



**Demonstration of technologies and vulnerabilities
in online web payment systems**

BSc Computer Science (G400)

By

Rayyan Iqbal

Supervisor: Professor Aad Van Moorsel

Word Count: 13,000

May 2020

Acknowledgements

I would like to sincerely thank Professor Moorsel for his continued guidance, wisdom and support throughout the creation of this project. I am especially grateful as he took on my paper and believed in me throughout this entire process.

TABLE OF CONTENTS

Chapter 1. Introduction	6
1.1 Project Overview	6
1.1.1 The Context	6
1.1.2 Project Aims and Objectives.....	6
Project Abstract.....	9
Chapter 2. Chapter Background Research	10
2.1.1 Card not present (CNP)	10
2.1.2 Card-present verification technologies	10
2.1.3 Card-not-present verification technologies.....	10
2.1.4 3 Domain Secure (3DS)	11
2.1.5 Chip Authentication Programme (CAP).....	12
2.1.6 chipTAN and iTAN	12
Chapter 3. Short Message Service (SMS) Spoofing Attack	14
3.1 European Banks	14
3.1.1 Attack Explanation.....	15
3.1.2 Attack Walkthrough	16
3.2 Security Analysis.....	18
3.2.1 Threat Modelling.....	19
3.2.2 Spoofing Attack.....	20
3.2.3 Data Sniffing	22
3.2.4 Denial of Service.....	23
3.2.5 Security Analysis Results	26
Chapter 4. STRIDE Design.....	28
4.1 Trusted Number Problem	28
4.1.2 Trusted Number Solution	29
4.2 Telecommunication Spoofing	32
4.2.1 Design Analysis	33
4.2.1 Functional Requirements.....	35
4.2.2 Non-functional Requirements.....	36
Chapter 5. Implementation.....	38
5.1.1 Technologies and Programming Languages	38
5.1.2 Program Walkthrough	39

Chapter 6. Evaluation	44
6.1 Testing.....	44
6.1.1 Testing Approach	44
6.1.2 Testing Plan.....	44
6.1.3 Testing Results	47
6.1.4 JUnit Testing.....	48
6.2 Design Requirement Assessment	49
6.2.1 Functional Requirement Assessment.....	54
6.2.2 Non-functional requirement assessment	54
6.2.3 Development Reflection	55
6.3 Results	55
6.3.1 Customer Risks	57
6.3.2 Risk affects advanced users too	57
6.3.3 Evaluation of Experiment	58
6.3.4 Future Work	58
6.4 Background Research	59
6.5 Aims and Objectives	60
6.5.1 Satisfaction of Objectives	60
6.5.2 Satisfaction of Project Aim	62
6.6 Reflection.....	62
Appendix A	65
COVID-19.....	65
Distributed Guessing Attack	66
Attack Walkthrough	66
Previous Research	66
Valid Card Generation with Luhn’s Algorithm	67
Mathematical Theory Crafting	69
Notes.....	70
Conclusions and The Future.....	70
Appendix B – Experiment Results	72
HSBC - EE	72
Lloyds Bank – Three.....	74
Barclays O2.....	76
Barclays Vodafone.....	79

Chapter 1. Introduction

1.1 Project Overview

This chapter aims to provide context to the reader by outlining project information such as aims, objectives and the set of rationale for conducting the project.

Additionally, there is an extended subsection which describes the project adjustments which were necessary following the COVID-19 pandemic.

1.1.1 The Context

Card fraud is the biggest type of fraud in the United Kingdom and has continued to grow in 2019. Fraud the Facts [1] reported a 29% rise in card fraud against UK retailers in 2018, which is an estimated £265.1 million in losses. Scarce online regulation [2] has resulted in low barriers to entry when becoming an online retailer. This has created an influx of small businesses who have suffered [3] following restricted access to capital for investment into fraud protection systems.

In the past decade, there has been a shift in consumer behaviour globally. Reports show a decrease in the number of consumers choosing to shop instore [4] and, an increase in the number choosing to shop online. Besides this, it is becoming more common for those who choose to shop in store to opt for alternative electronic forms of payment [5] such as:

- (1) A credit card,
- (2) Apple Pay, or
- (3) Bitcoin and other forms of electronic payment.

Similar shifts in behaviour have been seen in the fraud world, with less fraudsters producing physical counterfeits [1] and instead use stolen card details to process payments online. Fraud the Facts [ibam] reported a 33% decrease in the number of physical counterfeits used in store.

1.1.2 Project Aims and Objectives

This dissertation aims to illustrate the risks faced by web-based payment systems through a series of documented demonstrations.

The aim has been split up into the sequence of objectives below:

1. *Identify the primary technologies used in online payment systems and understand their technical implementation by exploring literature and scientific papers.*

This objective covers the preliminary research needed to obtain the payment system knowledge required for completing further objectives.

2. Synthesize existing research in conjunction with my own to identify a subset of vulnerabilities within the system.

This step covers the use of existing academia, in the form of research journals or papers, who have investigated a similar project to my own to help find variations of previously found and fixed vulnerabilities. From my own experience of being interested in software bugs, I have found it is a common occurrence for previously patched bugs to still be exploitable given the correct environment or new attack point. More generally, a slight modification of key input parameters may be able to bypass the newly introduced validation systems.

3. Design tools for vulnerabilities deemed high risk using threat modelling techniques

My third objective covers the use of threat modelling techniques to appropriately measure identified vulnerabilities and to design an implementation of those deemed high risk for my demonstrator. It is important to note, this design process will also explicitly investigate the implementation of an ethical solution, one which cannot be misused by malicious users later down the line. For these reasons, I will be designing with external validity in mind.

4. Implement designed tools into a demonstrator developed in Java which can demonstrate an identified attack

This objective covers the implementation and development of my demonstration tool which fits a key variable of my aim which is to demonstrate the execution of an identified vulnerability in real-time. The demonstrator will should look to provide appropriate logging to help illustrate the risks caused by a given vulnerability.

5. Conduct a series of evaluation and reflection on the produced demonstrator using evaluation heuristics

My final objective will occur nearing the end of my project where I will write a chapter reflecting on the outcome of my project, describing what went good, bad or could have went differently. Moreover, I will be showing my demonstrator to family members to evaluate how my findings have potentially changed their opinions on payment systems and if the software meets my initial expectations.

Project Abstract

This paper describes a set of analysis on the security of online payment systems, covering aspects from the technologies used by online retailers, to mechanisms implemented by banking organisations to reduce fraud online. We describe in detail a flaw in European Bank's phone-based customer verification system by conducting extensive experiments into the delivery of spoofed text messages, from a client's mobile, to the number used by the bank in the verification process. Our experiments demonstrate a method to bypass the authentication system which enforces a block on a customer's card following multiple suspicious transactions. This is a serious threat to customers who are victims of online fraud because the vulnerability bypasses a key fraud verification process used in card-not-present (CNP) transactions.

Chapter 2. Chapter Background Research

In this chapter, I will explore the existing cardholder verification technologies used in the payment eco system to observe the difficulties faced by banks and retailers globally, in safely determining if a cardholder is who they said they are when processing a transaction.

2.1.1 Card not present (CNP)

Card-not-present (CNP) transactions occur online and require the cardholder to provide information only known by the card holder. This has been a significant challenge for banks and retailers globally as CNP transactions take place all over the web, where the merchant and point of sale terminal are not in the same physical location [15] as the card and its holder.

2.1.2 Card-present verification technologies

Since the deployment of Europay, Mastercard and VISA (EMV) [16] in 2003, European countries have noticed a measurable and significant reduction [17] in face-to-face fraud. The protocol framework secures physical credit and debit card transactions by splitting customer verification into three phases:

- (1) card authentication,
- (2) cardholder verification and
- (3) transaction authorisation.

The system has seen significant adoption globally with figures from early 2008 suggesting there were over 730 [18] million EMV-compliant smart cards in circulation worldwide.

There have been difficulties in translating the authentication methods provided by EMV into the CNP channel since, there is no physical presence of both the card and its holder [17] and the point of sale terminal (POS). Consequently, new data security standards have been developed such as the payment card industry (PCI) data security standard (PCI-DSS) from the security standards council.

2.1.3 Card-not-present verification technologies

PCI-DSS aims to protect cardholder and sensitive authentication data present within the eco system and, limits its availability to fraudsters. The standard proposes 12 key

technical and operational requirements aims to provide non-EMV transactions, those taken place on the terminal or online CNP transactions, with the same fraud reduction capabilities as EMV.

These standards have seen little adoption in the CNP world as retailers believe the opportunity cost of implementing such standards does not outweigh the received benefit in reduced fraud. Consequently, the council's advice has been ignored which describes a primary account number (PANA) and its expiry date being an insufficient amount of data for cardholder verification. [ibam]

2.1.4 3 Domain Secure (3DS)

3 Domain Secure (3DS) is the most adopted and famous user authentication protocol for CNP transactions. It conducts a series of risk assessment to decide whether a payment should be challenged for two factor authentications. In October 2016, EMVCo revised the protocol for the inclusion of transaction risk assessment (TRA) and provides two options for authentication:

- (1) challenged authentication, and
- (2) frictionless authentication [19].

Understanding whether 3DS has been successful in reducing fraud online is a topic for research on its own, as researchers claim the system simply shifts liability of banks to customers whilst also undermining security principles and other anti-phishing initiatives [15].

Similar issues have been previously been raised by both customers and researchers regarding the EMV pin system where fraudulent cases reported to the Financial Ombudsman Service [20] where quickly dismissed and met with stock responses such as "Your card's chip was read and a pain was used, so you must be negligent" [15].

Further on my point regarding the system undermining existing security principles, 3DS is activated during shopping (ADS) depending on the level of risk associated with a transaction during TRA. If the risk meets the threshold, the payment is challenged, and an iframe or pop-up, both without an address bar, requests additional verification data from the user to complete the transaction. This arguably odd approach makes it difficult for users to determine if the pop-up shown is legitimate, supports secure data transfer (SSL) and more.

Recent studies have identified the system [21] could be susceptible to a man-in-the-middle (MITM) attack when a trojan is present on the victim's machine, with the ability to control all data communication flows. An attacker could replace the displayed iframe for the current transaction to one initiated by the attacker, tricking even the most-experienced users by using the same transaction amount to complete a transaction they did not start.

2.1.5 Chip Authentication Programme (CAP)

European banks have recently introduced the Chip Authentication Programme (CAP) to help deal with the soaring losses [22] as a result of online banking fraud. The technology has been described as an improvement over the previous static password approach, since some banks now request for certain characters from a character's password, or ask for them to be entered using drop-down boxes rather than the keyboard, in a bid to help resist keylogging attacks.

Customers are allocated a handheld, pocket-sized device known as a CAP reader, which is used in addition to the customer's debit or credit card for generating one-time codes (OTC) for both login and transaction authentication.

The system has received scrutiny for its optimisation centred design approach as Needham famously described optimisation to be 'the process of taking something that works and replacing it with something that almost works but is cheaper' [23] which, for our context, suggests banks are aiming to reduce the amount of information customer needs to enter and maximise backwards compatibility. Such an approach will inherently create vulnerabilities as they try to cut corners to make a product which some argue to be another tool for shifting liability towards the customer. In laymen's terms, this idea is best described as a reduction in statutory protection for electronic fraud victims.

2.1.6 chipTAN and iTAN

German banks have taken a similar, yet more secure, approach to one-time-password systems by using fingerprinted readers [24] to generate one-time authentication codes with RSA SecurID. ChipTAN and its predecessor iTAN also provide customers with a hand-held device containing a display, card slot and various optical sensors [21] on the back of the device. During a transaction, a flicker code is generated by the bank and displayed to the user, which is both unique to this

transaction and made up of five vertical bars which change their colour from black and white.

Customers will then place their device in front of the screen as shown in *figure 1* with the data being read by the device's sensors to ultimately generate a TAN number. Details of the transaction are subsequently shown on the device, as it allows for users to detect if there has been tampering, and generally includes the total amount of money in a transaction and the bank account number where the money is to be sent. In theory, even if an attacker could mess with the flicker code shown on the website, the displayed information is designed to enable customer to manually detect and prevent attacks.



Figure 1 – Scanning a flicker code with the chipTAN comfort system [25]

Researchers at the Red Penetration Team found chipTAN is still vulnerable to a MITM attack [21] as a result of the system's functionality design. In short customers can complete a group bank transfer, known as a collective transfer, in which money will be sent from one origin account to multiple recipients. Due to the device's limited display capabilities, a compromise must be made between the displayed transaction information and readability. Moreover, instead of each individual bank account number involved in the transaction being shown, the total number of recipients is shown. This means, when a customer is completing a bank transfer, an attacker could replace the shown flicker code using a MITM attack. The attack involves setting up a replica transaction using a collective transfer preventing users from seeing the recipient account number and subsequently are unable to detect tampering has occurred. [ibam]

Chapter 3. Short Message Service (SMS) Spoofing Attack

This chapter describes threat modelling analysis on the infrastructure behind the recently introduced text-based customer verification system in European Banks and outlines three ranked flaws in their approach to building a fraud protection system. We discover the highest rated threat to be a spoofed MITM attack during this verification stage which allows for this verification step to be bypassed.

3.1 European Banks

Recently banks in the United Kingdom have been providing customers with a new two-factor authentication step when there are concerns of fraudulent activity occurring on a cardholder's account. This mechanism is generally triggered following numerous suspicious transactions on a card, generally identified through an untrusted origin machine, recipient or payments in a currency differing to the card issuer.

The purpose of this system is to reduce the time taken for customers to remove blocks which may be placed on their accounts. In laymen's terms, it provides convenience through automation. Once the mechanism has been triggered, a text message is sent to the cardholder phones, as shown in *figure 2*, from the bank and the text payload contains a list of potentially suspicious transactions. At this point, the customer must decide for themselves to verify the shown transactions are recognised and initiated by themselves to subsequently:

- (1) Remove the raised account suspicion (generally a block on the card) and allow for the transactions to be processed, or
- (2) Retain the raised suspicion and disable their card until further investigations have been completed.

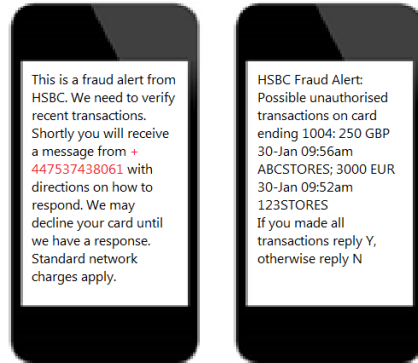


Figure 2 – Exemplar verification texts sent from HSBC using the new customer verification mechanism [26]

The number used for delivering the text is generally a number selected from a list of numbers published on the bank's website. In my paper, I will refer to these lists of numbers as the set of trusted numbers. Trusted numbers are published by banks as an anti-phishing initiative against fraudsters to help customers detect when a received text or phone call is truly from the bank. Moreover, the exemplar payloads are in place to further help customers with their detection process.

3.1.1 Attack Explanation

The basis for this attack stems from the decision by banking institutions to publicly disclose a list of trusted numbers on their site, which is a trade-off between user security and convenience. For the banks, it is in their best interests to make trusted numbers public, so that customers have enough information to detect commonly attempted phishing attacks via telephone. On the other hand, more advanced attackers can use this information to conduct increasingly sophisticated attacks through spoofing techniques which can be used to impersonate the bank.

Mobile spoofing techniques enable for an attacker to set [27] their originating mobile number, known as a sender id, in a telecommunication to one of their choice. This behaviour is generally achieved by tricking a short message service carrier (SMSC) [28] into relaying messages to other networks. From the home carrier's perspective, it appears the user has roamed into a foreign network's radius who is then requesting messages for delivery on behalf of the user.

As of March 2020, more than half of the countries globally remain vulnerable to an SMS spoofing attack but some vulnerable countries such as Hong Kong [29] can detect spoofing but still allow for the use of alphabetical sender IDs.

Mobile messaging is the largest communication medium globally with an estimated 6 billion texts being sent daily which, is a whopping 2.2 trillion [30] every year. The difficulty with the SMS platform is there is such a wide range of devices using the service in terms of their underlying hardware, computation power, battery power and user latency. Subsequently, there is not one universally accepted method of securing communications using cryptography due to the potential overheads of techniques such as private key storing on the mobile device may impose on user performance.

Instead companies have implemented their own data-encryption mechanisms using digitally signed messages to verify the authenticity [31] and integrity of the sender. Although such approaches can be successful in defending against such attacks, there has been little transition into industry and challenges faced by developers due to the messages being encrypted by the network service provider.

Theory Crafting

Applying this knowledge to the described customer verification system, *shown in figure 2*, we can see how the system may be susceptible to a MITM spoofing attack. An attacker could bypass the verification system by sending a spoofed message, impersonating the victim's phone number, to the bank's trusted number. Assuming the message is successful in propagating the SMSC, the bank would subsequently remove the placed card block as they believe it originated from the customer.

3.1.2 Attack Walkthrough

For this attack to be carried out, the attacker must already have access to the victim's card and enough sensitive data to process a payment online. Moreover, the attacker should have access to the victim's primary mobile phone number or full name, email and living address. Social engineering techniques can be used to identify the victim's phone number should it not be known initially.

To begin the attacker will attempt to purchase goods using the victim's card from online retailers until the transaction is declined but not challenged by the 3DS verification system. Assuming the attacker's payment has been declined but the card

details were entered correctly, we can deduce for one of the following statements to be true:

- (1) the card does not have enough balance to complete the transaction, or
- (2) the card has been challenged by the SMS verification system.

Since it would be very difficult for the attacker to determine which statement is true without potentially triggering additional security mechanisms, the attacker would work using the latter assumption which states the card has been challenged by the verification system.

Next the attacker needs to locate the card issuer using the card's BIN, the first six digits of the victim's card. The BIN will be used with an online database such as BinDB [14] or ExactBins [32] to find information about the card's brand, issuing bank name and type. *Figure 3* shows the results from entering my personal debit card's BIN into one of these online databases.

Bin:	475128
Card Brand:	VISA
Issuing Bank:	NATIONAL WESTMINSTER BANK PLC
Card Type:	DEBIT
Card Level:	CLASSIC
Iso Country Name:	UNITED KINGDOM
Iso Country A2:	GB
Iso Country A3:	GBR
Iso Country Number:	826
Bank's website:	www.natwest.com
Customer Care Line:	0800 200 400
Bank Address:	***** [?] Available on Ultimate Database
Formal Bank:	***** [?] Available on Ultimate Database
Commercial/Personal Card:	***** [?] Available on Ultimate Database
Additional Info:	

Figure 3 – Results from entering my card's BIN into BinDB [14]

Trusted Numbers and Finding the Victim's Mobile Number

As shown in *figure 3*, the issuing bank for my card is National Westminster Bank Plc, better known to us as Natwest. To proceed, the attacker can now crawl the web using a search engine or scraping software to identify the list of trusted mobile numbers for the Natwest bank. Luckily, my bank does not directly publish their list of trusted numbers but can still be discovered using advanced search engine queries with payloads such as 'Reply Y' to discover customer reports of receiving a verification text and documenting the sending number online.

Now the attacker must identify the mobile number of the cardholder, if this is already known then the step can be skipped. To find the phone number of the victim, public record systems such as 192.com [33] or leaked databases such as DeHashed [34] can be used to identify phone numbers relating to residents at a given address or email. In the worst cases, when none of these approaches are successful in discovering the mobile number of a victim, social engineering techniques with social media support can produce similar results.

Sending the Spoofed Text

Once all the aforementioned information is known, the attacker must now complete the final stage of the attack which is to deliver a spoofed text message. Banks will generally select one random number from a list of their own trusted numbers using a selection algorithm which picks the least busy machine. Since the attacker will be unable to determine which number was chosen, they must send a spoofed text message impersonating the victim to each of the numbers.

All the delivered texts should share the same positive acknowledgement payload 'Y' which is the standard expected response in this verification system, shown in *figure 2*, to remove the card block. Finally depending on the level of sophistication the attacker could potentially consult telecommunication logging systems to detect if any text has failed to propagate the SMSC and retry failed messages.

After the texts have been delivered, the attacker can now resubmit the previously declined transaction to have it processed without being challenged, frictionless authentication. There still exists a chance but generally a low probability the card could be challenged again, this can be prevented using a SOCKS proxy from companies such as VIP72 [35] close to the victim's origin machine and low-cost transactions.

3.2 Security Analysis

This subsection conducts a set of security analysis on the proposed two factor authentication system using the STRIDE threat modelling technique. STRIDE is a threat modelling technique developed by Microsoft [36] for identifying computer security threats by acting as an acronym for six security categories.

Commented [RI(1):

3.2.1 Threat Modelling

Each of the models shown in this section were produced using Microsoft's threat modelling tool [37] which analyses threats using data flow diagrams (DFD) to represent information flow in the system.

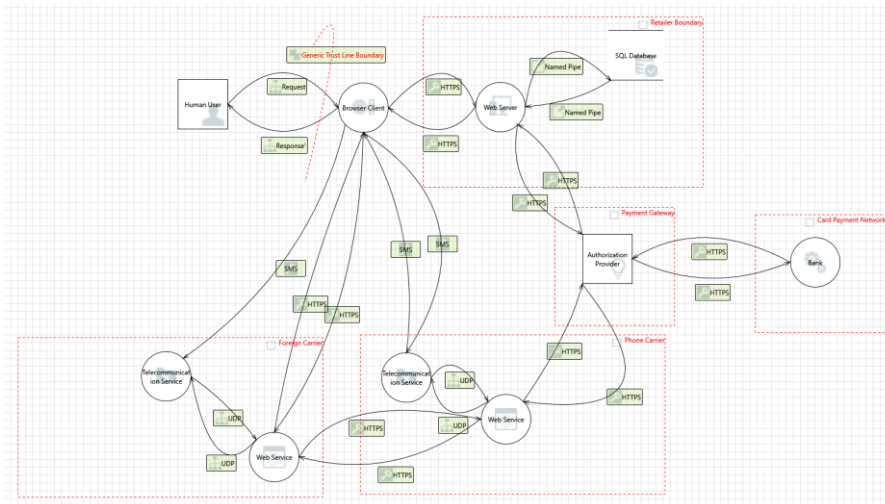


Figure 4 – DFD for an online payment system with SMSC networks

Figure 4 shows the DFD for an online payment system where a customer could be challenged by the text-based verification system. Rectangular boxes are used to represent entities, such as a human individual or organisation, whilst processes show a client's browser or web server in a circular box. Data may flow between any two entities or processes in the system and is shown using a directed arrow, with the direction in which the arrow points indicating the orientation of data flow.

Since data within the system is not isolated to a single local network, trusted borders are used to indicate when information is being transmitted across a network border or generic trust border. Trust borders are shown using a red-dotted line and can wholly contain any number of DFD elements.

Risk Measurement

To better understand the overall risk which a threat may pose to the system, I will be using an approach described by OWASP [38] which ranks threats based on two characteristics:

- (1) probability, and
- (2) impact.

Probability refers to the chance of an attack being exploited and is measured using the information needed by an attacker to perform the exploit, and the likelihood of an attack being successful. Impact on the other hand, measures the damage which could occur to the system or user. Both terms are assigned alphanumeric values in the range of high, medium and low which are subsequently associated to the numeric values: 15, 10 and 5, respectively. Each identified threat is finally assigned a numerical value and rank using a $Probability \times Impact$ calculation. [ibam]

3.2.2 Spoofing Attack

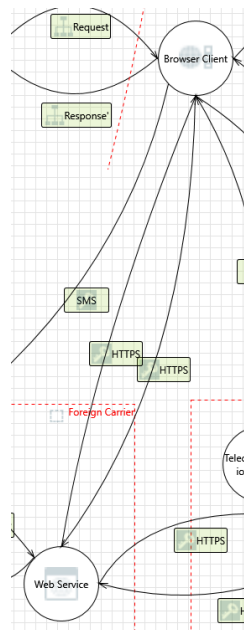


Figure 5 – HTTPS Interaction between the client and foreign carrier web service

Figure 5 shows the request and response interaction between a browser's client and the foreign mobile carrier's web service. Since the source of a client's request is not authenticated, an attacker can spoof the browser's client, leading to unauthorized access to the web service.

Attackers could use this to forge requests which change the program's execution to one of their choosing. In this case, an attacker could potentially pass a mobile sender ID which is handled by a different carrier in the request, forcing the foreign carrier to pass the request to the appropriate carrier internally who subsequently believes the request has originated from the actual client.

For an attacker to spoof the client's browser (mobile) they would require access to a foreign carrier's web service which we can assume to be likely as hundreds of online telecommunication suites such as ClickSend [39] and SendInBlue [40] who offer the service free of charge.

One downside to this attacker is that it would require the attacker to already have the client's sensitive card data before it can be successful. The SMS mechanism would not be triggered until suspicious transactions have occurred on the cardholder's account rendering the attack otherwise useless. For this reason alone, we can note some level of sophistication would be needed by the attacker to perform this attack as obtaining a client's card details is arguably a challenging task.

Granted, attackers could reduce the conditions required by purchasing card details off a black market. Regardless, this is still quite difficult for the common attacker to obtain and thus, I will be assigning a medium attack probability value.

Impact Analysis

To best understand the impact which can be caused by this attack, we must consider the worst-case scenario, where an attacker passes a forged request to change the program's execution for bypassing fraud protection systems. When a telecommunication service believes a forged text to have originated from the user, the integrity of the logging systems is tarnished which directly causes damage to the SMSC system. In addition, if the verification process was triggered and bypassed, sensitive customer information may become compromised resulting in additional unauthorised access to banking services. Granted as the system could never face complete destruction, the impact caused by this attack is also set at medium.

3.2.3 Data Sniffing

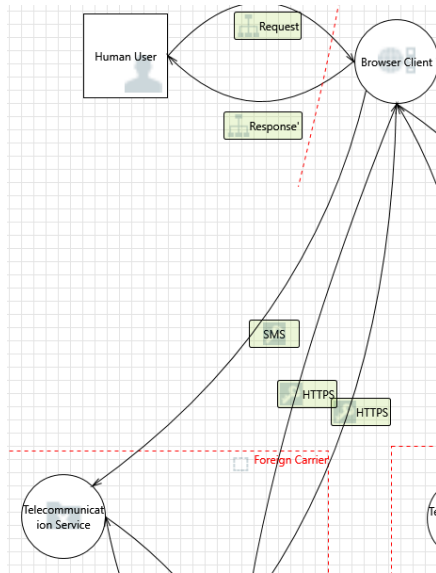


Figure 6 – SMS interaction between the browser's client and foreign carrier's telecommunication service

Figure 6 considers the SMS message interaction which can occur between a browser's client and the foreign carrier's telecommunication service. It is worth noting the data flow between the processes can only occur in one direction, from the client's browser to the communication service as foreign carriers do not have permission to send messages to a browser client which they do not manage.

When a browser client sends a text message using SMS to the communication service, it can be seen by attackers using data sniffing techniques. Depending on the encryption method used to transfer the messages, the attacker may have the ability to read the payload or contents of transmitted messages. Although messages are generally encrypted by the mobile network when in air, studies from radio companies like RTL-SDR [41] found text messages sent using GSM signals can be decrypted. In the past decade, mobile phones have moved to the newer standard 3G and 4G technologies which use more advanced cryptographic techniques like KASUMI [42] which have proven more difficult for attackers to break. Regardless, with improving computational power, we will likely see some new attack methods form which can break these layers of encryption.

Seeing the difficulties faced by hackers to decrypt encryption techniques used in cellular networks, I can confidently assign this threat with a low risk value since it would require for the attacker to be in close proximity of the victim [41] and the chances of said client using deprecated GSM signals [42] is extremely unlikely.

Impact Analysis

Regardless of the low probability, we must still consider the impact to both the individual and the system if an attacker is able to successfully decrypt and read messages transmitted between the client's browser and telecommunication server. In this event, an attacker may be able to view sensitive client banking data such as their PAN, expiry date or more could be compromised. In addition to this, personal information regarding bank statements may be sent from the bank to the user which could also be read. Subsequently, I am going to assign this threat with a medium impact rating as the compromised data may pose a threat to the confidentiality of users but not result in total system destruction.

3.2.4 Denial of Service

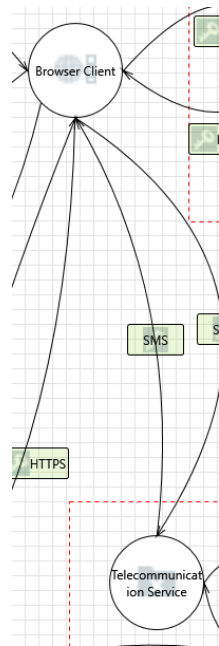


Figure 7 SMS Interaction between the client's browser and home telecommunication service

The final threat for discussion is a Denial of Service (DoS) attack during the SMS interaction shown in figure 7. In computing, a DoS attack is a cyber-attack [44] in which the attacker seeks to make a machine or resource temporarily unavailable by flooding the victim machine with increased traffic.

For the current context of figure 7, a DoS attack would look to overwhelm the telecommunication provider with a surplus of SMS traffic, potentially containing a malicious payload which is long and difficult to process, causing the service to be temporarily unavailable to the system's users.

Some systems are more prone to different types of DoS attacks depending on the level of input validation used at the attack's end point [ibam], and throughout the system. As a result, we must consider each of the system's data paths, and their data sanitation mechanisms, to understand the probability of such an attack occurring and being successful.

Insufficient Sanitation

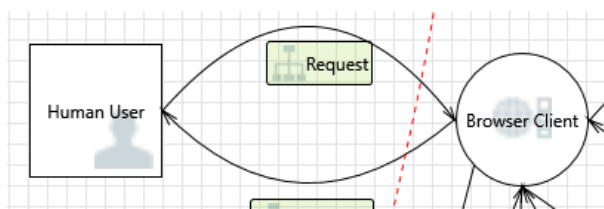


Figure 8 – Request and response interaction between the human user and client browser

Figure 8 shows where the data for the interaction shown in figure 7 originates. Data for the interaction is passed from the client's browser which takes input directly from the user. From this fact alone, we can deduce the sanitisation performed by the browser's client to be insufficient because attackers will use mobile rooting techniques [45] to access an elevation of privileges and subsequently, disable input defence mechanisms.

Probability Analysis

As for the attack's impact, researchers from the University of Berlin [45] presented a demonstration at a conference in 2010. They showed how an SMS flooding attack can be used to overwhelm and later disrupt mobile services on both individual devices and cellular networks by transmitting 120,000 SMS messages

simultaneously. The researchers most notable results included the complete suspension of a mobile device and service in a cellular network. Granted, their study has received some critique because it was conducted in isolation which, meant there was enough space in air for the messages to be transmitted quickly and at scale. Moreover, the external validity of this test is difficult to analyse as in the real world, air is heavily congested by other SMS messages and different devices using the same transmission medium.

Another study unveiled at the same conference, dubbed the “SMS of Death”, used special binary characters and overflow headers [ibam] to successfully crash older generation mobile devices such as Nokia [46] and Sony Ericsson [47]. The researchers used a single text message to trick carrier networks into continuously retransmitting the text to the recipient, eventually causing the client device to crash. This outcome provides a useful insight to our threat because it shows how exploits in a system’s validation system can be exploited to make a DoS attack require less resources for execution.

From the information gathered from these two studies, we can conclude the probability of a DoS attack occurring to be low, because the attacker must either be very sophisticated to access the number of required resources or, have an unaddressed vulnerability, known as a zero-day [48], in the telecommunication services’ input validation system.

Impact Analysis

Regardless of the low probability, we must still consider the potential system impact if an attacker is successful in carrying out the attack. The damage caused by the attack would depend primarily on the number of resources available to the attacker and can range from a temporary lag in service delivery times to a permanent loss of service, requiring maintenance from expert engineers [49] before the issue can be resolved.

When the system is overwhelmed with traffic, fault tolerance mechanisms take over and may force the service into a permanent shutdown to prevent further damage. Since one of the possible outcomes includes permanent system destruction, I will assign this vulnerability with a high impact rating.

3.2.5 Security Analysis Results

Table 1 – List of identified threats with their assigned numerical ratings

Interaction	Risk	Probability Rating	Impact Rating	Numerical
HTTPS	Spoofing	Medium	Medium	100
SMS	Data Sniffing	Low	Medium	50
SMS	Denial of Service	Low	High	75

Table 1 contains the list of identified threats with their now assigned numerical ratings derived using the described measurement model.

Highest Rated

The highest rated of the identified attacks was a spoofing attack between the client's browser and foreign carrier's web service. This threat allows for an attacker to spoof the client's browser and subsequently gain unauthorized access to the foreign carrier's web service, potentially passing forged requests which could change the execution of the program.

During my earlier analysis, I found it was possible for an attacker to send forge requests which may trick the foreign carrier's web service into delivering SMS messages as the spoofed browser client. Such an approach could be used to send positive acknowledgements when a customer is challenged with the SMS verification system and subsequently bypass this step to process a payment online.

Second Highest

The next highest rated attack was a DoS attack which used SMS flooding techniques to crash a carrier's cellular network, causing the service to be temporarily or permanently available. Although denial of service attacks are generally common with most Internet-based applications, this variation could pose as a significant threat to the availability of cellular communications in the future.

As discussed, the current limitation for this attack is the capacity of air transmissions and accessing the number of resources required to execute this attack. In the future, as devices move onto newer communication technologies, fewer devices will use air as their transmission medium, reducing congestion and making this attack easier to perform. Moreover, the discovery of new zero-day vulnerabilities happens almost

every year and if cellular networks are slow to patch known exploits in their input validation system, they may be abused by attackers in the form of DoS attacks against their system.

Lowest

The lowest and least promising attack used data sniffing to read messages transmitted between a client and their telecommunication service carrier. One of the biggest drawbacks to this approach was it requires the attacker to be within proximity of the client or telecommunication service and, have access to classified technologies which can break advanced encryption technologies used by 3G-4G signals. Although techniques exist to reliably decrypt older communication standards such as 2G, GSM signals, these technologies have become deprecated with many UK cellular networks no longer providing support.

Regardless, the data sniffing attack should be revisited in the future as computation power increases with the development of quantum computers, breaking complex cryptography algorithms becomes more practical [50] for attackers.

Summary

Following my security analysis of the online payment eco system using a telecommunication verification stage, I will continue to investigate my highest rated threat which was a spoofing attack, not only as a result of its assigned rating, but the threat which the attack poses to our security now. A vulnerability a newly introduced verification system could be catastrophic for customers if conducted at scale since it undermines the security principles introduced by banks to help protect customers from fraud.

My other identified threats in no doubt pose as a risk to the system but are more of a problem for tomorrow rather than now. To further investigate my identified vulnerability, I will be designing an application which can simulate this attack inside of my demonstrator.

Chapter 4. STRIDE Design

This chapter outlines the design process taken in creating a phone spoofing attack demonstration tool. In this chapter, I outline solutions to presented algorithm problems using design modelling techniques such as Unified Modelling Language [51] (UML) diagrams. Moreover, I outline the list of functional and non-functional requirements for implementation and allocate priority values for each of the defined requirements.

4.1 Trusted Number Problem

When the SMS verification system is triggered, the bank will randomly select a mobile number from their set of trusted numbers which then acts as the primary device used to complete the authentication process. Most banks will publish a list of their trusted numbers on their banking website to help users determine if a received text message is authentic.

The problem with implementing this scraping functionality is banks publish their trusted numbers in:

- (1) different locations on their website and
- (2) different positions on webpages, or
- (3) do not publish their trusted numbers at all.

This makes the process writing a scraping algorithm difficult since the criterion in which to select a number differs between banks. Ideally, I would like to write an algorithm which works universally, regardless of the publishing approach taken by a given bank.

In addition to this, banking websites are generally quite large and deliver high volumes of content as web pages, text and images. This means an algorithm designed to parse each individual resource for the trusted numbers may struggle to execute in polynomial time or become an intractable solution due to the rate limiting measures implemented by banks to combat DoS attacks.

This point raises another goal for this algorithm which would be to have a design which does not require the use of proxies or location-spoofing software for bypassing rate limits. Besides such an approach potentially being unethical as I should not design a program which significantly increases the web-traffic for a given bank but also there are technical problems in setting up proxies. Some devices which may

want to run the demonstrator will not provide proxy support and the application should not require specialised hardware for execution.

4.1.2 Trusted Number Solution

To help identify a solution to this problem, I am going to be considering the problem mathematically in the hope to design a solution which can be executed in polynomial time and does not require the use of proxies.

Definitions

Let x be the set of all webpages contained in a bank's website and element μ , the web page containing the trusted numbers, such that $\mu \in x$.

Intractable Solution

```
for each element in set x then
    if regex (trusted number) finds match
        write(log)
    else
        skip(page)
    fi
endfor
```

The problem with my intractable solution is the number of webpages, otherwise known as the cardinality of $|x|$ is too large for execution within polynomial time. As I have described previously, banking websites are large, generally containing thousands of different web pages. If our goal were to process each individual page, rate limiting mechanisms would take over, blocking our machine from making requests to the sight.

Although the time complexity for this algorithm is not a problem $O(n^2)$ due to the sheer size of our input array, such an approach is just not suitable. Instead, I will look to use optimisation techniques which can reduce the size of x .

Optimisation

One way we can reduce the size of x is by creating a new set y , such that $y \subset x$, is a subset of x but $\mu \in y$ remains true. In laymen's terms, create a subset from the set of webpages x such that the web page(s) containing the trusted numbers still exist within the new subset.

To achieve this behaviour, we would need a way to identify a new list of resources from a bank's website but ensure the number set is contained within these resources. One way this can be done is by using search engine techniques to only identify web resources which contain a phone number. It is worth noting not each of the numbers shown on a banking website are 'trusted numbers' as some are simply numbers for customer support or the telephones of their physical banks. As a result, further optimisations could be made to only identify webpages which contain a phone number or refer to the SMS verification mechanism.

Further optimisations would be necessary since there are hundreds of formats banks could use to represent a mobile number. Search engines such as Google do not allow for regular expression querying which would be required to identify such a specific subset. For this reason, we must also consider web pages which simply refer to the verification step as the trusted numbers could be represented in a different format to expected.

Final Algorithm

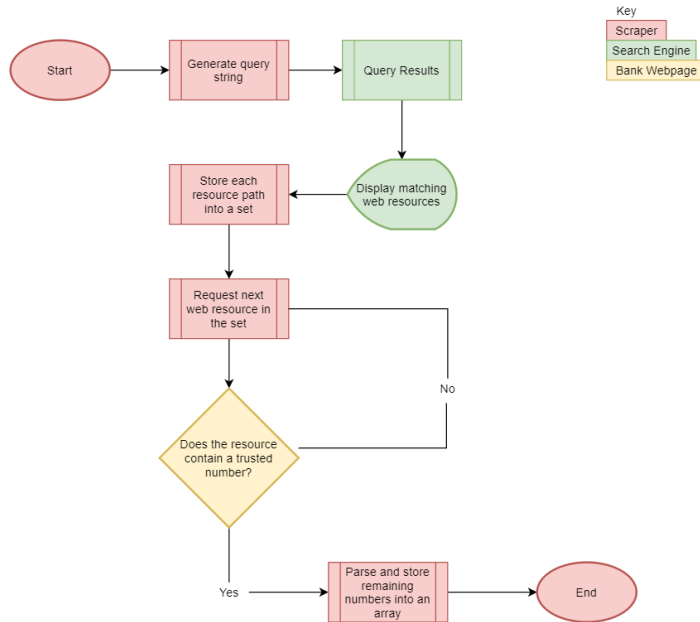


Figure 9 – Flow Chart displaying the optimised scraping algorithm, created using draw.io [52]

Figure 9 is a flow chart representing the steps taken by the involved components when executing the algorithm. Entities are indicated using the colours from the key, shown in the top right corner of the diagram, to show where operations will take place in the system. For example, the scraper demonstration tool generates a query string for the search engine and this request is subsequently processed by the search engine.

Optimisation by reducing the number of web pages significantly improves the algorithm's performance but would be difficult to describe in terms of Big-O as it is impossible to estimate the new size of x . Regardless, it is true the time complexity of the optimised algorithm is $< O(n^2)$ and thus, an improved solution.

Drawbacks

There are some downsides to my presented solution because it works on the assumption banking websites publish their trusted numbers. When the trusted

numbers are not available directly from the bank, the scraper will be unsuccessful in scraping the data. From my own crawling tests, the numbers can generally still be discovered through browsing social media platforms and mobile database to see customers mentioning receiving the verification text.

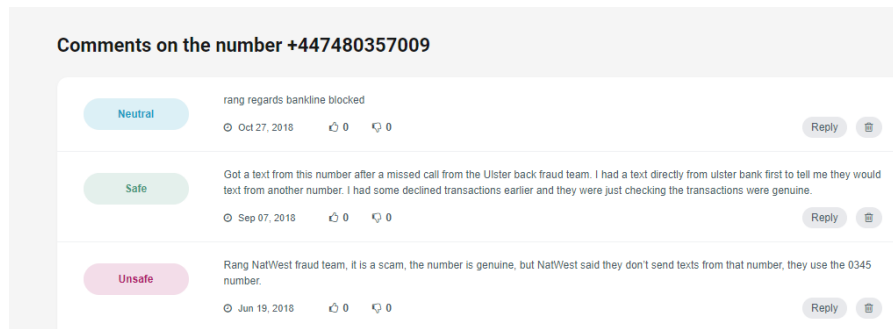


Figure 10 – Manual discovery of a Natwest trusted number [53]

Figure 10 shows Natwest's primary trusted number which I personally have received verification texts from and identified through social media crawling queries. Regardless of potentially missing some trusted numbers, writing an algorithm to scrape all this data is simple unnecessary as it may lead to texts being sent to genuine customer numbers which could be unethical.

Additionally, the algorithm may be unable to scrape each of the bank's trusted numbers every time due to differences in formatting used on their websites. Regardless the goal for the algorithm is simply to illustrate the risks faced by this vulnerability and I believe it is satisfactory in displaying external validity. It is probably for the best my application to not be perfect as it may lead to misuse.

Conclusions

The optimised algorithm will work by using advanced search engine queries to reduce the number of web resources for scraping and individually parse each page using regular expressions to identify a bank's list of trusted numbers.

4.2 Telecommunication Spoofing

My next design challenge is to create a system which can deliver spoofed SMS messages. As mentioned, to bypass the verification, an attacker must send a spoofed SMS message with the sender ID set to the victim's mobile number to each in the trusted number set.

One approach would be to setup my own foreign carrier locally by purchasing the necessary hardware to create a cellular network. Since this is my undergraduate degree, I will not be taking this approach as it is a challenging process to become an entity trusted by popular cellular networks and costly with no funding.

Alternatively, we can use our security analysis to understand telecommunication carriers provide a web service which is designed to help customers deliver texts who may not physically have access to their phones. This service is also seen to be the root of this exploit since there is no authentication system in place to ensure a customer owns a given phone number.

So, the solution to this problem is relatively simple, I will be using an online marketing platform known as cSpoof [29] as I am in good contact with the companies owner and he has granted me permission to use his platform for research purposes. One of the main benefits to using an existing network provider such as cSpoof is that their infrastructure is already trusted by major cellular networks such as EE, Three, Vodafone and more, making the propagation process through the SMSC easier.

4.2.1 Design Analysis

Finally, I will look to design the correct aesthetic for my system by using human computer interaction principles like Jakob Nielsen's usability heuristics [54] to identify a design space and provide the rationale for my choices.

System Goals

Let us start off by identifying the aim for my demonstrator which, is to demonstrate an identified vulnerability by simulating an attack and displaying insightful results to the user. From the application's aim alone, we can derive the key heuristics driving my design approach:

- (1) Stay consistent in presentation and behaviour throughout to ensure a demonstration's learning objective is clear and easy to understand
- (2) Be familiar in appearance and operationally so users who have used similar systems in the past do not need to go through a learning stage before using the demonstrator
- (3) Reduce memory load by storing inputs as it ensures the reliability of produced results by removing elements of human error allowing for users to focus on the application's results versus usage.

User Input

One of the most important design aspects for my application is user input. Currently, the plan is for the system to request the user to input the cardholder's bank name so that the list of trusted numbers can be appropriately scraped from the bank's website. As explained in my algorithm analysis section, the generated search engine query is a key factor in determining if and how many trusted numbers will be located.

Therefore, to reach the highest probability of success, the user must enter in the bank's name correctly as bank names such as the Royal Bank of Scotland are often abbreviated to acronyms such as RBS or, may simply just make a mistake when entering in the name. Minor errors may even result in the algorithm being unable to find a bank's trusted number set even if it is published on their website.

As a result, I have chosen for the bank's input name to be entered using a drop-down because it improves the reliability of my system and helps reach the design goal of reducing memory load. Engineering my system to only allow for users to select from a list of hard-coded bank names will help ensure the algorithm only provides results which are optimal but also, helps prevent against users who may seek to misuse my application.

One downside to this solution, however, is that it would require for me as the developer to push updates when new banks are added to the program. For this reason, I will only be supporting some of the most popular UK Banks such as Natwest, Lloyds, HSBC and more, ignoring smaller banks such as Virgin Money. Regardless of this drawback, my application should still be successful in reaching the project's initial aim which is to simply demonstrate the phone spoofing vulnerability.

4.2.1 Functional Requirements

Table 2 – List of functional requirements for the phone spoofing demonstrator

Identifier	Name	Description
F1	<i>Scrape trusted numbers</i>	The system must be able to scrape the list of trusted numbers from a given bank's website
F2	<i>Deliver spoofed texts</i>	The system should be able to deliver messages to any UK mobile number, beginning with a +44 prefix, and allow for a user-defined sender ID.
F3	<i>Produce results</i>	The system will produce result and display results to a visual log about the application's operations, for example the status of delivered text messages and identified trusted numbers.
F4	<i>Export results</i>	The system must allow for users to export logging results externally, allowing for the long-term storage of a simulation walkthrough.
F5	<i>Provide support for major banks in the UK</i>	The system should provide support for at least 5 of the most popular UK-based banks [55] implementing the text-based verification system.
F6	<i>Scraping execution</i>	The time elapsed for the set of trusted numbers to be scraped for a given bank should not exceed one minute.

Table 3 – List of functional requirements priorities

Identifier	Priority
F1	High
F2	High
F3	High
F4	Low
F5	Medium
F6	Medium

Table 5 contains the list of assigned priorities for each of my functional requirements. Every requirement is assigned a rating in the range of High, Medium or Low to help me decide which implementation elements should be focused on given development constraints such as time.

4.2.2 Non-functional Requirements

Table 4 – List of non-functional requirements for the phone spoofing demonstrator

Identifier	Name	Description
NF1	Executable on machines running Windows 7 or above.	The system should be consistent in its execution on machines running the Windows operation system, version 7 or above, and require access to the Java Virtual Machine.
NF2	Utilize threading for increased performance	The system should aim to utilise threading techniques to help distribute processing and subsequently, prevent external computations from affecting the main program's performance.
NF3	Input validation	The system should include validation mechanisms to prevent users from entering data which may cause the program to run incorrectly.
NF4	Enforce restrictions	The system should seek to restrict the potential for application misuse by enforcing restrictions on input fields such as a bank's name and sending phone number.
NF5	Timestamped Logging	The system should provide logging in real-time with associated time stamps to help improve a user's awareness when operating the program and the system's data integrity.

Table 5 – List of non-functional requirement priorities

Identifier	Priority
NF1	High
NF2	Medium

NF3	High
NF4	High
NF5	Low

Chapter 5. Implementation

In this short chapter, I describe the technologies selected for development and provide rationale for each of my choices. Besides this, I document a simple screenshotted program walkthrough which shows functionality of the demonstration application.

5.1.1 Technologies and Programming Languages

I decided to develop my demonstrator using the Java programming language because it has been one of my strongest languages during my time at university and provides support for gradle which allows for the easy implementation of third-party libraries. Moreover, since I was aiming to create an application which is robust, previously knowledge of the in-built testing software JUnit pushed me towards developing in Java.

Selenium

The primary technology used in my application is Java Selenium, which is a portable web automation framework [56] which provides and enables the automation of web browsers. One of the reasons I chose to use Selenium was the excess of documentation which made the initial process of learning how to use the framework easier. Besides this, the framework inherently provided support for Java when working with a gradle project, so I just felt for it to be the right option.

Java Swing

To create the graphical user interface (GUI) for my system, I used the Java Swing library [57] which is part of the foundation classes for Java. In my first year of university, I had hands on experience with this library and was confident in my ability to create an application which had a simple yet operational interface.

It is important to highlight the goal of my application is to demonstrate and provide information for my users which is why I have avoided the use of flashy art and complex colour schemes. Since the application had limited design goals, I believed Java Swing to be the best approach because I could achieve a simple design without compromising performance from excessive imports.

Spoofed Messages

To handle the delivery of spoofed SMS messages, I created a system which is based on a request and response architecture to handle their delivery. The company providing me with access to their telecommunication infrastructure, cSpoof, has a private application programming interface (API) which allows for me to request the delivery of SMS messages using web requests.

One reason why this is beneficial for my project is it meant I did not have to design an additional automation element to submit text requests on the cSpoof website. Granted, there is no in-built java library which made this development easy and so my implementation is not perfect.

5.1.2 Program Walkthrough

When the program is started, the GUI for the demonstrator appears and requests for the user to select a bank for scraping. Figure 3 shows the initial interface shown and the available bank input field's combo box.

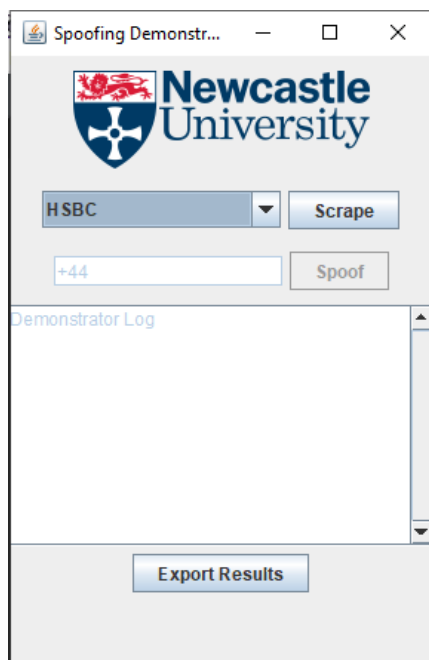


Figure 11 – Screenshot of demonstrator application on program start

After the user has selected a bank in *figure 11*, the bank name input becomes unavailable as the automation browser Selenium opens and uses the Google search engine to identify a list of possible trusted number hosts. For each identified candidate, the bot will seek to parse the page's content for a trusted number and subsequently documents them to the demonstrator log in real-time.

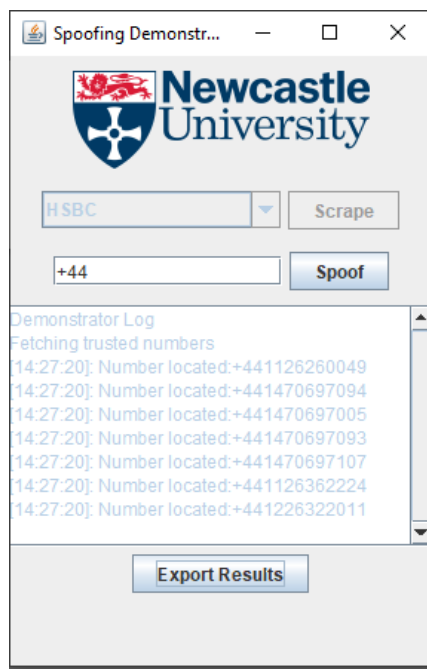


Figure 12 – Screenshot of the demonstrator after scraping has completed

Figure 12 shows a screenshot from the demonstrator after the scraping process has been completed. In this example, all the unique numbers were located on a single page which, giving them an equally assigned timestamp. Now the program requests for the user to enter a UK mobile number starting with the +44 prefix.

Incorrectly entered phone numbers are prompted with an error message displayed in the console log, as shown in *figure 13*.

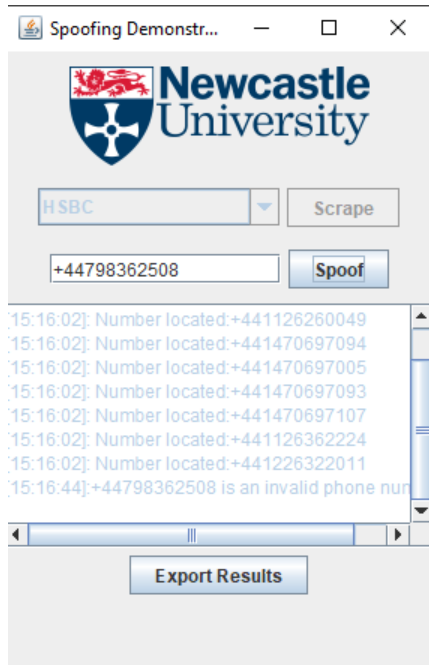


Figure 13 – Screenshot of the demonstrator when an incorrect phone number is entered

Once a valid phone number has been entered, the application attempts to send a text message from each of the trusted numbers to the entered mobile. The spoofing aspect of my vulnerability has been implemented in reverse to prevent any misuse. Results regarding the status of a message's SMSC network propagation are posted to the demonstrator log, as shown in *figure 14*.

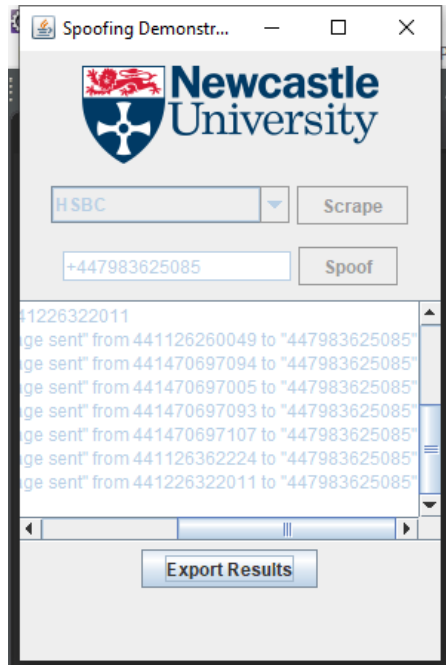


Figure 14 – Screenshot from the demonstrator after a valid phone number is entered

In the example above, the phone number entered is my own and within a few seconds I received a text message from each of the identified trusted numbers containing the payload Y. Figure 15 shows my phone's locks screen following the receipt of the texts.

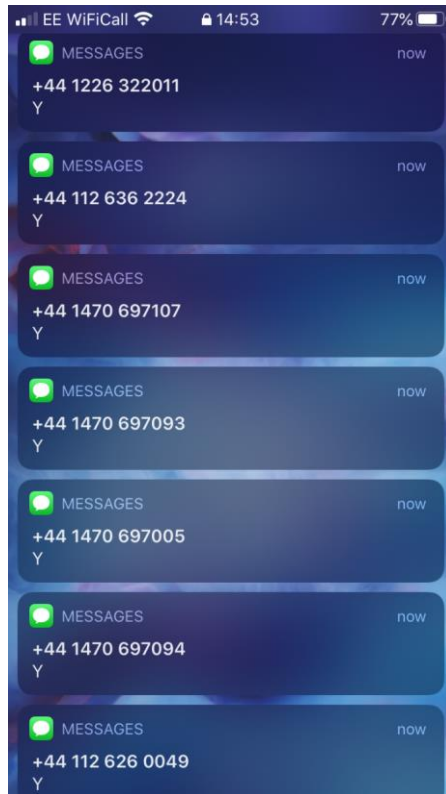


Figure 15 – Screenshot of my phone's lock screen following the delivery of the spoofed texts

Chapter 6. Evaluation

6.1 Testing

In this section, I described the testing which was completed on my application to ensure the developed application met the outlined requirements and expectations. I should note not all testing is documented in this section since I am limited by word count but would still like to document important tests I conducted.

6.1.1 Testing Approach

Throughout the development of my application, I completed a series of tests throughout to ensure the system produced is operating as expected, reliable and meeting the defined design requirements. I used two main types of testing for my application:

- (1) **Black box testing** – Testing the workings of a system component without looking into its internal details, generally seeing if an expected output is produced when given a certain input.
- (2) **White box testing** – Testing the internal workings of a system's component i.e. ensuring a routine itself takes the correct approach to producing the right output

One way in which I made sure to have a robust approach to testing was by using three different types of data designed to make my program run correctly, fail and throw an error. When the system is unsuccessful in passing a test, it allows for me to identify the problem and make the necessary code adjustments to fix the problem, meaning my produced code will be more reliable.

The three different types of test data used were:

- (1) **Normal data** – Data which is expected by the system and should generally be successful in processing.
- (2) **Boundary data** – Data values which includes maximums and minimums of conditional borders i.e. just inside/outside boundaries, typical and error prone values.
- (3) **Erroneous Data** – Data which is designed to make the system fail or throw an error when entered

6.1.2 Testing Plan

Table 6 – List of tests for my demonstrator application

#	Test	Input	Expected Output
1	Does the application successfully deliver text messages with a spoofed sender ID?	Bank Name: HSBC Phone: +447983625085	Text received from each of the trusted numbers containing the payload 'Y'
2	Does the application write operational information to the demonstrator log in real-time?	As above.	Text regarding the application's operation should be placed to the demonstrator's log in real-time.
3	When results are exported, does the content of the output file match that shown in the demonstrator's log?	Bank Name: HSBC	The outputted file should have contents which match the data shown in the demonstrator's log.
4	Are the trusted numbers for a given bank always scraped in under one minute?	Bank Name: HSBC, Lloyds, Natwest	The run-time for the scraping operation should never exceed one minute.
5	When a user enters an invalid phone number, is it detected by the application's input defence mechanisms?	Boundary Data +4479836250855 +44798362508 Erroneous Data hello 4447983625085 4479863 25085	The system should not crash or process any of the entered mobile numbers as they are invalid.
6	Do the generated request parameters for the web-based API fit the requirements for a HTTP(s) request?	'GET' request Key = value Key2 = value2	?key2=value2&key=value

6.1.3 Testing Results

Table 7– Evidenced results from the documented testing plan in table 8

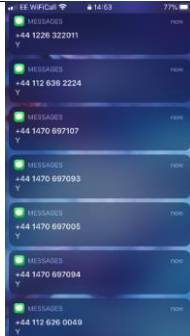
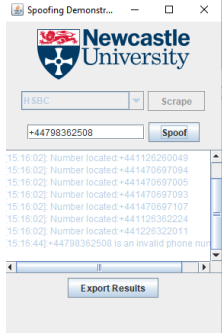
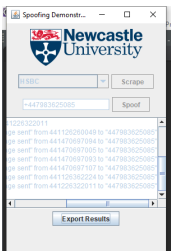
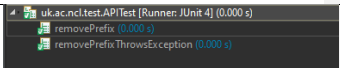
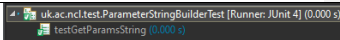
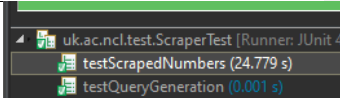
Test	Actual Result	Evidence	Passed?
1	Text message was received from each of the identified trusted numbers.		
2	The application writes operational information about the scraping and spoofing process to the demonstrator log.		
3	The log's content matches that of the output file exactly.		

Table 8 – List of the JUnit testing classes and their results

JUnit Class Name	Purpose	Results
APITest	Tests functionality from the API class which is the request response architecture for delivering spoofed text messages.	 uk.ac.ncl.test.APITest [Runner: JUnit 4] (0.000 s) removePrefix (0.000 s) removePrefix.ThrowsException (0.000 s)
ParameterStringBuilderTest	Tests the functionality of a helper class known as Parameter String Builder which is used for the API framework	 uk.ac.ncl.test.ParameterStringBuilderTest [Runner: JUnit 4] (0.000 s) testGetParamsString (0.000 s)
ScraperTest	Provides performance and reliability testing on the trusted number scraping feature of the system.	 uk.ac.ncl.test.ScraperTest [Runner: JUnit 4] (24.779 s) testScrapedNumbers (24.779 s) testQueryGeneration (0.001 s)

6.2 Design Requirement Assessment

Table 9 – Colour coordinated key to determine if a requirement has met expectations





Not met	Partially Met	Met	Met above and beyond expectations
			

Table 10 – Evaluation if proposed design requirements have met expectations.

Requirement Identifier	Reasoning	Successfully met expectations?
F1	Expectation has been met because the system can successfully set a set of trusted numbers from a bank's website.	
F2	Expectation has been met above and beyond the proposed requirement because messages can be sent to all phone numbers beginning with a '+44' prefix, even if the post-fixed number is not from the UK.	
F3	<p>Expectation has only partially met the proposed requirement because the application produces results about the system's operations i.e. scraping trusted numbers and attempting to deliver texts.</p> <p>However, due to a constraint invoked by the chosen telecommunication carrier (cSpooF), results produced do include information about text bounces and 'true' SMS delivery statuses.</p>	

F4	Expectations have been met as described since the demonstrator can export the log's content to the user's file system for long-term storage.	
F5	Expectation has been met with the system supporting five of the most popular UK banks implementing the phone-based verification system.	
F6	Expectation has been met above and beyond with the average time taken for scraping numbers being less than 30 seconds. This is nearly twice as fast as the proposed requirement of one minute.	
NF1	Expectation has been met to expectations with the program executable on any machine running Windows 7 or above with access to the Java Virtual Machine (JVM)	
NF2	Expectation has only been partially met because the system uses blocking threads for increased performance. Although processing has been delegated amongst threads, the invoked threads are blocking meaning processing on the main thread cannot	

	<p>execute until the invoked threads have finished running.</p> <p>One way this could be further improved would be using asynchronous threading which do not block the main thread in its execution.</p>	
NF3	<p>Expectation has been met above and beyond expectations as there is a strong real-time validation system in place to sanitise input data. A strong validation system helps prevent the program from being misused i.e. to send spoofed text messages with an alphabetical sender ID or used to deliver text messages to countries outside of the UK.</p>	
NF4	<p>Expectation has been met by restricting the number of banks which can be selected by the user to prevent a malicious system user from attempting to use my program for an attack in the real-world. Granted, I believe there exists room for improvement by only allowing for texts to be delivered to verified contacts. If implemented, this would further prevent users from</p>	

	using the program to potentially perform an attack and thus, further reduce potential misuse.	
NF5	<p>Expectation has been successfully met since the demonstrator provides real-time logging to the demonstrator about operational information.</p> <p>There is, however, room for improvement because currently the system uses blocking threads which cause for the timestamps assigned to scraped numbers to be slightly inaccurate.</p> <p>When multiple trusted numbers are found on the same web resource, the timestamp given will be identical for each number found on the page.</p> <p>Besides this, selecting a telecommunication carrier who can provide more propagation data would allow for enhanced text delivery logging.</p>	

6.2.1 Functional Requirement Assessment

Extending on the information provided in table 10, two of the functional requirements were met above and beyond expectations meaning extensive features have been implemented in addition to the proposed functionality.

One of the requirements, F3, was assigned a high priority but failed to meet the proposed expectations following constraints imposed by the selected telecommunication carrier. At the start of my project, I was planning on working with a company known as ClickSend [X], who initially, was willing to assist me with my project and had access to significant propagation data which would have been beneficial in helping me meet my project's aim of illustrating a vulnerability.

Later down the line, they changed their mind and so I had to quickly find a new provider cSpoof. One of the main reasons for working with this company is that I already was in partial contact with the site's owner which made it easy for me to build the necessary relations when proposing my idea. One significant drawback to working with this company, however, is their infrastructure does not have access to propagation data meaning I would struggle to complete further experiments i.e. to see if cellular networks are filtering certain messages instantly versus needing to use a phone and wait until a text is received. Sadly, due to the imposed time limitations and unforeseen changes by ClickSend, I stuck with cSpoof until the end of my project.

6.2.2 Non-functional requirement assessment

All the non-functional requirements met the proposed expectations except for one, NF2, which was sadly only partially met following time constraints during development. Following the extenuating circumstances described in my appendix, changes to my working condition meant I had fallen behind my initial schedule and resulted in me having less time to invest in building a robust application.

Looking back, I was happy to have assigned priorities to requirements since NF2 was only a medium priority objective meaning it would not have significant impacts to my project's aims or objectives if not met. In the future, I may still look to complete this objective as the implementation of asynchronous threading would increase the fluidity of my application and make it more user-friendly as the demonstrator's responsiveness seeks to increase.

6.2.3 Development Reflection

During my development process, I felt the design stage to be most crucial in creating an application which was successful in demonstrating an identified attack. Designing algorithmic solutions to problems ensured when I came around to completing my implementation there were no unforeseen problems for example the code could not be demonstrated in polynomial time.

When it came to hitting objectives I decided it would be in my best interest to use an optimised algorithm, one which may not always produce the best or most correct results, but a result which can still be used for the purpose of demonstration. Looking back at my objectives, my fourth objective stated the tool should 'demonstrate an identified attack' which I feel has been achieved by my approach.

The current approach taken for the phone spoofing implementation allows for the user to understand the underlying premise of an attack and the risks which it would impose to their security whilst making online payments. Besides this, if we consider the optimised approach from an ethical standpoint, it is in my best interests to make sure my application can never be used to potentially cause harm in the future.

Sadly, due to the current conditions of our world, it has been difficult for me to complete an evaluation heuristic such as a cognitive walkthrough on my developed application. Although through the display of my application to family members, without intervention, the application has been relatively successful in meeting the criteria specified by my project's fourth objective.

One mistake which I made during my development was not including a background section inside of my actual demonstrator application since it would assume the user has read my current paper or the appropriate payment systems knowledge in advanced to understand the illustration. So, I feel the objective could have been reached more successfully had the application been extended to include an initial background section.

6.3 Results

During my project, I investigated the real-world application of the phone spoofing attack by conducting a controlled simulated attack against four of the largest cellular network providers in the UK. The purpose of this experiment was to both investigate

the external validity of my identified vulnerability and try to relate back to my project's aim by demonstrating how the attack could be seen in application in the real-world.

Table 11 – List of tested cellular networks and associated banks

Cellular Provider	Associated Banks
EE	HSBC
O2	Barclays
Vodafone	Barclays
Three	Lloyds

Table 12 – Results from the experiment

Cellular Provider	# of Trusted Numbers	# of Received Texts
EE	7	7
O2	2	2
Vodafone	2	2
Three	3	3

Table 11 contains the list of cellular networks with the associated banks selected for the attack. Meanwhile, Table 12 contains the experiment results documenting the number of successfully scraped trusted numbers and received spoofed texts from the client's machine. Evidence of this experiment can be found in the appendix of this document.

During this experiment, texts were delivered to the client's machine with the sender ID set to a trusted number. This was done to ensure my experiment did not have any unintended side effects for example, potentially triggering another fraud protection mechanism.

For all my conducted tests, each device received a spoofed SMS containing a positive acknowledgement payload from all the scraped trusted numbers. This suggests UK cellular networks do not implement a filtration system to prevent against spoofed numerical sender IDs and demonstrates the increased validity of my claims.

Since there is no filtration system in place, this means a spoofed text message could be sent from the client's phone to the trusted number and subsequently attempt to bypass the in-built verification system. In the context of a real-world scenario, when a credit or debit card has been stolen and used for online purchases, attackers can bypass the two-step verification system in place to guarantee the authorisation of a CNP payment.

6.3.1 Customer Risks

As we have concluded in my experiment, I proposed it is plausible for an attacker to bypass the two-factor authentication system implemented by banks which would allow for the authorisation of fraudulent transactions. One significant problem raised by this vulnerability is that it directly places the liability for a transaction into the customer.

From the bank's point of view, to bypass this verification system, an attacker would need to have physical access to the victim's phone and since a customer will likely have had continued access to their device, banks will be reluctant to side with the customer. How could a customer report a fraudulent transaction which they have confirmed using their own device was initiated by themselves?

As we have seen in the past, when newer security systems are developed and flaws are reported, the Financial Ombudsman Service is quick to side with banks and view cases where one could argue customer negligence with stock excuses such as 'You have acknowledged the payment through text so, you have been negligent'.

6.3.2 Risk affects advanced users too

One important aspect of my illustration is understanding the risk which this vulnerability poses to customers of all skill levels. Customers who are familiar with making CNP transactions have likely seen this phone-based verification system in the past and when challenged, are quick to reply with a positive or negative acknowledgement text message.

The problem for customers is that my experiment found for the spoofed messages to be received within seconds of their sending being recorded in the demonstrator log. Therefore, even if an on-the-ball customer, detected an unauthorised transaction and replied with a negative acknowledgement text to the verification process they would likely lose out by the speed and accuracy of a computer. As described in my testing

section, the average time taken to conduct this attack is less than 30 seconds giving customer's limited time to both detect and respond to an unforeseen situation.

Besides this, if a customer were to reply with a negative acknowledgement text after the computer has already delivered a positive acknowledgement, the automated system would stop listening making the customer believe the transaction has been cancelled. When... in reality, the computer has already won the race and an attacker was given authorisation to complete the transaction.

6.3.3 Evaluation of Experiment

This experiment was quite successful in satisfying the aim of my project which was to illustrate the risks faced by payment systems through a series of demonstrations. Post-experiment analysis allowed for me to demonstrate the threats which are faced to the different parties involved within the eco system, for the customers there is an increased risk of being susceptible to CNP fraud and diluted liability. As for the banks, there is an inherent risk of increased disputes for fraudulent transactions.

By completing a manual study, it helps to further exemplify the external validity of my project since it attempts to provide a real-world simulation which may provide readers with a better understanding of the attack's context.

6.3.4 Future Work

The focus of my project was to demonstrate this text-based verification system bypass used by European Banks. My attack worked on the premise that it was possible to deliver an SMS message which has a sender ID spoofed to the victim's mobile number. It is important to understand the ability to deliver spoofed messages is a vulnerability which could be explored further to identify different flaws in banking systems.

There is no doubt a chance some banks are providing additional services which can be operated via text message. For example, smaller banks who may not have the necessary capital to develop a mobile application could use an automated texting system to setup transfers or view personal account information such as balances and previous transactions. I believe additional research needs to be conducted into

different ways spoofing can be used against banks to help further identify problems with their approach.

Besides texting, I have not even begun to investigate the possibility of conducting spoofed calls to bypass further verification mechanisms. From my own experience, a previous bank of mine allowed for me to authorise suspicious transactions by ringing an automated hotline given I could provide some personal information such as my date of birthday and rang using the mobile number linked to my bank account.

American Express, a popular American bank, for example, does not implement a text-based verification system and instead is solely reliant on manually calling customers to remove card blocks or wait for them to remove blocks using their automated system. Generally, customers are required to input sensitive details such as their social security number (SSN) or a personal security pin. Like my own vulnerability, this information can generally be purchased through black markets which undermines the security of such systems.

6.4 Background Research

The second chapter of my dissertation conducts background research into payment systems by investigating scientific papers and research journals. This stage was crucial in helping me understand the different technologies used in online payment systems as I investigated their underlying technical details for example, the EMV protocol used in Chip and Pin payments. Additionally, reading papers which investigated vulnerabilities in payment systems helped me understand the methods and approaches used by researchers when trying to identify security flaws.

One piece of work which was particularly motivational for me was by the Red Penetration Team [21] which presented a flaw in the German bank's chipTAN comfort system, a handheld fingerprinted reading system designed to reduce fraud online. Their work presented a simple MITM attack which changed payment details before received by the client and, bypassed several fraud protection mechanisms put in place by the bank.

The first objective for my project was to identify technologies and their technical implementations which I believe has been met to a satisfactory level following my background research. After reading hundreds of papers, many which are not documented, my knowledge about payment systems has become much stronger and I feel confident in discussing technical details at a high level.

An approach I took to help me reach this objective more easily was by maintaining a reading document which kept crucial findings and quotes from read papers so that in the future, when I needed to cite or check a fact, I referred to my own organised document which made the process of finding the answer much easier than manually researching it. This was also beneficial in achieving working towards my projects second objective as it became easy to draw comparisons and contrast the work of different researchers.

Although I do remain partially disappointed following the adjustments were made to my project following the pandemic. Before the pandemic, my project was exploring an increased number of topics then documented in this paper which subsequently resulted in me investigating additional areas of payment systems which are no longer relevant to this paper. Regardless, I do not find this has taken away from meeting this initial objective.

There is discussion about these additional topics contained in Appendix A and the changes which were made to my project following the global pandemic.

6.5 Aims and Objectives

6.5.1 Satisfaction of Objectives

Table 13 – Table containing an assessment into if objectives have been successfully met

#	Objective	Explanation	Met?
1	Identify the primary technologies used in online payment systems and understand their technical implementation by exploring literature and scientific papers.	<ul style="list-style-type: none"> Background research section completed in the preliminary stages of my project helped increase my knowledge of card payment protocols, technologies and their underlying technical implementation 	

2	<i>Synthesize existing research in conjunction with my own to identify a subset of vulnerabilities within the system.</i>	<ul style="list-style-type: none"> • Utilisation of existing research the security of one-time-passcode systems such as chipTAN • Development of threat models for a telecommunication verification mechanism based on previous payment model assumptions
3	<i>Design tools for vulnerabilities deemed high risk using threat modelling techniques</i>	<ul style="list-style-type: none"> • Creation of algorithms for identified implementation problems using mathematical analysis and UML modelling techniques • Interface design based on human computer interaction principles • Techniques from OWasp used to assign threat levels for identified vulnerabilities to help deem those which are 'high risk'
4	<i>Implement designed tools into a demonstrator developed in Java which can demonstrate an identified attack</i>	<ul style="list-style-type: none"> • Development of a tool in Java which uses a web automation framework to simulate the phone spoofing attack • Real-time logging systems which provide operational information to illustrate the attack's progress
5	<i>Conduct a series of evaluation and reflection on the produced demonstrator using evaluation heuristics</i>	<ul style="list-style-type: none"> • Experiment conducted using the demonstrator in a real-world scenario with associated evaluation of risks to different parties

- Assessment of whether the demonstrator has helped to satisfy the overall project's aims and objectives
- Sadly, no evaluation heuristics such as a cognitive walkthrough were conducted following difficulties in locating participants with the pandemic

6.5.2 Satisfaction of Project Aim

"Illustrate the risks faced by online payment systems through a series of demonstrations"

The aim of my project has been met successfully following the development of a tool which can demonstrate an identified phone-spoofing attack. Users can use my tool to conduct a simulation attack which provides logging information about the a

The aim of my project has been met successfully following the development of a tool which demonstrates my identified phone-spoofing attack. Users can use my tool to conduct a simulation of the attack and receive demonstrational information in the application's logging software. Produced results can be further exported for individual analysis and understanding.

Moreover, the defined experiment with its associated risk analysis to involved parties helps provide the specifics for who is at risk and the different risks which are faced by clients and banks.

6.6 Reflection

I have gained a strong understanding about the different parties, protocols and security mechanisms which exist within the world of payment systems. Besides the background research conducted for my project, my own exploration into identifying different threats has made me much more confident in taking a practical approach to discover security flaws inside of a system.

Throughout this project, I have grown both individually as a person and in terms of my own knowledge. Working on a long-term project such as this requires significant

planning and dedication, skills of mine which have improved directly from the hours which I put into writing this paper.

My understanding about the different parties, protocols and mechanisms which exist in the world of payment systems has increased substantially and I have developed a passion for the industry, hoping in the future to write my master's thesis on a similar topic. Before writing my paper, I was always interested in payment systems but now I feel like I can confidently take a theoretical approach to identifying security issues for a given system. It feels like a learning curve, necessary to understanding how a vulnerability can be identified, has been broken and the possibilities for future projects remains extremely exciting to me.

Creating business relations with companies to help assist me on my project has significantly enhanced my social skills as I had to be both punctual and professional in proposing my ideas. Besides this, since my topic is arguably on a controversial side of ethical borders, I feel like I am more adequately equipped to take on challenging research topics in a way which seeks to provide a positive outcome whilst remaining in legal boundaries.

My ability to write about complex topics in a way which could be understood by someone completely unexperienced in the field makes me feel my own understanding of the selected topics has improved significantly. Especially as I have taken a highly analytical approach which focused on identifying a theory and demonstrating it along with its risks.

At the beginning of my project, I was worried where it was going to end up but as I started to invest more of my time the outcome became clearer. If you were to ask me where I thought this paper would be today, I doubt my previous self could even remotely picture how it had turned out.

Finally, I really feel like my project has been successful regardless of the pandemic which hindered both my working conditions and motivation drastically. There is no doubt I have identified my own security vulnerability, demonstrated it and am obviously quite proud of where the work has finished.

Appendix A – Global Pandemic

COVID-19

This additional subsection aims to provide context for the adjustments made to my project following the global pandemic COVID-19. Before COVID, my project had considerably different aims and objectives which have now been revised to allow for my project to be completed in new working conditions. By the end of this section, the reader should become more informed about what my project was initially meant to be and understand the reasoning for my changes.

Coronavirus is a new virus [6] which attacks your lungs and airways. As of the 12th March, the World Health Organization (WHO) classified the illness as a global pandemic [7], giving it an official name, COVID-19. The pandemic has inadvertently caused significant changes to my working life due to the removal of in-class teaching at Newcastle University and, the introduction of a safety net policy for students.

At the start of my project, I had planned to explore a set of three vulnerabilities in detail and these were:

- (1) A text-based customer verification bypass,
- (2) A variation of the Distributed Guessing Attack and,
- (3) Luhn's Algorithm with the Distributed Guessing Attack.

Since the pandemic, I have been struggling to produce work at my normal productivity level due to new working conditions and, as expected, increased levels of stress. Moreover, I had started to fall behind my initial project plan proposed in my proposal which made process of exploring each vulnerability more difficult. Besides this, the introduction of a safety-net policy has given me the leigh weigh to produce a satisfactory dissertation given the extenuating circumstances.

Nevertheless, I will be continuing with my project but have only chosen to investigate a single vulnerability in depth which, is the text-based customer verification bypass. The reason for choosing this vulnerability is the exploit was discovered by myself through threat analysis techniques but also, I along with my supervisor believe it is important to disclose and publish.

Finally, I think it is important to give some context on my other two unexplored vulnerabilities as they may be researched in the future, whether it be done by me during my master's thesis or, by other graduate students.

Distributed Guessing Attack

The first unexplored vulnerability was a variation of the distributed guessing attack, initially discovered by M. Ali in his master's thesis during April 2018. Ali presented a brute forcing technique [8] which divided the guessing amongst many retailers, illustrating the fact card payment networks, such as VISA, do not have a centralised validation system in place to detect failed payment attempts across retailers.

This attack worked by exploiting the discrepancies in fields required by online retailers to process a payment online. Retailers are split into three groups, based on their required sensitive data, and these were:

Table 14- List of groups required for the Distributed Guessing Attack

Group #	Fields Required
1	Retailer requires only the primary account number and expiration date
2	Retailers require each field in Group 1 and the card's verification value
3	Retailers require each field in Group 2 and the cardholder's address

Attack Walkthrough

To complete this attack, the victim's primary account number (card number) must be known in advanced. Once known, software developed by Ali, sequentially brute forces a single sensitive card field at a time, starting with its expiry date (Group 1) and iterating up until the final group described in *table 1*.

Card payment networks brought about changes to combat this attack in 2018, making it mandatory for online retailers to require up-to Ali's second group [9] of online retailers. These changes have subsequently rendered the initial guessing attack useless as brute forcing both the expiry date and card verification value simultaneously is an intractable problem.

Previous Research

My research intended to explore a variation of this algorithm following the legislation changes because I believe the attack is still possible given a card's expiration date to

be less secure than its verification value. This statement is true because online retailers store payment information about customers inside of a database whereby, their cards verification value is encrypted using a one-way hashing algorithm. When an unknown one-way algorithm is used, it becomes mathematically impossible for an attacker to reverse engineer the hash.

You may better understand my point if you have previously stored payment details online. Generally, you are required to resubmit your card's verification value since it is not even known by the retailer itself. And so, in the event of a retailers database becoming compromised, it can be a fair assumption to assume an attacker could access both a customer's primary account number and expiration date because they are generally stored in plaintext or use basic hashing algorithms such as SHA1 [10] or MD5 which can be easily cracked with software such as Hashcat.

Besides this, it is known that online black markets exist all over the web which allow for the purchasing of sensitive card details. From my own research, it appears to be common practice for underground markets to present buyers with a card's primary account number and expiration date before purchase, with the actual purchasing step being used to obtain the card's verification value.

To summarise, my presented attack variation would work under the same principles presented by Ali, from using the card's bank identification number to identify numeric fields in the customer's address and distributing guessing attempts across multiple unique retailers. The key difference, however, is in order to conduct the attack, the card's primary account number and expiration date must be known in advanced. In Ali's terms, to perform the distributed attack the attacker must begin from the second group of retailers.

Valid Card Generation with Luhn's Algorithm

My second intended vulnerability was partially dependent on the success of my distributed guessing attack variation because, this attack considered an interesting mathematical theory which states:

'Can we generate credit card numbers which are both valid and linked to real customers?'

Luhn's algorithm also known as modulus 10 is a simple checksum formula developed by Hans Peter Luhn during his time at IBM [11] and it is a simple way to check if a

credit card or identification number is valid. This algorithm is generally described as the king pin in online payments because it has seen adoption in almost every online retailer. One reason for the algorithm's mass adoption is validating card numbers before attempting to process a payment reduces the number of redundant requests which, must be made to a card payment following a user entering in a card number. If the system can successfully detect when a card number is invalid before making the request, the appropriate measures can be put in place to not send the redundant request.

This algorithm can be extended to produce a set of card numbers which are valid for a given bank. The first six digits of a primary account number are known as a card's Bank Identification Number (BIN) and provide information about [11] the card's brand, issuing bank, type and more. When banks allocate customers with a new card, they begin with the card's BIN and use Luhn's Algorithm to generate the remaining 10 digits. *Table 15* provides a visual run through on how we can extract data from card numbers.

Table 15 – Primary Account Number digit analysis

Card Number: 4751-2800-4201-000X
Digits 1-6: Bank Identification Number (BIN)
Digits 7-15 – Assigned by the issuing bank to denote a personal account number (highlighted in red)
Digit 16 – Numeric checksum assigned by the Luhn's algorithm to check if a card number is corrected (highlighted in green)

Using the information from *table 15*, it becomes clearer on how selecting a popular issuing bank and card brand can make the subsequent generation of a card number linked to a real customer possible. Theoretically, a bank can only maintain a total of $(10^9 - 1) = 999,999,999$ valid card numbers which adhere to both a given bank identification number and Luhn's algorithm.

Undoubtedly banking organisations are aware of this and have measures in place to prevent a single BIN from reaching full capacity as it would for fraudsters to endlessly generate valid card numbers. Even if an attacker could generate a valid customer's

card number, they would need to be able to discover each of its sensitive data fields before a payment could be authorised. If they took a non-distributed guessing approach, the card would likely be blocked by the banks fraud protection system before any progress could be made.

Mathematical Theory Crafting

This subsection completes some basic probability theory to determine if the generation of a card number and its associated fields is truly plausible.

To begin, let us assume a single BIN has been filled up to half of its capacity which, is approximately 10^5 valid card numbers. We will attempt to generate 500 random card numbers with the hope of at least one being valid.

The probability of generating a card number which is linked up to a real customers account is $\frac{10^5-1}{10^9-1} \times 100 \approx 0.01\%$ which, evidently, is extremely unlikely. But if we were to repeat the generation 500 times, which can be done computationally in seconds, the probability of generation would be $(1 - (1 - 0.01)^{500}) \times 100 \approx 99\%$. A probability of 99% almost guarantees that at least one of the generated cards will be linked to a real customers account.

Understandably this theory may have been significantly easier to both test and investigation before the patches were pushed by card payments networks in response to the findings by Ali but, even still, locating a card's expiration date is not as difficult as finding its verification value. In practice, the expiration date of a card can estimate by understanding most credit and debit cards expire within 2-3 years [13] after their issuing date. The first issuing date of a card can also generally be found using information stored on BIN databases such as BinDB [14] which are organisations who maintain information about all bank BINs.

Moreover, on the mathematics, we can roughly estimate a cards expiration date to be within any month in a two year expiration range and so, the probability of correctly guessing a generated card's expiration date is $\frac{1}{24} \times 100 = 4.1\%$. Applying this to our previously iterated probability, the chance of us generating a card and guessing the expiration date would be $0.041 \times 0.99 \approx 4.1\%$. Now even though a 4% chance of generating a card is quite low; referring to Ali's work where he found it only takes 6 seconds [8] to guess the remaining card fields. This data is a useful insight into

understand how if such an operation were to be conducted at scale, it could be successful in generating a plethora of valid credit card details.

Notes

As a note, once the card's expiration date and number has been located the distributed guessing attack would be used to identify the card's verification value. For each of the generated cards, an estimated expiration date is selected, and the card's verification value is brute forced. We can almost guarantee the verification value to be identified as simple distributed brute force should locate the number. If the estimated expiration date was correct, once the verification value is found the payment will be authorised.

Besides this, I have been unable to further research into the discovering a new way to identify a card's expiration date. Since the new legislation was introduced, companies such as Amazon who allow for their users to add card numbers solely off of their card number and expiration date have stopped checking if the expiration date is valid until a payment is attempted to be authorised. When a payment needs to be authorised, the user is subsequently prompted for their card's verification value and no insight about what went wrong is given should the payment fail to process.

Retailers which have been listened in the appendix of Ali's paper [8] have also changed their approach in handling online payments following his responsible disclosure. Regardless, it is still likely for web payment systems to exist who first verify a card's expiration date before the card's verification value. If the latter statement is found to be true, my described idea will be exacerbated further, assuming the distributed guessing attack variation is successful.

Conclusions and The Future

From my analysis, I believe it to be possible for an attacker to generate valid card numbers which are linked to real customer accounts and have remaining sensitive details such as the card's expiration found using estimation and brute force techniques. The challenges faced by this vulnerability which should seek further investigation are identifying a card's expiration date, currently done by estimation, as it is likely for an alternative and more full-proof approach to exist.

Such a vulnerability sadly appears to be inherent with the design of our global payment system because card numbers are generated using public algorithms and the data surrounding such as BINs are also very open. This attack is generally not a

problem if manually conducted by a single user, but if operated at a large scale with cloud competition the ramifications could be catastrophic to banking users.

Some security experts may argue the discovery of a valid card and its sensitive details remain to be insufficient information to cause any financial damage to a user's account because two step customer verification systems like 3DS 2.0 will step in to prevent a transaction from being authorised. The problem with this argument is attackers can use SOCKS proxies to spoof the location of their machine closer to the victim which may prevent such systems from being triggered. Moreover, if the two-step system is triggered, some banking organisations may opt to send a verification text code which can be bypassed (as explained later in this paper). Regardless, I have no doubt in stating this theory requires further investigation to test the validity of my claims.

Appendix B – Experiment Results

HSBC - EE

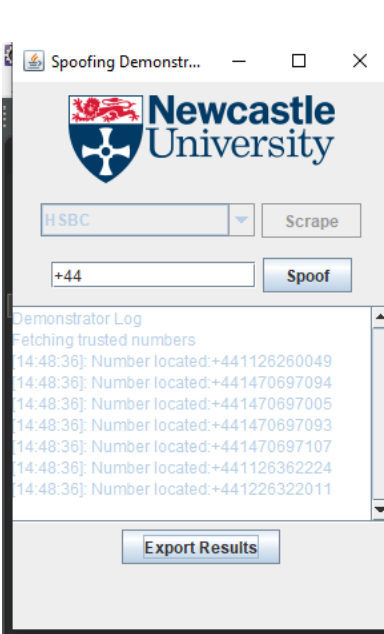


Figure 16 - Snapshot of the scraped numbers for HSBC

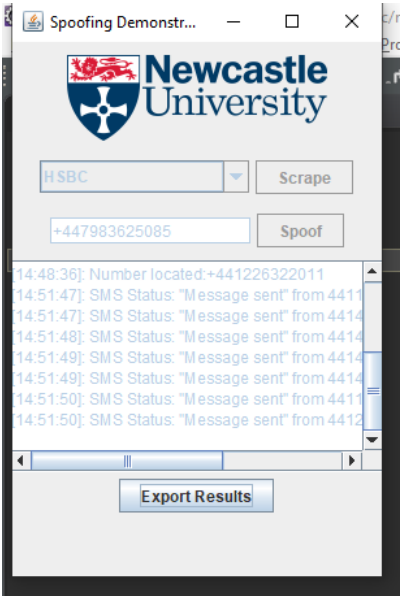


Figure 17 – Screenshot of the sent spoofed text

HSBC Export Results.txt - Notepad

File Edit Format View Help

Demonstrator Log

Fetching trusted numbers

[14:48:36]: Number located:+441126260049

[14:48:36]: Number located:+441470697094

[14:48:36]: Number located:+441470697005

[14:48:36]: Number located:+441470697093

[14:48:36]: Number located:+441470697107

[14:48:36]: Number located:+441126362224

[14:48:36]: Number located:+441226322011

[14:51:47]: SMS Status: "Message sent" from 441126260049 to "447983625085"

[14:51:47]: SMS Status: "Message sent" from 441470697094 to "447983625085"

[14:51:48]: SMS Status: "Message sent" from 441470697005 to "447983625085"

[14:51:49]: SMS Status: "Message sent" from 441470697093 to "447983625085"

[14:51:49]: SMS Status: "Message sent" from 441470697107 to "447983625085"

[14:51:50]: SMS Status: "Message sent" from 441126362224 to "447983625085"

[14:51:50]: SMS Status: "Message sent" from 441226322011 to "447983625085"

Figure 18 - Snapshot of the exported demonstrator log

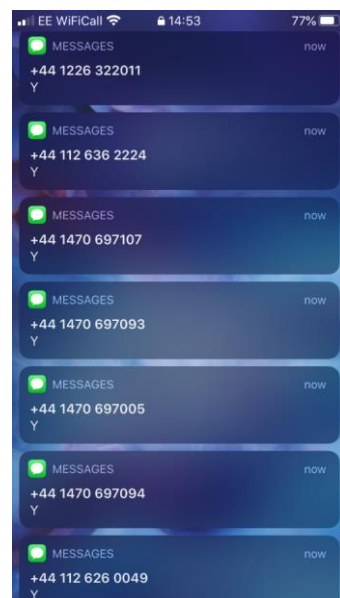


Figure 19 - Snapshot of the client phone for EE with HSBC Bank

Lloyds Bank – Three

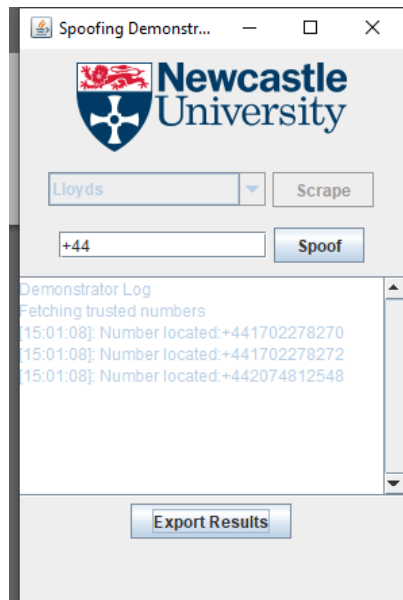


Figure 20 – Snapshot of Lloyd's scraped trusted numbers

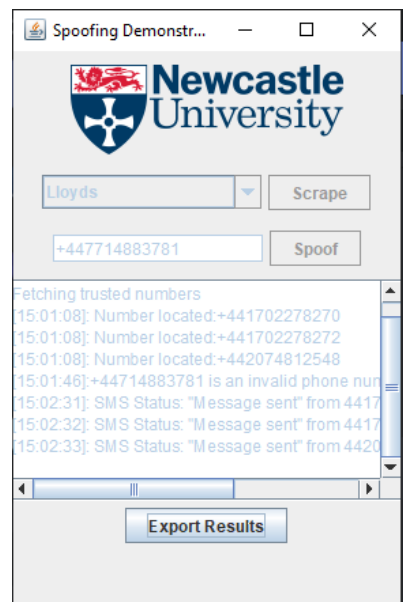


Figure 21 – Snapshot of the sent spoofed text for Three mobile



Figure 22 - Snapshot of the client's phone receiving the spoofed texts

```
Lloyds results.txt - Notepad
File Edit Format View Help
Demonstrator Log
Fetching trusted numbers
[15:01:08]: Number located:+441702278270
[15:01:08]: Number located:+441702278272
[15:01:08]: Number located:+442074812548
[15:01:46]:+44714883781 is an invalid phone number
[15:02:31]: SMS Status: "Message sent" from 441702278270 to "447714883781"
[15:02:32]: SMS Status: "Message sent" from 441702278272 to "447714883781"
[15:02:33]: SMS Status: "Message sent" from 442074812548 to "447714883781"
```

Figure 23 – Snapshot of the demonstrator log for Lloyds

Barclays O2

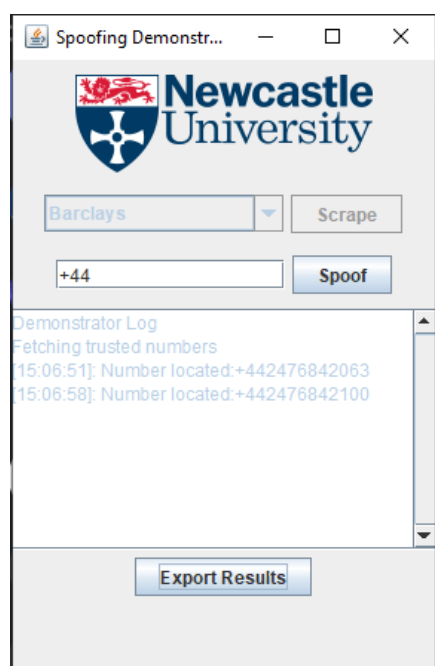


Figure 24 – Snapshot of Barclay's scraped trusted numbers

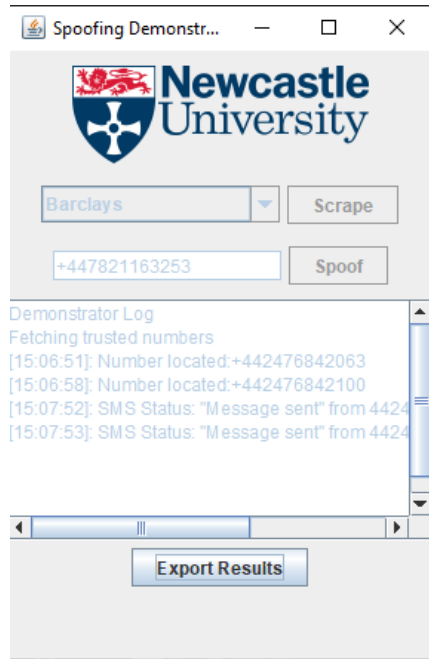


Figure 25 - Snapshot of the delivered spoofed texts for O2

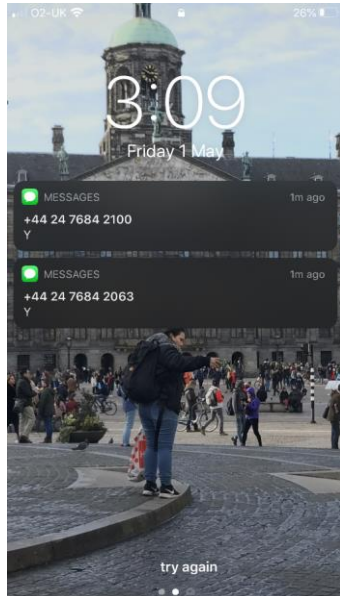


Figure 26 - Snapshot of the O2 Client's lock screen

Barclays Results O2.txt - Notepad

File Edit Format View Help

Demonstrator Log

Fetching trusted numbers

[15:06:51]: Number located:+442476842063

[15:06:58]: Number located:+442476842100

[15:07:52]: SMS Status: "Message sent" from 442476842063 to "447821163253"

[15:07:53]: SMS Status: "Message sent" from 442476842100 to "447821163253"

Figure 27 - Snapshot of log's result for O2

Barclays Vodafone

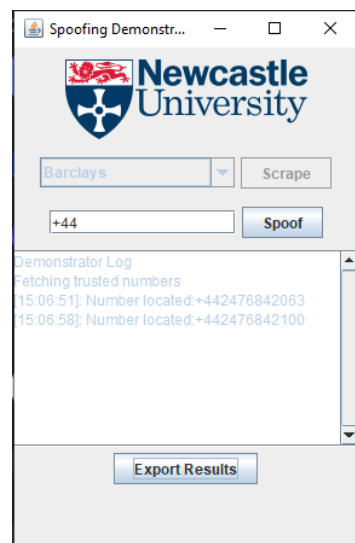


Figure 28 - Snapshot of the scraped trusted numbers for Barclays

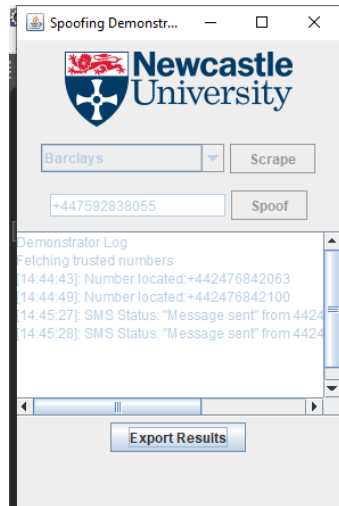


Figure 29 - Snapshot of the delivered spoofed texts for Vodafone

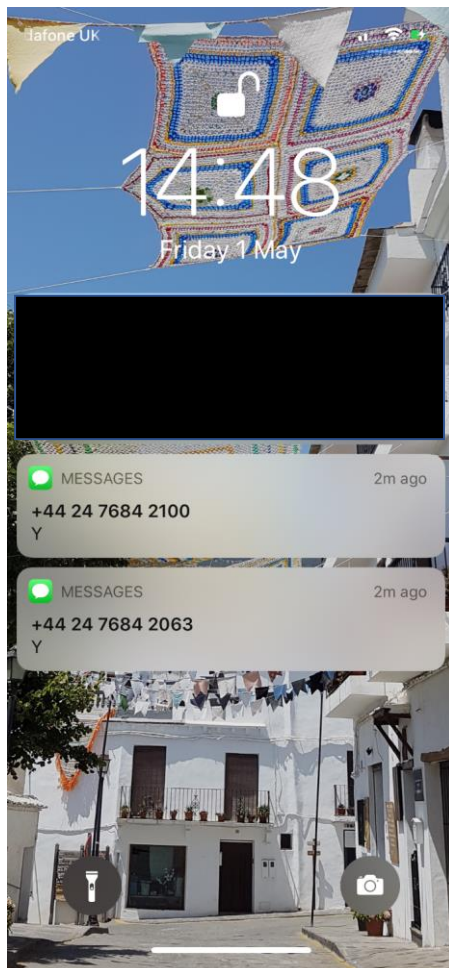


Figure 30 - Snapshot of the Vodafone lock screen

Barclays.txt - Notepad

File Edit Format View Help

Demonstrator Log

Fetching trusted numbers

[14:53:32]: Number located:+442476842063

[14:53:39]: Number located:+442476842100

[14:54:28]: SMS Status: "Message sent" from 442476842063 to "447592838055"

[14:54:29]: SMS Status: "Message sent" from 442476842100 to "447592838055"

Figure 31 – Snapshot of the log's result for Vodafone

REFERENCES

- [1] UK Finance. *The Definitive Overview of Payment Industry Fraud*. 20 Jan. 2019.
- [2] Hern, Alex, and Jim Waterson. "Ofcom to Be Put in Charge of Regulating Internet in UK." *The Guardian*, 12 Feb. 2020, www.theguardian.com/media/2020/feb/12/ofcom-to-be-put-in-charge-of-regulating-internet-in-uk.
- [3] Hadar, Mary. "Think Your Credit Card Is Safe in Your Wallet? Think Again." *Washington Post*, 11 Sept. 2019, www.washingtonpost.com/business/think-your-credit-card-is-safe-in-your-wallet-think-again/2019/09/11/05e316e4-be0e-11e9-b873-63ace636af08_story.html. Accessed 4 May 2020.
- [4] Ali, Fareeha. "A Decade in Review: Ecommerce Sales vs. Retail Sales for 2007-2018." *Digital Commerce 360*, 2018, www.digitalcommerce360.com/article/e-commerce-sales-retail-sales-ten-year-review/.
- [5] ---. "UK PAYMENT MARKETS SUMMARY 2019." *UK Finance*, 1 June 2019, www.ukfinance.org.uk/sites/default/files/uploads/pdf/UK-Finance-UK-Payment-Markets-Report-2019-SUMMARY.pdf.
- [6] NHS. "Coronavirus (COVID-19)." *Nhs.Uk*, 22 Apr. 2020, www.nhs.uk/conditions/coronavirus-covid-19/#overview. Accessed 4 May 2020.
- [7] WHO. "WHO Announces COVID-19 Outbreak a Pandemic." *Www.Euro.Who.Int*, 12 Mar. 2020, www.euro.who.int/en/health-topics/health-emergencies/coronavirus-covid-19/news/news/2020/3/who-announces-covid-19-outbreak-a-pandemic.
- [8] Ali, Mohammed Aamir, et al. "Does the Online Card Payment Landscape Unwittingly Facilitate Fraud?" *IEEE Security & Privacy*, vol. 15, no. 2, Mar. 2017, pp. 78–86, theses.ncl.ac.uk/jspui/bitstream/10443/4567/1/Ali%20MA%202019.pdf, 10.1109/msp.2017.27. Accessed 4 May 2020.

- [9] Georgescu, Radu. "Card Brands Now Require CVV Security Code." *Www.Helcim.Com*, 20 Apr. 2018, www.helcim.com/article/card-brands-require-cvv/. Accessed 4 May 2020.
- [10] Google Research. "SHattered." *Shattered.Io*, 17 Feb. 2017, shattered.io/. Accessed 4 May 2020.
- [11] Wikipedia Contributors (2019). *Luhn algorithm*. [online] Wikipedia. Available at: https://en.wikipedia.org/wiki/Luhn_algorithm.
- [12] Kagan, J. (n.d.). *The Value of Bank Identification Numbers*. [online] Investopedia. Available at: <https://www.investopedia.com/terms/b/bank-identification-number.asp> [Accessed 5 May 2020].
- [13] Kokemuller, N. (2019). *How Long Does It Take for a Credit Card to Expire?* [online] Pocketsense. Available at: <https://pocketsense.com/long-credit-card-expire-16923.html> [Accessed 5 May 2020].
- [14] BinDB (2020). *BIN Database - Search card issuing bank name and country*. [online] www.bindb.com. Available at: <https://www.bindb.com/bin-database.html> [Accessed 5 May 2020].
- [15] Murdoch, S. and Anderson, R. (2010). *Verified by Visa and MasterCard SecureCode: or, How Not to Design Authentication*. [online] Available at: <https://www.cl.cam.ac.uk/~rja14/Papers/fc10vbvsecurecode.pdf> [Accessed 5 May 2020].
- [16] Wikipedia Contributors (2019a). *EMV*. [online] Wikipedia. Available at: <https://en.wikipedia.org/wiki/EMV>.
- [17] Security Standards Council (2010). *Payment Card Industry (PCI) Data Security Standard PCI DSS Applicability in an EMV Environment A Guidance Document*. [online] Available at: https://www.pcisecuritystandards.org/documents/pci_dss_emv.pdf.
- [18] Murdoch, S.J., Drimer, S., Anderson, R. and Bond, M. (n.d.). Chip and PIN is Broken. *Chip and PIN is Broken*. [online] Available at: <https://www.cl.cam.ac.uk/research/security/banking/nopin/oakland10chipbroken.pdf> [Accessed 5 May 2020].
- [19] Ali, Mohammed & van Moorsel, Aad. (2019). Designed to Be Broken: A Reverse Engineering Study of the 3D Secure 2.0 Payment Protocol. 10.1007/978-3-030-32101-7_13.

- [20] Financial Ombudsman Services (2020b). *Financial Ombudsman Service*. [online] Financial Ombudsman. Available at: <https://www.financial-ombudsman.org.uk/>.
- [21] RedTeam Pentesting GmbH (2009). *Man-in-the-Middle Attacks against the chipTAN comfort Online Banking System*. [online] Available at: https://www.redteam-pentesting.de/publications/2009-11-23-MitM-chipTAN-comfort_RedTeam-Pentesting_EN.pdf [Accessed 5 May 2020].
- [22] Drimer, S., Murdoch, S. and Anderson, R. (2009b). *Optimised to Fail: Card Readers for Online Banking*. [online] Available at: <https://murdoch.is/papers/fc09optimised.pdf>.
- [23] Anderson, J. (2017). *Security Protocols XXIV: 24th International Workshop, Brno, Czech Republic, April 7-8, 2016, Revised Selected Papers*. Springer.
- [24] Insight (2020c). *Biometric Authentication / Insight UK*. [online] www.uk.insight.com. Available at: <https://www.uk.insight.com/en-gb/shop/rsa/biometric-authentication>.
- [25] Förde Sparkasse (2020c). *Der Blog der Förde Sparkasse - Das Online-Magazin Ihrer Sparkasse*. [online] Blog der Förde Sparkasse. Available at: <https://blog.foerde-sparkasse.de/> [Accessed 5 May 2020].
- [26] HSBC. “Received a Text? | Security Centre - HSBC UK.” *Www.Hsbc.Co.Uk*, 5 May 2020, www.hsbc.co.uk/help/security-centre/received-a-text/. Accessed 5 May 2020.
- [27] Wikipedia Contributors. “Caller ID Spoofing.” *Wikipedia*, 30 Apr. 2020, en.wikipedia.org/wiki/Caller_ID_spoofing. Accessed 5 May 2020.
- [28] Wikipedia Contributors (2020e). *SMS spoofing*. [online] Wikipedia. Available at: https://en.wikipedia.org/wiki/SMS_spoofing [Accessed 5 May 2020].
- [29] cSpoof (2020b). *cSpoof - SMS Gateway and Marketing*. [online] cspooof.com. Available at: <https://cspooof.com/> [Accessed 5 May 2020].
- [30] Morreale, M. (2017). *Daily SMS Mobile Usage Statistics / SMSEagle*. [online] [Smseagle.eu](http://smseagle.eu). Available at: <https://www.smseagle.eu/2017/03/06/daily-sms-mobile-statistics/>.
- [31] WIRED Staff (2016). *So Hey You Should Stop Using Texts for Two-Factor Authentication*. [online] WIRED. Available at: <https://www.wired.com/2016/06/hey-stop-using-texts-two-factor-authentication/>.
- [32] ExactBins (2020c). *BIN Database free / BIN lookup free / Free credit card bin lookup – Free Online Demo / ExactBin*. [online] www.exactbins.com. Available at: <https://www.exactbins.com/bin-lookup> [Accessed 5 May 2020].

- [33] 192.com (2020a). *Search for People, Businesses and Places - 192.com*. [online] 192.com. Available at: <https://www.192.com/> [Accessed 5 May 2020].
- [34] Dehashed (2020c). *DeHashed — #FreeThePassword*. [online] www.dehashed.com. Available at: <https://www.dehashed.com> [Accessed 5 May 2020].
- [35] vip72 (2020i). *V.I.P. Services - Security, Anonymous proxy, VPN Service, Proxy Service, Proxy Server, Hide your IP*. [online] Vip72.com. Available at: <http://vip72.com/> [Accessed 5 May 2020].
- [36] Wikipedia Contributors (2019c). *STRIDE (security)*. [online] Wikipedia. Available at: [https://en.wikipedia.org/wiki/STRIDE_\(security\)](https://en.wikipedia.org/wiki/STRIDE_(security)).
- [37] Microsoft (2018). *Microsoft Security Development Lifecycle Threat Modelling*. [online] Microsoft.com. Available at: <https://www.microsoft.com/en-us/securityengineering/sdl/threatmodeling>.
- [38] Ramadlan, M. (2013). *Introduction and implementation OWASP Risk Rating Management*. [online] Available at: <https://owasp.org/www-pdf-archive/Riskratingmanagement-170615172835.pdf> [Accessed 5 May 2020].
- [39] ClickSend (2020c). *Bulk SMS Gateway & Email Marketing Platform – Email & SMS Marketing*. [online] ClickSend. Available at: <https://www.clicksend.com/gb/> [Accessed 5 May 2020].
- [40] SendInBlue (2020j). *Product Features*. [online] Sendinblue. Available at: <https://www.sendinblue.com/features/> [Accessed 5 May 2020].
- [41] RTL-SDR (2013). *Receiving, Decoding and Decrypting GSM Signals with the RTL-SDR*. [online] rtl-sdr.com. Available at: <https://www.rtl-sdr.com/receiving-decoding-decrypting-gsm-signals-rtl-sdr/> [Accessed 5 May 2020].
- [42] Wikipedia Contributors (2019b). *KASUMI*. [online] Wikipedia. Available at: <https://en.wikipedia.org/wiki/KASUMI> [Accessed 17 Oct. 2019].
- [43] ISPreview (10AD). *New Report Examines the UK Impact of Switching Off 2G Mobile - ISPreview UK*. [online] www.ispreview.co.uk. Available at: <https://www.ispreview.co.uk/index.php/2019/10/new-report-examines-the-uk-impact-of-switching-off-2g-mobile.html>.
- [44] Cloudflare (2019b). *Cloudflare*. [online] Cloudflare. Available at: <https://www.cloudflare.com/learning/ddos/glossary/denial-of-service/>.

- [45] Hoff, J. (2016). *Rooting for Newbies: How to gain root access*. [online] Android Community. Available at: <https://androidcommunity.com/rooting-for-newbies-how-to-gain-root-access-20161120/> [Accessed 5 May 2020].
- [46] Goodin, D. (2011). *How to slay a cellphone with a single text*. [online] www.theregister.co.uk. Available at: https://www.theregister.co.uk/2011/03/21/sms_of_death_explained/ [Accessed 5 May 2020].
- [47] Wikipedia Contributors (2020l). *List of Sony Ericsson products*. [online] Wikipedia. Available at: https://en.wikipedia.org/wiki/List_of_Sony_Ericsson_products [Accessed 5 May 2020].
- [48] Wikipedia Contributors (2019e). *Zero-day (computing)*. [online] Wikipedia. Available at: [https://en.wikipedia.org/wiki/Zero-day_\(computing\)](https://en.wikipedia.org/wiki/Zero-day_(computing)).
- [49] phoenixNap (2019d). *7 Proven Tactics To Prevent DDoS Attacks: Make a Security Plan Today!* [online] PhoenixNAP Global IT Services. Available at: <https://phoenixnap.com/blog/prevent-ddos-attacks>.
- [50] MIT (2019c). *How a quantum computer could break 2048-bit RSA encryption in 8 hours*. [online] MIT Technology Review. Available at: <https://www.technologyreview.com/2019/05/30/65724/how-a-quantum-computer-could-break-2048-bit-rsa-encryption-in-8-hours/>.
- [51] geeksForGeeks (2017). *Unified Modeling Language (UML) / An Introduction - GeeksforGeeks*. [online] GeeksforGeeks. Available at: <https://www.geeksforgeeks.org/unified-modeling-language-uml-introduction/>.
- [52] draw.io (2020f). *Flowchart Maker & Online Diagram Software*. [online] [app.diagrams.net](http://draw.io). Available at: <http://draw.io/> [Accessed 6 May 2020].
- [53] phonenumbersIE (2020k). *+447480357009, 07480357009 – Number of ratings: 37×....*. [online] www.phonenumbers.ie. Available at: <https://www.phonenumbers.ie/number/447480357009> [Accessed 6 May 2020].
- [54] Nielsen, J. (2006). *Heuristics for User Interface Design Ten Usability Heuristics*. [online] Available at: <http://lore.ua.ac.be/Teaching/SE3BAC/practicum/acceptanceAndUsabilityTesting/TenUsabilityHeuristics.pdf> [Accessed 1 May 2020].
- [55] CFI (2018a). *Top Banks in the UK - Overview and Guide to Top 10 UK Banks*. [online] Corporate Finance Institute. Available at: <https://corporatefinanceinstitute.com/resources/careers/companies/top-banks-in-the-uk/>.

[56] Selenium Project (2020l). *The Selenium Browser Automation Project :: Documentation for Selenium*. [online] www.selenium.dev. Available at: <https://www.selenium.dev/documentation/en/>.

[57] Wikipedia Contributors (2019f). *Swing (Java)*. [online] Wikipedia. Available at: [https://en.wikipedia.org/wiki/Swing_\(Java\)](https://en.wikipedia.org/wiki/Swing_(Java)) [Accessed 6 May 2020].

