear, like for category 1 devices, this will make things more difficult for an attacker.

Short-Term Secret Keys. From the Agent's side, our proposal will encrypt all communications between the Agent and the FTM. The secret keys are only valid for a limited period of time (ideally, the keys will be valid only for the transaction duration, but if the performance is affected by the secret key generation it is possible to extend their validity), and they can only be generated if the Agent answers a challenge only he knows about. This will prevent an attacker from using the Agent's device, or from impersonating the Agent in any other way (for example, by spoofing the Agent's MSISDN) to generate valid session keys and access the system.

Malicious Agent. As we have referred before, if an Agent is malicious and wants to cheat on the User in order to get his money, we will need to conduct a more complex attack, because the FTM will send the User a confirmation for each transaction made in his account. So the malicious Agent will have to execute a Man-in-the-Middle attack to prevent the User from getting that confirmation. Of course, a Man-in-the-Middle attack would be impossible if the communication with the User is conveniently encrypted. This would require that the FTM and the User share a secret key or that the User has a set of public/private keys. Either way, the secret/private key needs to be stored at the User's device, which is something that may easily get lost or stolen, so the same observations on the about a solution based on storing secret keys in the device are valid here, and a more appropriate solution would be to have a similar

out-of-band multi-path confirmations. Besides, as we referred before, this is not possible to implement with category 1 devices.

Social Engineering Attacks. Attacks where the User or Agent is lead to disclose sensitive information by his own will are more difficult to prevent. One solution may be to use a kind of service watermark or seal of origin. The USSD service installed in the device would show a special icon, or mark. Any message sent by the FTM would have to display the same mark. This can be as small as an icon, but its absence would alert the user to an imposter FTM message. This solution will probably not be suitable for all devices categories, but if implemented would train device owners to search for signs of authenticity in received messages and discard the ones that do not present that graphic authenticity mark. For consistency, this should be implemented both at User and Agent side.

PIN Guessing Attacks. PINs are usually very short, and users tend to keep them for long, increasing the probability of a successful PIN guessing attack. The FTM should implement at the User and Agent side a mechanism that allows for only a number of consecutive wrong PIN value insertions, and lock the Agent or User account if that value is exceeded. This would prevent from PIN guessing attacks. Also, to increase the system overall security, both User and Agent PINs should have a validity associated and the FTM system should force the PIN to be changed when that validity is about to expire. When that happens, the service would send a message to the User or Agent for them to change their PIN and preventing them from using the rest of the services until that happens.

Chapter 5

Protocol

Next, we propose and describe a protocol we devised to mitigate some of the attacks referred. This protocol is based on SSL/TLS, but was modified in order to fit the USSD limited structure. The modifications will be kept to a minimum, in order to maintain the best possible correspondence to the SSL/TLS standard.

As referred before, the maximum size of a USSD message is 182 bytes and it presents a special format, depicted in the following Figure 6.

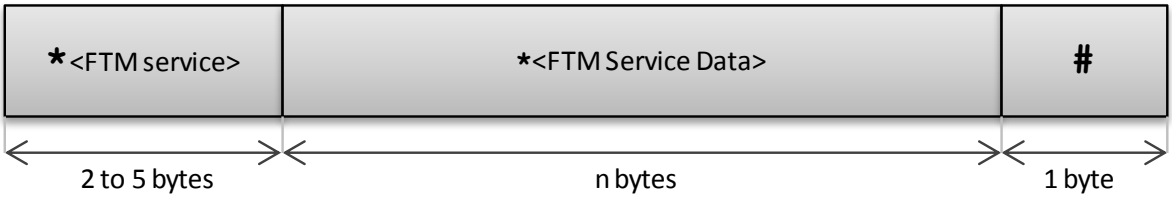

Figure 6 – FTM USSD message general format. FTM service identification, FTM service data and end-of-message indicator

In this figure, we assume that the *<FTM service>* is the USSD service number identifying the FTM application and that this application identifier value can range from 1 to 9999. Hence, its length can go from 2 to 5 bytes.

As stated previously, the USSD message starts with a *, is followed by the USSD application value and terminates with a #. Any additional parameters are placed after the USSD application identifier and start with a *.

The FTM Service Data is where we will send information with respect to the FTM service, and we will try to include in this field our SSL/TLS-based security protocol.

USSD does not handle multiple packets by itself. This should be done at the application layer, in case we need to send a message bigger than 182 characters. In order to guarantee the correct reception of a message greater than the maximum allowable size of the USSD message, the application must support out-of-order delivery and loss of messages. This implies that both the service application in the FTM and in the devices must be capable of temporarily storage messages, keep track of their order delivery and ask for message retransmission in case of message loss. Although possible, it would increase the delivery time and the processing time of a full message as well as storage space. We are not aiming for this message scheme. Instead, we will analyze SSL/TLS handshake message exchange and fit them to the USSD message size.

Keeping the messages within the limit of 182 characters has two advantages: the service is much simpler to implement in both sides (FTM, and device) because we do not need to deal with multi-packet messages, and the problems referred before, and the number of USSD messages exchanged will be kept to a minimum instead of varying with the size of the messages.

Another aspect that needs to be taken into account is that USSD does not guarantee message delivery. This means that the USSD service application at the FTM and the User and Agent devices must be capable of detecting incomplete transactions and USSD timed out sessions, and proceed accordingly to manage USSD sessions and keep data coherence.

5.1 Agent Session Registry at the FTM Platform

In order to guarantee secure communications between the Agent and the FTM, we propose a session authentication based on the following:

1 – The FTM platform has a set of public/private keys.

2 – The Agent has a prior knowledge of the FTM public key. This information can be given to the Agent upon his registry to the FTM platform which, by the time being, is done manually through the backoffice. Each Agent should be registered at the operator physical site, or other physical agency, by presenting the mobile device as well as other information and/or certificates that will be used later to prove the Agent is a legitimate one. This physical registry step will also provide the Agent with a secret code, or set of secret codes (eg., a PIN number, a password, or a code matrix) that will be used later to register the Agent's session. Each Agent will have his own secret code and this secret code will be registered by the FTM platform.

3 – The Agent initiates an USSL/UTLS session with the FTM.

First Phase – USSL/UTLS authentication:

Similar to SSL/TLS, the handshake is processed as following:

#	Direction	Message	Contents
1	Agent → FTM	client hello	Protocol version Agent's random List of Cipher Suites
2	FTM → Agent	server hello	Protocol version FTM's random Session ID Chosen Cipher Suite
3	FTM → Agent	certificate	FTM Certificate
4	FTM → Agent	server hello done	Signature on hash (Agent random || FTM random, data)
5	Agent → FTM	client key exchange	Agent public DH value
6	Agent → FTM	change cipher spec	
7	Agent → FTM	Finish	Handshake message
8	FTM → Agent	change cipher spec	
9	FTM → Agent	Finish	Handshake message

Table 1 – USSL/UTLS authentication handshake exchange. The format follows the standard SSL/TLS handshake protocol very closely

This is essentially the SSL/TLS handshake protocol, transmitted on top of USSD as a transport layer. We will show below the adaptations we propose in order for the SSL/TLS packets to fit in our FTM service data, as depict in Figure 6.

After this, the Agent and the FTM platform share a Master Secret and have authenticated each other. This Master Secret should be used to generate Session Keys for each session started, but should never be transmitted over-the-air (OTA).

As state before, we do not want to deal with multi-packet USSD transmission, so we will study SSL/TLS packets contents and adjust them whenever necessary to fit into our USSD FTM service data field.

The SSL record has the following general format that is encapsulated in a TCP message:

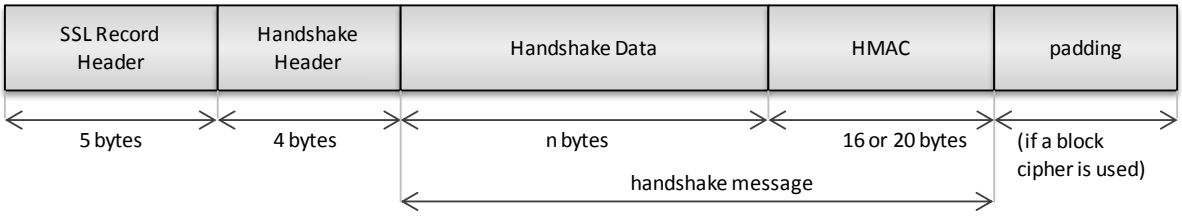

Figure 7 – SSL/TLS handshake message record

The SSL Record Header has the following format:

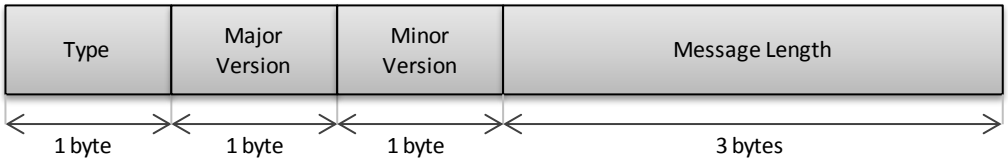

Figure 8 – SSL Record Header format

Here, the Record Type will have two possible values:

- 20 – Change_Cipher_Spec, where the protocol signals that it switches to the last negotiated cipher suite
- 22 – Handshake, for Hello and other connection-initiation messages

Major Version and Minor Version will be set to 3 and 1 respectively, to comply with the TLS standard. The Message Length gives the length in bytes of the Handshake Header and Handshake Data, but do not include the HMAC and padding sizes.

The Handshake Header filed is simply comprised of the Handshake Type and the Message Length, respectively, 1 byte and 3 bytes. Again, this does not include the HMAC field or the padding field.

For our USSD version of the protocol we will not need to include the padding field, but the HMAC will have to be considered. So, we will have to account for the SSL Record Header, the Handshake Header, the Handshake Date and the HMAC, which will lead to 29+n bytes. Below we will see the value of this variable part for each handshake message.

General SSL/TLS messages are very irregular in format, so next we will describe the account for each SSL/TLS handshake, header and message. This will allow us to compute the number of bytes of each message exchanged, and clearly check if each SSL/TLS message fits in a USSD message or if we need to limit, and thus simplify, the information exchanged.

We will include in our USSD version the SSL Record Header, the Handshake Header, the Handshake Data and the HMAC fields. Note that we don't really need to use 3 bytes for the Message Length, in the SSL Record Header or in the Handshake Header. The USSD message string can have a maximum of 182 bytes, and this value will fit in 2 bytes. This will allow us to save 2 bytes for data, if needed. This is not reflected on the next tables, where we opted to describe the messages as close as possible to the SSL/TLS standard.

- Client_Hello:

Field	Size and description
Handshake type: 1 – CLIENT_HELLO	1 byte
Message Length	3 bytes
Handshake data	<Message length> bytes - Major version: 1 byte (set to 3) - Minor version: 1 byte (set to 1) - Random data: 32 bytes - Session Id length: 1 byte - Session Id: 0-32 bytes (used to resume a previous session, sidestepping the key-generation phase. Client usually indicates 0) - Cipher suite length: 2 bytes - Cipher suite list: $2\text{-}2^{16}$ – 1 bytes (represented by a 2-byte number and sorted in preference order. Server may choose any suite in the clients list) - Compression list length: 1 byte - Compression list: 1-255 bytes (usually is 0, representing null)

Table 2 – Client_Hello handshake message. Message field listing and byte account.

The maximum size of handshake data is 65860 bytes.

The length of the Client_Hello message highly depends on the cipher spec list. This is limited to what the device API can offer. If the Agent is only able to support a maximum a 2 cipher suites, the size of the handshake data is highly reduced to 43 bytes. This is suitable for our USSD version, so we limit the number of cipher suites to only two possible cipher suites.

- Server_Hello:

Field	Size and description
Handshake type: 2 – SERVER_HELLO	1 byte
Message Length	3 bytes
Handshake data	- Major version: 1 byte (set to 3) - Minor version: 1 byte (set to 1) - Random data: 32 bytes - Session Id length: 1 byte - Session Id: 0-32 bytes (used to resume a previous session, sidestepping the key-generation phase. Server specifies a session id) - Cipher suite: 2 bytes (2-byte number of the suite chosen by the server) - Compression method: 1 byte (usually is 0, representing null)

Table 3 – Server_Hello handshake message. Message field listing and byte account.

The size of handshake data is 70 bytes.

- Certificate:

Field	Size and description
Handshake type: 3 – CERTIFICATE	1 byte
Message Length	3 bytes
Handshake data	- Certificate list length: 3 bytes - Certificate list: $1\text{-}2^{24}$ – 1 bytes

Table 4 – Server_Certificate handshake message. Message field listing and byte account.

The maximum size of handshake data for the SSL/TLS protocol is 16777218 bytes.

The certificate list is usually a chain of certificates, with the first being the server's own certificate, followed by any authenticating certificates from CAs. In this case, each CA signs the server's certificate with its private key. The CA's certificate contains the CA's public key, which the client uses to verify that the CA was, in fact, the signer of the server's certificate. The client typically has its own copy of the CA's certificate, making the transaction secure. Depending on the size of the server's certificate, this size can be greatly reduced.

- Server_Hello_Done:

Field	Size and description
Handshake type: 14 – SERVER_HELLO_DONE	1 byte
Message Length	3 bytes (set to 0)

Table 5 – Server_Hello_Done handshake message. Message field listing and byte account.

There is no handshake data, so its size is 0 bytes.

This message is used to tell the client that the server has finished its hello sequence and that the client may begin the key-transfer process.

- Client_Key_Exchange:

  For RSA for the key exchange we have the following values:

Field	Size and description
Handshake type: 16 – CLIENT_KEY_EXCHANGE	1 byte
Message Length	3 bytes (the value is 128 due to RSA encryption)
Handshake data	- Major version: 1 byte (set to 3) - Minor version: 1 byte (set to 1) - Random data: 46 bytes (cryptographically secured, i.e., generated by a cryptographically strong pseudorandom generator)

Table 6 – Client_Key_Exchange handshake message. Message field listing and byte account.

The size of handshake data is 128 bytes due to RSA encryption.

The message contains a pre-master secret encrypted with the FTM's public key that was in its certificates. The minor and major version fields are encrypted along with the random data. Notice that if we account for the HMAC of 20 bytes, as well as for the SSL Header of 5 bytes, this message will be 157 bytes long. Adding the USSD delimiters and FTM application identifier will produce a USSD message with 164 bytes. It still fits the USSD size constraints.

After this message, both the Agent and the FTM platform can generate keys for the rest of the session.

- Change_Chiper_Spec:

Field	Size and description
Record type: 20 – CHANGE_CIPHER_SPEC	1 byte
Data Field	1 byte (set to 1)

Table 7 – Change_Cipher_Spec message. Message field listing and byte account.

The Change_Cipher_Spec is not a handshake message but rather a SSL record type that indicates a switch to the last negotiated cipher suite. In this case, we do not need to compute the 4 bytes for the Handshake Header, and so the size of this message is 1 byte (the size of the Data Field).

- Finish:

Field	Size and description
Handshake type: 20 – FINISH	1 byte
Message Length	3 bytes (value is 36)
Handshake data	- MD5 hash: 16 bytes - SHA1 hash: 20 bytes

Table 8 – Finish handshake message. Message field listing and byte account.

The size of handshake data is 36 bytes.

The Finish message is encrypted and the hashes are computed over all the previous handshake messages, verifying that the unauthenticated messages were not tampered with.

At the end of the USSL/UTLS authentication phase, both the FTM and the Agent will have a set of session keys that can be used to protect and encrypt the communications that follow. The session keys are obtained through the master key they agreed upon and for which both contributed. The inputs for the master key are the Agent and FTM random values and the pre-master secret sent in the Client_Hello, Server_Hello and Client_Key_Exchange messages. The details of the generation of the session keys from the master key depend on the protocol being used. The Agent and the FTM now will independently produce a set of session keys, to be used in the next communications. The method is slightly different for TLS and for SSL, but the end result will be

- Agent write MAC secret: MAC_{Agent}
- FTM write MAC secret: MAC_{FTM}
- Agent write key: Key_{Agent}
- FTM write key: Key_{FTM}
- Agent write IV
- FTM write IV

The Agent and FTM MAC keys, MAC_{Agent} and MAC_{FTM}, are used to guarantee the integrity of the messages sent by each other, while the Agent and FTM write keys, Key_{Agent} and Key_{FTM}, are used to encrypt the messages sent.

Second Phase – Agent authentication:

During the second phase, the FTM platform asks the Agent to introduce a secret personal code, in order to confirm its identification. This secret code was given to the

Agent upon his physical registration to the FTM platform, as described before, and can take several forms: a PIN number, a password or a specific code from a code matrix only the Agent and the FTM platform knows about. Whatever aspect the secret code takes, it should not be saved in the Agent device and it will not be transmitted to the FTM. Instead, the Agent will answer with a hash of the secret and the FTM will compare the received hash with its own hash of the correct code. This will constitute and out-of-band verification, and will allow the FTM platform to verify that the Agent is a legitimate one, as only the legitimate Agent is able to correctly reply to the FTM request.

During this phase, replay attacks can be prevented if the USSD message includes a protection mechanism, like a random value, a nonce, or a timestamp. Assuming that a random value is used, the messages exchanged would be like the following:

#	Direction	Message	Contents
1	FTM → Agent	Agent code request	FTM's random (32 bytes) secret details (2 bytes)
2	Agent → FTM	Agent code	FTM's random (32 bytes) Agent's random (32 bytes) hash of secret (32 bytes)

Table 9 – Agent Authentication exchange. Message field listing and byte account.

The message sent by the FTM to the Agent will be encrypted with the Key_{Agent} session Agent key computed before, and will concatenate with a MAC computed with the MAC_{Agent} session MAC key. We propose the size of 32 bytes for the MAC field, but this size can be reduced to 16 bytes to meet performance requirements, if necessary.

The hash of the secret can be computed in two different ways, which will impact slightly on the performance and security of the protocol. Either we use a standard MD5 or SHA-1 HMAC of 16 or 20 bytes respectively, in which case the FTM does not need to store the Agents secret code but only the hashes of the value, or we use the MAC_{Agent} session MAC key to compute a hash that is different for every session. In this case, the FTM will need to store the real Agent secret code in order to verify the correctness of the hash value he received from the Agent, which will impact on performance on the FTM side. On the other hand, this solution prevents replay attacks because even if the secret code maintained for a long time, the MAC_{Agent} key will be different for each USSL/UTLS session and an attacker will not be able to perform a replay attack.

This step can be modified to include a protection agains social engineering attacks, as referred in 4.5 and 4.6. If the mitigation is to be implemented, the first step should be modified. The FTM would send the Agent a number of extra bytes in the Agent code request that would display in the Agent's device as a special mark or seal of origin, graphically identifying the FTM service to the Agent. If the device is not capable of displaying such mark, either would not display any mark at all or present a message stating that the sender is the correct one. This measure would not increase the number of messages exchanged, but would add a few more bytes to the first message and some processing in the Agent's device, but it would help avoiding the social engineering attacks.

5.2 User Registry and Activation

Recalling the User registry and activation described in 3.1:

1. Presentation: User physically goes to an Agent and presents personal ID documents (ID Card, SSN/NIF, etc) and contacts (mobile phone number, email, etc). He shows also his device unique Mobile Subscriber Integrated Services Digital Network Number (MSISDN).
2. Registration: Agent sends a message to FTM with the User name and MSISDN.
3. Registration Acknowledge: FTM validates message and in case of success replies to the Agent:

   ```
   User <MSISDN> was successfully registered.
   ```

4. Activation Request: FTM sends message to the User:

   ```
   You have been successfully registered with PIN <PIN>. Please
   confirm service activation sending the message "ACT <PIN>" to
   number <LA>.
   ```

5. Activation: User must send the message to the number indicated with the PIN he received.
6. Activation Acknowledge: FTM validates the message and, if successful, replies to User:

   ```
   Service activation was successful. Current PIN - <PIN>. For
   security reasons, change your PIN as soon as possible.
   ```

The message flow between the Agent and the FTM can be protected with the established session keys, where all messages are sent encrypted and contain a

MAC for integrity. This will occur at steps 2 and 3, but the other steps are unprotected.

Step 4, Activation Request, does not include the Agent's identification in the initial proposal. We suggest that such identification should be added to this message, for two reasons: to inform the User of his Agent (the User can only be registered at one Agent at a time), and to prevent from possible Agent impersonation attacks.

The protection mechanism between the User and the FTM raises some difficulties due to the fact that we cannot impose certain device requirements to the User's device. Maybe the device has enough computational power to implement a protection mechanism similar to the one we propose for the communication between the Agent and the FTM, but this is not a certainty. Besides, the performance of the USSD channel may be a drawback in this case and the User may choose not to use the service if the communications take too long. Anyway, even if we do not assume that the communication between the User and the FTM is protected by a set of shared secret keys that change from time to time, security can be increased if three final steps are added to the process:

7. Secret Code Hand Out. Agent gives the User a sealed envelope with a secret code matrix. The Agent does not know the matrix itself, but the envelope is marked with a serial number that identifies the matrix in the FTM platform.
8. Secret Code Hand Out Notification. Agent sends a message to FTM with the User MSISDN and the secret matrix serial number.
9. Secret Code Hand Out Information. FTM sends the User a message indicating that the matrix serial is associated to the User

```
Matrix number <matrix serial number> was attributed to you.
```

Step 7 will provide the system with an out-of-band verification, while step 8 will inform the system of the matrix associated to the user. The matrix codes will be used to confirm future user transactions. Step 9 will confirm to the user that the Agent took the correct procedure.

The matrix secret codes sealed envelopes should be provided to the Agent by the FTM system, at his request. This way the FTM will be able to keep a track of the envelopes distributed. As usual in other cases, the User must be instructed to refuse the matrix envelope if it shows any sign of being tampered. In case the User looses the secret code matrix, he simply should go to an Agent, proceed with a code cancelation and asks for a new one. This process is not covered here. Notice that because Agents need to apply for the role, we expect them to present a certain guarantee of honesty. We also expect that the User is aware of the registration steps he must follow and is able to detect if an eventually dishonest Agent gives him an open code envelope (it should be closed and sealed) or none at all, and knows how to present a complaint about that Agent incorrect procedure.

5.3 User Deposit

Recalling the User deposit process, describe in 3.2:

1. Presentation: User physically goes to the Agent and hands out the amount of money.
2. Deposit Request: Agent sends a message to the FTM, with the User MSISDN and the amount.

3. Deposit Acknowledge: FTM validates the message and, if correct, replies to the Agent:

```
The <currency> <amount> was deposited successfully on the client
<MSISDN> account.
```

4. Deposit Information: FTM sends a message to the User:

```
<currency> <amount> were deposited in your account.
```

Like before, the messages exchanged between the Agent and the FTM system will be protected with the USSL/UTLS session keys. Step 4 is a confirmation to the Use that the Agent sent the correct information to the FTM platform. If step 4, Deposit Information, is modified to include the user balance he will have an extra means to confirm that he has not been the victim of any attack to his account, but if this information is to be sent as a regular, unencrypted message, then the balance information should be omitted. Yet, we suggest that this message is modified to include the Agent identification (his name, at least).

5.4 User Withdrawal

Recalling the process described in 3.3:

1. User Withdrawal Request: User sends a message to the FTM, indicating his PIN and the amount to withdraw.
2. Withdrawal Authorization: FTM validates the message and, if successful, generates a voucher code, identifying specifically the withdraw of the amount requested, and sends a message to the User:

```
Your request was successful. To withdraw <currency> <amount> from
your account go to the Agent and show the code <voucher code>.
```

3. Presentation: User goes to the Agent and presents the voucher code.
4. Agent Withdrawal Request: Agent sends a message to FTM, with his own PIN and the voucher code.
5. Withdrawal Verification: FTM validates the message and, if successful, replies to the Agent:

```
Voucher <voucher code> was successfully withdrawn. Deliver
<currency><amount> to the client.
```

6. Withdrawal Information: FTM sends message to the User:

```
Agent will deliver you <currency><amount> that were withdrawn from
your account.
```

7. Withdrawal: Agent gives the requested amount to the User.

The attack were an adversary gains access to the User's device and is able to retrieve sensitive information out of the device's list of previously dialed numbers, namely, is able to obtain the User's PIN out of the Step 1 message, can be prevented if the protocol is modified in such a way that the User never sends his PIN without being requested to. In this case, Step 1 should be modified in order to include 2 extra steps:

1'. User Withdrawal Request: User sends a message to the FTM, indicating the amount to withdraw.
2'. User Identification Request: FTM replies to the User, asking for his PIN:

```
Please insert your PIN.
```

3'. User Identification: User inserts his PIN.

Then, the other steps would follow.

Again, the only messages that are able to be protected by our protocol are the ones described in steps 4 and 5. If the User device is able to perform some encryption and hash computation, we could use a USSL/UTLS session handshake to provide for the User and FTM session keys, and all the following messages between the User and the FTM could be encrypted. Because we cannot assume that, we propose a simple modification to the protocol, with more two intermediate steps after original step 1, or corrected step 3', as described below:

a. User Withdrawal Verification Request: FTM validates the message and if correct sends the User another message asking for a certain code in his secret code matrix:

```
Please confirm your request with code from coordinate <matrix
coordinate>.
```

b. User Withdrawal Verification: User answers with the requested value.

Although these new steps do not fully prevent from an eavesdropper to collect information on the User's secret code matrix and latter impersonate the User to the FTM and withdraw money from the user's account, the attack will be more difficult this way. We have already presented some other mechanisms that can further protect this type of attack.

Again, social engineering attacks can be prevented if the User is instructed to ignore messages from the FTM that do not show a watermark or seal of origin mark, but

this protection mechanism may be difficult to implement in category 1 devices, if not impossible.

There is another problem with the initial message flow: if the Agent does not have enough cash on hand for a withdrawal, he would know it only after presenting the voucher to the FTM system. The problem can be solved simply if on step 3, Presentation the User also informs the Agent of the amount to withdraw, a value the Agent will confirm after sending the voucher code to the FTM. This implies that the Agent has to trust the value the User told him is correct. Another possibility is to add an extra 2-step Agent Withdraw Confirmation after step 3, where the FTM asks the Agent if he can support the withdrawal transaction. If he says yes, the procedure will continue, but if he says no, the voucher is canceled and the transaction is undone.

Another security concern is related to the Agent unequivocal identification in the message. MSISDN identifiers are not very user-friendly, so Agent and User names should also be used and checked against the MSISDN (this does not mean that the name should be unique, but the combination of name and MSISDN must be unique). This would prevent attacks were an adversary stoles a User's device that includes a voucher and tries to withdraw the money. If the device MSISDN is checked against a name, this attack would be difficult to success and this procedure would also minimize the impact of typos.

Chapter 6

Implementation

Our proposal as not been implemented yet, but we foresee that some issues need to be addressed when developing the necessary code.

6.1 FTM Implementation

While we do not expect any kind of problems for the FTM platform, that we assume to be a set of high processing capabilities servers, we present next a list of properties and functions that need to exist on the FTM side.

- A pair of public/private keys. The Agents will have access to the FTM's public key, which will allow them to start a USSL/UTLS session.
- A database of each Agent's secret code, or a set of secret codes, if the option for a code matrix is used. Each record must keep track of the Agent's code validity, the number of consecutive fails, and the number of times the Agent tried to start a USSL/UTLS session in a hour (or another gap of time that is more suitable). This value will allow the FTM platform to detect if the number of sessions is within a normal utilization level or, on the contrary, it is too high for the Agent's profile, and hence we may suspect that a DoS attack is occurring by replaying the Agent's older messages, for instance.

- A database of User's secret codes or code matrixes. Each record will keep track of the User's secret code serial number as well as the Agent's ID. Every time a Agent delivers a sealed secret code envelope to a User, he must send the envelope's serial number to the platform, along with the user's MSISDN. This will be recorded in the FTM database, as well as the date of delivery. Again, the FTM platform should keep track of how many times the User failed to correctly indicate the requested code, the maximum number of transactions expected within an hour (or another amount of time considered more suitable), and if a code matrix is used, the secret codes already used. This registry will help detect anomalous usage of the service.
- The User database account settings should also include a limit for the amount of e-money the User is authorized to withdraw. For this case, we suggest that the FTM service strongly advises for each User to indicate that amount, for security reasons.

6.2 Device Implementation

Implementing this security service on the device needs to take into consideration the device's category. As we stated before, category 1 devices will allow for very limited secured USSD service application, if any at all, but for category 2 and 3 we expect to implement a USSD application with a certain level of security. We assumed that the Agent's device would have some advanced processing capabilities, so we can impose that the Agent possess at least a device of category 2 and that it is able to encrypt, decrypt and perform hash computations. This can be done through STK

applications, taking some advantage on the SIM's secure capabilities, or if a category 3 device is present, use the devices TCB component to perform these tasks.

STK and the USAT enable the SIM to initiate actions that can be used for value-added services delivered over mobile devices. This enables the SIM to provide an interactive exchange between a network application and the end-user, and access, or control access to the network. For GSM 2G, the STK is defined on [14] and USAT for 3G is defined on [15]. STK- based applications are supported by category 2 and 3 devices and they offer some level of security with SIM-based encryption. There are two possibilities of user interfaces:

1. the customer dials a USSD service on his device and the USSD application service will request more information on successive USSD messages or
2. the operator installs an application that will have its menus embedded in the normal device user interface.

The first approach is simpler to implement, but requires that the Agent or User are familiar with the USSD service structure. It may be difficult for infrequent users, but experienced ones will find this approach more efficient. Anyway, the USSD service implemented should not allow for the User to send sensitive information in the first message, in order prevent from reading out of the previously dialed numbers phone list.

As for the second approach, the costumer can send a first USSD message to the operator to request the service and is then commanded to download a SIM application to reside in his device in the standard phone menu.

For devices of category 3, more solutions are possible, besides the ones presented before. A Android smartphone application may be downloaded from the Android Market, and other devices may also provide more advanced applications from the OS provider store, like AppStore for the iPhone.

Whatever solution is implemented, the developer must have a common main concern: the USSD service must delete all previous messages and associated information from the device's records, so that no critical information is accessible through the device's memory in case and adversary has access to it. This is especially true for the case where a long term key or PIN is used.

A survey on available open-source code for SSL libraries for embedded system showed that there are already some solutions that can be explored and adapted if needed. Some of the available solutions are presented next.

6.2.1 CyaSSL[2]

CyaSSL is an embedded SSL library, C-language-based, targeted for embedded RTOS environments, that claims to be small, fast and possess a good feature set. The library is available for a variety of OS, including Linux and embedded Linux, Android, iPhone, etc. According to the site specification, CyaSSL uses the following cryptographic libraries: CTaoCrypt, Crypto++ and NTRU, which provide RSA, DES, 3DES, MD5, SHA-1, DSA, Diffie-Hellman and AES-256 cipher suites. It also

[2] http://www.yassl.com/yaSSL/Home.html

supports random number generator, large Integer support, and base 16/64 encoding/decoding.

This library is open source and free download of the source code is available as long as the user adheres to version two of the GPL license. A commercial use of the library requires commercial licensing.

6.2.2 PolarSSL[3]

PolarSSL claims to be an SSL library written in ANSI-C that makes easy for developers to include cryptographic and SSL/TLS capabilities in their embedded products. It is characterized for its small memory footprint, portability, whose code is easy to reduce and expand and with no external dependencies on other libraries other than libc. This library offers several cryptographic algorithms, like AES, DES, Triple DES for ciphers, MD5, SHA-1 for cryptographic hash functions, RSA, DSA and Diffie-Hellman key exchange for public-key cryptography.

The licensing is done according to the dual licensing model, under the open source GPL version 2 as well as a closed source commercial license.

[3] http://polarssl.org/

Chapter 7

Performance Analysis

In this chapter we will make some considerations related to performance analysis of our proposal. To our analysis, the performance of this proposal is dependent mainly on five things:

- Device capabilities. This is due to the type of device used. Assuming only the Agent's type of device, the one that is requested to process hash computations, encrypt and decrypt, the performance will be affected by the type of device he possess. Even category 2 devices can perform better or worse, depending on its processor or amount of memory. The same applies to category 3 devices, adding to the fact that this type of devices may include many more applications that will compete for processor and memory.
- FTM servers' capabilities. Mainly, FTM servers will compute hashes, encrypt, decrypt update databases and log transactions. We expect FTM servers to have increased processing capabilities, multi-processing, etc, so hash computations and encryption/decryption will have good performance. Database access and logging will depend on the number of transactions requested, but will mostly comply with a certain level of performance.
- Number of messages exchanged. Obviously, the more messages, the worse for the service performance.

- Transmission time of the USSD protocol. Transmission time depends on the distance each device is to a mobile station but will not vary greatly with each use.
- Processing time of the USSD service. This value is difficult to foresee because it will include the FTM servers capabilities for one of the sides, and the device capabilities for the other side. Some experiments on the Portuguese TMN mobile operator showed an average value of 7 seconds for each USSD message, but this value greatly depends on the type of service that is requested. Simpler services, like the value spent on a call, can take up to 3 seconds, while more complex ones, like the information about the remaining value to spend, which involves database consulting and data processing, may take up to 22 seconds. If database access and logging on the FTM side is done in background whenever possible, this service will mainly demand the use of cryptographic functions on each side, so processing time will be mostly dependent on the performance of those function at each device or server.

The number of messages cannot be further reduced without compromising the security properties that are the main goal of this work, but it is possible to limit the number of times the Agent need to establish a secure session. For example, the keys are stored in the device, with an associated timestamp, that will invalidate them after some time and force the process to be restarted.

USSD provides an adjustable timeout value for each message. The default is 600 seconds, but during the first phase, this timeout should be the minimum possible. Our suggestion is to use a value around 10 seconds. No user intervention is required

other than starting the application. USSD is a real-time session-oriented service and the session remains open until the user, application, or time-out releases it. For the Agent authentication phase, with a timeout of 10 seconds per message, the agreement of the shared Master Secret would take a maximum of 100 seconds. For user-friendlyness of the service there should be some feedback to the Agent for each step, in the form of a progress bar, for instance.

Chapter 8

Further discussion and analysis

As mentioned before, there are some limitations when trying to use a secure protocol in some old devices. Although this proposal tries to put the maximum effort on the FTM side, some computation has to be done on the device side. But SSL/TSL has a major advantage of being a well-known protocol, and this solution can be extended to other communication channels with limited bandwidth, like Bluetooth, RFID, etc.

Another advantage of this solution is that we are able to change the cryptographic library and deal simultaneously with different versions of the protocol only on the FTM side. This allows having several levels of security enforcements, depending if we are dealing with category 2 mobile devices, or category 3 smartphones with TCB components. This information may be of great use if the communications between the User and the FTM are to be secured as well. Our suggestion is that the protocol version should be a configuration value introduced during the application installation in the User's device and should only be allowed alterations if the user goes physically to the operator location and presents his reasons: a new mobile device to be registered, or a software upgrade, for instance. Otherwise, the protocol version to use should be fixed and protected. In other words, it should prevent dumbing down attacks.

The USSD application on the FTM platform should be implement in such a way that withdrawal and other transaction limits should be read from a configuration database that would include not only the client account id, his account type but also his

protocol version. The reasons why this should be included on the FTM databases and not only the phone itself is to prevent attacks where the adversary is able to modify this value in order to access a higher withdrawal limit. The information sent by the mobile device is checked against the information stored in the FTM databases. If they match, the operation is allowed, otherwise, we may be in the presence of an attack to the system, the transaction should be aborted and an appropriated log and alarm should be generated.

The proposed solution includes the hashing of the messages exchanged between the Agent and the FTM, which will provide for the integrity of the messages that can be checked at both sides. Also, the secrets exchanges are also hashed, either with SHA-1 or MD5, or with the MAC_{Agent} session key, which will change every time the Agent starts a new USSL/UTLS session.

Authentication is supported through the PIN value for both the User and the Agent. Confirmation of their identity can also be achieved through the use of the out-of-band secret code request. Both mechanisms will also provide non-repudiation of the operations being executed, but this is only valid for the User and Agent sides. The FTM authentication and non-repudiation to the Agent is only guaranteed by the use of the FTM's public key, but we have not suggested any type of FTM certificate verification on the Agent side, other than FTM Certificate verification to a devices pre-installed one, because that would require access to a Certification Authority, which may not be possible. Besides, checking against a pre-installed Certificate has some disadvantages: certification revocation will be very difficult and a virus or other malicious application may be able to change or delete the FTM's certificate. As for

the User, communications are not encrypted and he is not able to certify that the FTM is not being impersonated by a malicious adversary.

User MSISDN alone cannot be considered a good authentication method because it can be spoofed, but can be used it to detect misused or possible attack of the service, for instance, if the same MSISDN is active at the same time in different mobile cells.

As for communications confidentiality, it is only assured between the Agent and the FTM through the use of session encryption keys. Our proposal does not cover User communications encryption although this is also possible if certain requirements are met, like the User's device is, at least, a category 2 device.

We have proposed a protocol that follows SSL/TLS standards but when defining some optional parts of the standard our advice is to follow TLS standard rather than SSL's. Although similar, SSL 3.0 implementation is considered slightly weaker than the TLS 1.0 implementation. Our solution is a lightweight version of the standard, and we suggested the use of a maximum of two cipher suites. This will provide some protection against downgrade of the protocols to a less secure version or to a weaker one.

Attacks to the SSL/TLS protocols are reported periodically, as well as theoretical studies of possible attacks. These reports and studies conduct to bug fixes, recommendations and stronger algorithm implementations, and although many of them are related to the standard use of the protocol over TCP, some of them may apply as well to USSD communication channel, so developers of this service should

maintain a close attention to these reports in order to fix possible cryptographic vulnerabilities on the system.

PIN change command is not in the scope of this work, but a mechanism to prevent social engineering attacks as suggested in previous chapters, with a watermark or seal of origin character, could be used whenever the User (or Agent) issues a PIN change command. Also, every message sent to the Agent or User should include this special mark. The owner of the device should be aware that without that mark he can be the victim of a phishing message or other malicious attack. The implementation of this security mechanism may be very difficult in category 1 devices, if not impossible, but for category 2 and higher, this mechanism will train the device owner to search for marks of credibility in applications. Of course, the transmission of the mark should be tamper-proof and resistant to replay and guessing attacks.

The fact that the User is only able to be registered at one single Agent may look like a inflexible option but provides some extra securities assurances: the User will only have to memorize one PIN and carry one secret-code matrix envelope, it will be easier to control MSISDN spoofing, because the Agent's MSISDN will be paired with the User's MSISDN, so any other combination must be interpreted as an attack. It is also easier to detect and manage if an attacker is doing PIN guessing on a robbed or lost device and block the User's account.

Chapter 9

Suggestions for future work

This work is only a proposal for a small part of the FTM system. Some issues are intentionally not covered.

9.1 FTM authentication to the User

This may only be possible if the User and the FTM share a secret or if the User is able to verify the FTM certification.

For the first case, the shared secret should be given to each other when certain conditions are met, like when they are physically near each other, for instance, the User is at the FTM's facilities and they acknowledge each other, or through another TTP, like an Agent or any other entity that both trust. But having the Agent certifying the FTM's identity is not a good solution: the Agent itself is not authenticated to the User and he may be malicious and collude with a malicious FTM in order to deceive the User and lead him to deliver them his money or other type of sensitive information.

The second case requires that the User receives the FTM's certification and is able to contact some kind of certification authority in order to verify the FTM's identity. This requires User's advanced processing capabilities as well as another

communications channel to a trusted authority that would increase the complexity of the system's architecture.

FTM authentication to the User may be impossible if they have not previously met and if the User only possess a category 1 device, but for the other cases, is a line of work that deserves to be explored.

As referred before, the inverse situation of the User authentication to the FTM is resolved through the use of PINs, shared secret codes and the MSISDN of the User's device. A solution similar to the one presented in [13] may be used to increase the User's PIN security, especially when transmitted unencrypted on a USSD message, This study suggests that the PIN should be modified and combined in a certain manner with a code from an external source every time the User needs to initiate a transaction. This would provide one time keys and prevent PIN guessing attacks, but User's may be resistant to use a solution that requires for PIN modification every time and it will not provide confidentiality not integrity of the transactions.

9.2 Agent – User mutual authentication

As noted before, the Agent and the User do not authenticate each other, but the communications flow of each transaction imposes that the User acknowledges of everything the Agent commands the FTM respecting the User's account. Yet, the problem of having the User and the Agent authenticate each other, especially the first time they meet, may be solved if we consider the FTM as a TTP.

If we can use the FTM as a TTP, the platform may send a message to the User with a special code, and another message to the Agent with another code. Then, the User would have to reply the FTM with the Agent's code, and the Agent would do the same with the User's code. The FTM would combine the User's answer with the code sent previously to the User, and would also combine the Agent's answer with the code sent to him. The result of each combination would be sent to the participants, one to each other, and they would have to check visually if the values were same and confirm it to the FTM. The FTM would already know that the values were the same or not, but with this confirmation the FTM may be certain that they have truly acknowledged each other. This type of confirmation goes in the clear on the User's side, so further analysis would be needed in order to detect and mitigate vulnerabilities in this scheme.

One other aspect of the system could be used to authenticate the User and the Agent for the first time: they have to be physically near each other. The User and the Agent are able to visually check the information that each other receives from the FTM. An interesting work that explores the fact that both unauthenticated parties are physically near to each other is presented in [10]. Although the work refers to camera phones and bar-code readers, which require, at least, category 2 devices, the proposed solution imposes no prior knowledge of the participants or the use of a TTP.

9.3 FTM authentication to the Agent

This aspect was not covered in this work, mainly because it would require that the Agent has also a certificate or a set of public/private keys. The number of steps in the authentication phase would increase, but even so, the protocol could be altered to comply with this requirement and performance analysis would have to be conducted in order conclude on the impact of these extra steps.

9.4 User confidentiality

As noted throughout the work, there is no encryption on the User's communications to the FTM. This aspect could be studied in order to find a solution, even if it does not fit all devices' categories, to protect the User's communications. As noted before, the fact that an eavesdropper is able to listen to Users USSD communication may endanger him: not only is he giving away security information, like his PIN and other secret codes, he is also transmitting information about his account, the amount of money he is carrying and the amount of money he is going to receive.

For category 2 and 3 devices, a solutions similar to the one we propose for the Agent communication is possible, although that solution would have to take into account the fact that presently, the User is unknown to the FTM until he registers with an Agent. How much would such a solution impact on performance, and what could be done to category 1 devices is something that should be addressed in future works.

9.5 Human-Errors

One problem that is not discussed in this work is how to correct for human error. For example, the Agent may incorrectly introduce the User's MSISDN when sending a deposit message to the FTM. This may cause the money to incorrectly being deposited in another User's account. The error will be detected because the User will not receive the corresponding Deposit Information message, but the procedures to allow the Agent to correct this mistake were not shown. This aspect deserves more analysis, Human error is something that we have to account for when introducing values into the system and we need to provide a means for detecting and correcting them.

Chapter 10

Summary and Conclusions

This work analyses an existing m-finance scheme and proposes some modifications in order to improve its security over the USSD communication channel.

USSD itself is not secure, but it is available everywhere and is inter-operable, i.e. is not telecom dependent. Besides, it requires no new hardware, and requires practically no software download. Its issues relative to security and confidentiality can be addressed through the use of secure protocols on top of it. Despite so, its simplicity will not allow for a very complex application and is not an ergonomic solution, but it suits a certain market very well.

We analyzed system the treat model, considering the existing transactions, and we proposed a solution that would protect a certain path of the communication, namely, between the Agent and the FTM. We suggested the implementation of SSL/TLS over USSD, a lightweight version that we called USSL/UTLS. Such implementation of the SSL/TLS protocol is actually independent of the communication channel used. The same mechanism we proposed can be used if the communication if done over bluetooth, NFC, WiFi, or other. The main advantage of our proposal is the fact that SSL/TLS is a standard that is still considered secure, but in case it suffers any evolution that increases its security, that modification can easily be deployed in our solution, with little modification of message flow already implemented.

In conclusion, we are convinced that our proposal for using USSL/UTLS can be implemented in current category 2 and 3 devices and that it increases the security in Agent – FTM communications. It also requires a minimum of user intervention and provides extra security measures that authenticate the Agent, and provides information integrity and confidentiality.

References

[1] ETSI: Digital Cellular Telecommunications System; Unstructured Supplementary Service Data (USSD) – Stage 1 (GSM 02.90). Technical Report, ETSI, March 1997

[2] ETSI: Digital Cellular Telecommunications System; Unstructured Supplementary Service Data (USSD) – Stage 2 (GSM 03.90). Technical Report, ETSI, December 1996

[3] ETSI: Digital Cellular Telecommunications System; Unstructured Supplementary Service Data (USSD) – Stage 3 (GSM 04.90 version 5.0.1). Technical Report, ETSI, May 1997

[4] Paik, M.: “Stragglers of the herd get eaten: security concerns for GSM mobile banking applications”. HotMobile '10: Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications, pp 54-59, Annapolis, Maryland, USA, 22-23 February, 2010

[5] Sarajlic, A.; Omerasevic, D.: “Access Channels in m-Commerce Services”. 29th International Conference on Information Technology Interfaces, ITI 2007, pp 507-512, June 25-28, 2007, Cavtat, Croatia

[6] Kumar, D.; Martin, D.; O’Neill, J.: “The Times They Are A-Changin’: Mobile Payments in India”. CHI ’11 Proceedings of the 2011 Annual Conference on Human Factors in Computing System, pp 1413-1422, May 7–12, 2011, Vancouver, BC, Canada

[7] Barkan, E.; Biham, E.; Keller, N.: “Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communication”. Journal of Cryptology, Volume 21 Issue 3, pp 392-429, March 2008

[8] Karunanayake, A.; De Zoysa, K.; Muftic, S.: “Mobile ATM for developing countries”. MobiArch '08: Proceedings of the 3rd international workshop on Mobility in the evolving internet architecture, pp 25-30, August 22, 2008, Seatle, WA, USA

[9] McKitterick, D.; Dowling, J.: “State of the Art Review of Mobile Payment Technology”. TCD Computer Science Technical Reports, TCD-CS-2003-24, pp 22, 2003, Department of Computer Science, Trinity College, Dublin, Ireland

[10] McCune, J. M.; Perrig, A.; Reiter, M.K.: “Seeing-Is-Believing: using camera phones for human-verifiable authentication”. International Journal of Security and Networks, Volume 4 Issue 1/2, pp.43–56, February 2009

[11] Toorani, M.; Shirazi, A. A. B.: “Solutions to the GSM Security Weaknesses”. NGMAST ’08 Proceedings of the 2008 The Second International Conference

on Next Generation Mobile Applications, Services, and Technologies, pp.576-581, University of Glamorgan, Cardiff, UK, Sep. 2008

[12] Panjwani, S.: “Towards End-to-End Security in Branchless Banking”. HotMobile ’11, Workshop on Mobile Computing Systems and Applications, ACM, March 1–2, 2011, Phoenix, Arizona, USA

[13] Panjwani, S.; Cutrell, E.: “Usably Secure, Low-Cost Authentication for Mobile Banking”. SOUPS '10: Proceedings of the Sixth Symposium on Usable Privacy and Security, Article No. 4, July 14-16, 2010, Redmond, Washington, USA

[14] ETSI/SMG: GSM 11.14 Standard, Specification of the SIM Application Toolkit for the Subscriber Identity Module - Mobile Equipment (SIM-ME) interface. http://www.etsi.org, 1996

[15] 3GPP: 3GPP TS 31.111, Universal Subscriber Identity Module (USIM) Application Toolkit (USAT). http://www.3gpp.org/ftp/Specs/html-info/31111.htm, 2011