National College of Ireland

Android Browser Exploit Prevention Using NFC Tag Filtering

MSc Internship
Cyber Security

Tony Thomas
Student ID: x18147330

School of Computing
National College of Ireland

Supervisor: Ben Fletcher

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Tony Thomas
Student ID:	x18147330
Programme:	Cyber Security **Year:** 2019
Module:	InternShip
Supervisor:	Ben Fletcher
Submission Due Date:	29/01/2020
Project Title:	Android Browser Exploit Prevention Using NFC Tag Filtering
Word Count:	6431 **Page 20**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.
I agree to an electronic copy of my thesis being made publicly available on NORMA the National College of Ireland's Institutional Repository for consultation.

Signature: .....

Date: .....

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Android Browser Exploit Prevention Using NFC Tag Filtering

Tony Thomas
X18147330

Abstract

Near Field Communication (NFC) is one of the prominent and popular communication technologies used for transmitting information between two devices within a limited range with the help of radiofrequency[1]. The integration of this technology into smartphones has led manufacturers, developers and the end-users to a growing interest in this technology and its use cases. This technology is widely adopted in various application domains such as contactless payment systems, ticketing, access control, identity management, device tracking etc. Even though the range of NFC is limited to a few centimetres, the NFC Forum has not made sufficient effort in securing all aspects of this technology. The NFC tag is one such area. Unlike other types of NFC embedded devices such as contactless smart card, debit/credit cards that use authentication, encryption and other forms of security measures, these tags can easily be read, written, spoofed, intercepted or altered by anyone with a valid NFC reader such as a smartphone. Overwriting these tags with malicious content could jeopardize the security of the system since these devices are designed to execute actions automatically without any user intervention. In this research paper, we evaluate this security issue and propose a secure middleware interface between the NFC reader and the tag so that only valid tag content gets executed by the reader. Our middleware application explicitly focuses on identifying malicious URL's within the tag since they have the peak chance of compromising the security of the device. This will ultimately add an extra layer of security while reading information from the tag.

Keywords : NFC, Mobile Security, Browser, Exploit, Android, Blacklisting, URL, Tag, Spoofing.

1 Introduction

Near Field Communication (NFC) is a wireless technology that is used to establish unidirectional/bidirectional communication between two electronic devices with the help of radiofrequency waves over a short-range[1]. It operates in the frequency of 13.56 MHz with the range up to 4 cm. NFC can be considered as a subcategory of the Radio Frequency Identification[1] (RFID) technology. The data can be exchanged very easily just by keeping the two devices close to each other without the need for any middleware software provided that both the devices support NFC. Due to this reason, NFC is adopted by various other technologies such as ticketing systems, payment systems, access control, identity management, data transfer, pairing etc. NFC operates under three different modes namely reader/writer mode, card emulation mode and peer to peer mode [1]. In reader/writer mode, the active device(reader) will establish the connection with a passive device such as an NFC tag[2]. The card emulation mode allows readers such as a smartphone to act like a smart card to perform the transaction without the use of the actual card. In the case of peer to peer mode, two devices such as smartphones can directly set up a bi-directional communication channel between each other to exchange files and other information.

[1] RFID: https://www.abr.com/what-is-rfid-how-does-rfid-work/

According to research[2] conducted by "ABI Research" a New-York based Technology Research company, there is a huge prospect for NFC tags over the next decade. Their analysts predicted that the NFC market will continue to expand by 17.9% over the next decade, hitting almost $50 billion by 2025. This would mean that more people would start using NFC on their smartphones and the higher volume manufacturing capabilities would make the NFC tags cheaper and more affordable than ever before. This would definitely be a positive sign from a business and consumer perspective since the consumers can have on-demand access to contents, offers and direct one-to-one communication with the business on the go. They can also check and assure the authenticity of a product by analysing the respective NFC tags. However, from a cybersecurity perspective, the result of this research isn't as appealing as it seems to be. This is because the security of the NFC tag is not entirely safe and sound as we thought it was.

While going through one of the research papers about the security of Near Field Communication, we came to know that even a simple NFC tag can possess great threat to the security of a smartphone. Further, continuing in this direction we identified that there lies huge opportunities for conducting a study about the security of these tags. Even though most tags have the feature to lock itself with a password, encrypt its contents or to make them read-only, attackers still find ways to exploit the tag by attacks such as Denial of Service, eavesdropping, interception, spoofing and modification[3]. This unrestricted capability to modify tag contents could lead to various kinds of security problems. With tag modification, an attacker can inject malicious content into the tag either through an executable command or in the form of malicious URLs. The NFC tag authentication can be considered as a pragmatic solution against this type of attack but they are not implemented in every field of this technology and are confined to financial and ticketing services [4]. In this regard, J.Wu [5] in his research also proposed a method for checking the authenticity of the tags and prevent it from being spoofed. It involved generating a unique ID with the ID of the tag on each scan and verify it against a third-party server. This approach was also promising but it required the users to be registered and signed in with the server during the interaction. But the problem was that this registration and login requirement could create functionality problem in terms of the usability of the NFC tap and go feature. This prompted us to conduct the study on this particular topic and to suggest a feasible solution. Through this research we attempt to find a solution to our **research question : " Can NFC tag filtering provide an efficient way in detecting malicious URL from an NFC Tag? "** In order to find an answer to the question, we are introducing a secure intermediate framework for the tag reader (Smartphone) in the form of a middleware application. The main task of this application is to act as an intermediary between the tag reader and the NFC tag in order to detect malicious content. We are solely focusing on identifying malicious URL from the tag because they are most likely to compromise the security of the reader. Although there are numerous antivirus softwares available that provides secure web browsing capabilities, a big number of smartphone users use their phones without the installing such application. This requires a lightweight middleware application that can be loaded into the reader for detecting malicious URL from the tags.

The rest of the paper is divided as follows: A short overview on NFC technology and its application is covered in section 1. Section 2 discusses and contrasts some of the previous work done by researchers on the security of NFC. In section 3 we will discuss about the methodology that we are going to follow in order to find an appropriate answer to our research question.

[2] https://blog.nxp.com/industrial/abi-research-on-nfc-tags

Section 4 and 5 explains about the design specification of our proposed middleware application and its implementation. This is followed by the evaluation in section 6 where we discuss the performance evaluation of our application. And finally, we conclude at section 7 by discussing about the future scope of this research work.

2 Related Work

The recent development in Near field Communication and its expansion into mobile devices have given rise to various security challenges and vulnerabilities that the research community have to discuss and resolve. This section discusses about how communication takes place using the NFC technology, its mode of communication and some of the main security challenges and threats raised by security researchers in the field of Near Field Communication. This section is divided into 3 subcategories. In first sub-section we will look into the communication method between two NFC enabled devices. In the second sub-section we analyse the reader/writer mode by taking a closer look into its protocol stack and in the third section we will look into the security of the reader and the tag by reviewing the related work done by researchers.

2.1 Overview of NFC Technology

The NFC technology is used for establishing communication between two devices when they are placed within the range of 4 cm. It provides instant connectivity with a maximum speed of 424 kb per second and operates under the standard of ISO/EC 18000-3 [2]. Initiators are the devices that start the communication and respondents are the one at the receiving end of the communication. NFC mobile and readers have the capability to generate their own power, so they are active devices, while an NFC tag can't generate power on its own and requires the other party's radio frequency to power itself and is therefore considered as a passive device. Although there are three modes of NFC interaction, our work will focus specifically on the read/write mode as the methods as NFC tag falls under this mode. An active and passive device uses two modulation methods to communicate with each other. If the communication happens from an active to a passive device the Amplitude Shift Keying (ASK)[3] modulation will be used for the communication. Whereas if the communication happens from a passive device to an active device, the Load Modulation technique will be used [2].

In order to improve the performance and quality of the current modulation method, numerous theories were proposed, some of them focus on modifying the existing coding schemes whereas some propose completely new modulation techniques. For example, the writers of [6] suggested a strategy for achieving faster data rates for NFC devices that uses the frequency of 13.56 MHz They conducted a comparative study between Amplitude-Shift Keying (ASK) and Phase-Shift Keying(PSK)[4] under real time environment and discovered that PSK overperformed ASK by 23% in terms of power requirements and energy efficiency. In other research[7] the authors introduced a multi-level Phase Shift Keying(PSK) modulation for faster data transmission rate of systems that operate in13.56 MHz frequency. Azad et al.[8] suggest a Direct Antenna Modulation (DAM) technique to improve NFC communication efficiency as

[3] https://www.tutorialspoint.com/digital_communication/digital_communication_amplitude_shift_keying.htm
[4] https://www.tutorialspoint.com/digital_communication/digital_communication_phase_shift_keying.htm

most NFC devices run at a low radio frequency. Most of the articles we have discussed so far focus mainly on enhancing the performance and efficiency of the NFC technology. Only a little comprehensive research has been undertaken on the technology's security aspect. This clearly shows the trade-off between security and performance. In the following section we will identify how the NFC reader/writer mode works and will address the current weaknesses in its layers. This will help to realize the need for a safe middleware to detect malicious payloads between the tag and the reader.

2.2 Reader/Writer Operating Mode

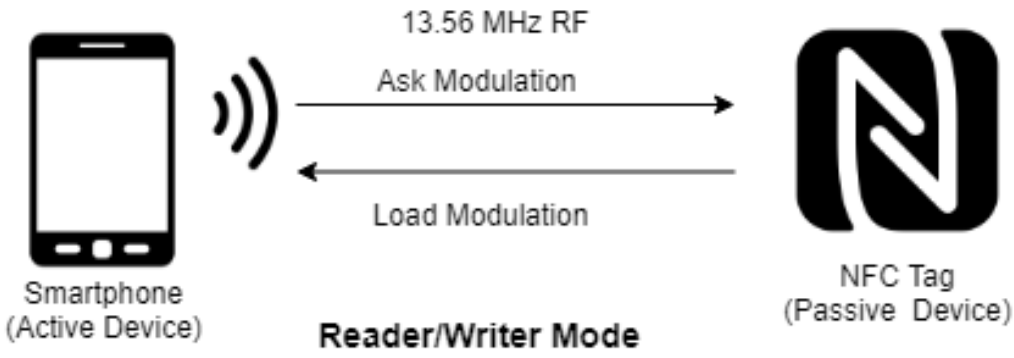


Figure 1: Reader/Writer Mode in NFC

In this mode, the smartphone initiates interaction as an active device and can perform both reading and writing operation on the NFC tag. Such tags are passive because they are not capable of generating radio frequency on their own. The reader/writer mode protocol architecture is classified into application, datalink and physical layer[2].

- Application layer

  All the software applications used within the reader for reading or altering the tag contents are included in this layer. They can use either the NDEF data format or their own format for managing the contents of the tag. Our application will be implemented in this layer and will communicate with the tag by making use of the NFC controller. L Renaud [9] developed an android based application as a proof of concept for hacking credit card based on the NFC technology.

- Datalink layer

  This layer determines the type of specification used for the communication between the tag and the reader. They are mainly of 4 types and these specifications decide what kind of instructions and commands a reader can carry out. NFC Data Exchange Format (NDEF)[5] is the standard format used for exchanging data between a tag and a reader. NFC Forum[6] is the organisation that defines and standardise them. An NDEF record

[5] https://www.dummies.com/consumer-electronics/nfc-data-exchange-format-ndef/
[6] https://nfc-forum.org/about-us/

includes a set of payload field and a header field that comprises of 5 flags[10]. Each NDEF record varies in terms of their length and the payload type which can be identified with the help of the unique record value. Roland Et al [10] stressed that the NFC Data Exchange Format could be subjected to different types of attacks without a suitable security mechanism. Mulliner[11] extended his research work through the development of new technique, tools and to leverage these vulnerabilities. Roland[10] revealed a fresh realistic attack named "Record Composition Attack" against the signed NDEF message.

- Physical layer

  This is the lowest layer where the data transmission happens via Radio Frequency. The communication on the physical layer is based on the ISO/IEC 14443 standard. Many of the researchers have already discovered several vulnerabilities and flaws in this standard. One of them was Wolfgang[12] who introduced a practical execution of relay attack on the system that uses the same protocol. He made use of a smartphone and a modified RFID tag emulator to redirect the Radio Frequency communication. What distinguishes his work from the rest is the fact that his solution complied with the ISO standard, while most other recommendations did not take this into account.

2.3 NFC Security

2.3.1 NFC Tag Security

The NFC tags operate under the reader/writer mode. These tags can be read or overwritten by an NFC reader like a smartphone. The common way to manipulate these tags is to write malicious data into the tag and thus compromise the device as soon as it comes in contact with the tag. The attacker can also clone the tag and thus duplicating the contents of the original tag or even removing the contents from the original tag. The use of applying the digital signature on the NFC Data Exchange Format (NDEF)[10] message seemed primarily to be a good alternative, but vulnerabilities were found later[4]. At Black Hat Security 2012, Charlie Miller[13] exhibited an attack where he was able to exploit a smartphone that uses NFC just by bringing it into contact with an NFC tag. The demonstration shows clearly that a mobile device can be hacked by injecting malicious code into the tag or by placing the target in contact with the modified tag. However, Miller did not suggest any preventive measure against the attack. Proper Authentication can be seen as a protective approach against this attack where special read and write permissions can be allocated to tags so that it can be accessed by only legitimate users. When it comes to authentication, tags can achieve it in two different ways. When there is no prior key exchange takes place between a tag and a reader to create a secure connection, we can use the offline authentication. The main disadvantage with this authentication is that it demands more computational power and the tags are only intended to handle low computation power[14]. In the case of online authentication, the process of authentication is done with the help of an online server. The main step involved in this authentication is to extract the unique id of the tag and cross verifying it with the ID in the server to check the authenticity. Although this authentication appears to be making the tag safer, the users have to be logged during the communication with the server. This will make an impact on the functionality and usability of the tag. In a recent study[15], an authentication scheme of two layers namely Public Key Cryptography(PKC)[7] and Public Key Infrastructure

[7] https://www.ibm.com/support/knowledgecenter/en/SSB23S_1.1.0.13/gtps7/s7pkey.html

(PKI)[8] are embedded into an authentication method between a reader and tag to provide a multi-layer authentication. The users were able to determine the authenticity of the product with the help of this method. The above studies show the role encryption plays in solving both the tag authentication problems.

2.3.2 NFC Reader Security

NFC readers are devices capable of reading or altering an NFC tag's content. These readers can usually be connected to a server either wired or wirelessly, for processing and storing information from the tags. All the readers that act as a card emulator processes sensitive information like payment credentials, passwords etc and it is necessary to secure the communication between the reader and the server through encryption[2]. One of the prominent researches regarding the security of the Reader was done by Mulliner. His work[11] focussed mainly on attacks and vulnerability analysis on NFC-enabled smartphones. He found numerous previously unknown vulnerabilities based on his work. He utilized these vulnerabilities by developing new kind of attack such as spoofing, malicious worms and denial of service that made use of the NDEF data format. One important thing to note here is that Mulliner carried out these attacks ,10 years ago, but there is still no evidence of a solid countermeasure against some of these attacks. Besides the attacks mentioned above there is a comprehensive debate going on about the potential attack surface in NFC including the Google Wallet attacks[16][17] and some mitigation techniques[18][19].

The following table below(Table 1) summarizes all the NFC attacks and vulnerabilities we have so far reviewed in this literature review. Looking at the table in greater detail, it is obvious that many of the discussed attacks and vulnerabilities still exist and do not have appropriate prevention.

Table 1:Discussed NFC based attacks and their countermeasures

Means of Communication	Attacks/Vulnerabilities	Countermeasures
NFC Tag/Reader	• Tag Modification • Tag spoofing • NFC Worms • Data corruption	• Tag Authentication • Digital Signature
Radio Frequency	• Eavesdropping • Man in the middle • Relay Attack • Denial of Service	• Secure communication channel • Use active/passive communication mode • Monitor the RF field

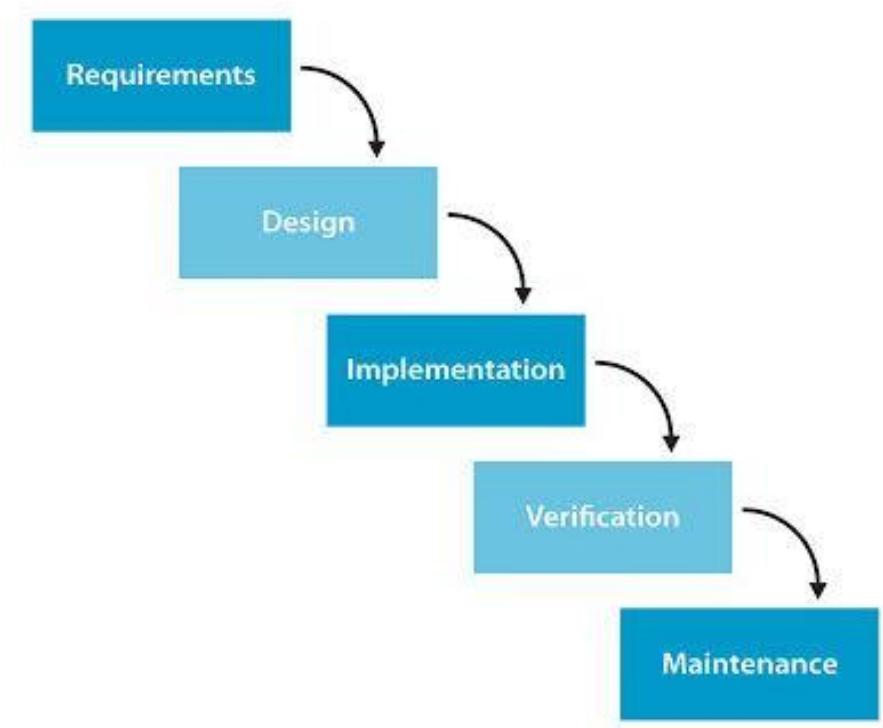
So far, we have discussed about various attacks, vulnerabilities and related works that was carried out by researchers in the field of NFC explicitly relating to the security of the reader, tag and the communication modes. We have also discussed various ways NFC phones can be hacked and how it can be mitigated.

[8] https://www.thalesesecurity.com/faq/public-key-infrastructure-pki/what-public-key-infrastructure-pki

3 Research Methodology

In the above section, we discussed various studies that have been carried out in the past regarding the security of NFC technology. This chapter gives a brief understanding about the methodological approach that we are taking in order to develop our application after taking the previously reviewed papers into consideration. Waterfall model is one of the software development approaches for developing an application in a linear-sequential manner[9]. According to Sommerville[20], this approach is known to be sequential and quite traditional model in the software industry and is used best when the application design is less complicated and when the requirements are absolutely clear. We will be adopting this methodological approach for developing our middleware application. This methodology will allow us to divide the project activities into distinct stages so that we can develop the application in a step by step manner. This approach is considered to be very sequential and conventional in the software development industry and is best used when the developer has a clear idea of what to do at every development stage. Since we already have a pre-planned idea on how to approach the problem it would be convenient for us to use the Waterfall model for our application development. The figure below depicts the various stages involved in the waterfall model.

Figure 2:Waterfall Model architecture



- Requirements

  Requirement analysis is the first stage in the waterfall model. During this phase, we have gathered all the necessary requirements needed for developing our proposed application. The main aim of our middleware application is to provide security to the NFC enabled smartphone from URL based attacks while interacting with the tag.

[9] https://activecollab.com/blog/project-management/waterfall-project-management-methodology

According to a global survey by statcounter[10] more than 75% of the global population uses Android as their operating system. Since it is the most popular operating system, we decided to develop the application as an Android Application Package (APK). One of the main requirements of the proposed application is that it should focus specifically on the URL based content from the tag. The application also must perform blacklisting and display the status of the URL in terms of malicious or non-malicious. This should be displayed with the help of color codes. (Red=malicious, green=non-malicious).

- Design

After gathering all the necessary requirements of our application, the next task is to design the application. In order to make the application easy to use, we have made a simple design that contains an activity representing a clean User Interface (UI) with a single screen. Once the device comes in contact with a tag containing a URL, the application will read the URL and displays it on the interface along with the appropriate colour code. Section 4 contains more information on the design specification.

- Implementation

We are using the android platform in order to develop the middleware application. As an open-source project, android offers a great deal of support and support and compatibility for application development and its deployment. We are developing the application with the help of Android Studio which is the official Integrated Development Environment (IDE) for Android operating system. Although other IDE's are available for the development, such as IntelliJ IDEA, Eclipse and many others, the User Interface of Android Studio, its stability and enhanced auto completion makes this a suitable platform for creating the software. We will be using JAVA for the application development. More information regarding implementation will be discussed under section 5.

- Verification

Verification and testing are a crucial part of software development. This will help in verifying whether the system meets the requirements. We will be completing performance, functionality and security testing on the application once it is deployed and installed into the reader. We will discuss more testing in section 6.

- Maintenance

Maintenance is the final stage of the waterfall model. This phase occurs after the application is fully operational and up and running. It can be in the form of fixing the software bugs, update package etc. Some of the key strengths and limitations of this model is described in the table below (table 2).

[10] https://gs.statcounter.com/os-market-share/mobile/worldwide

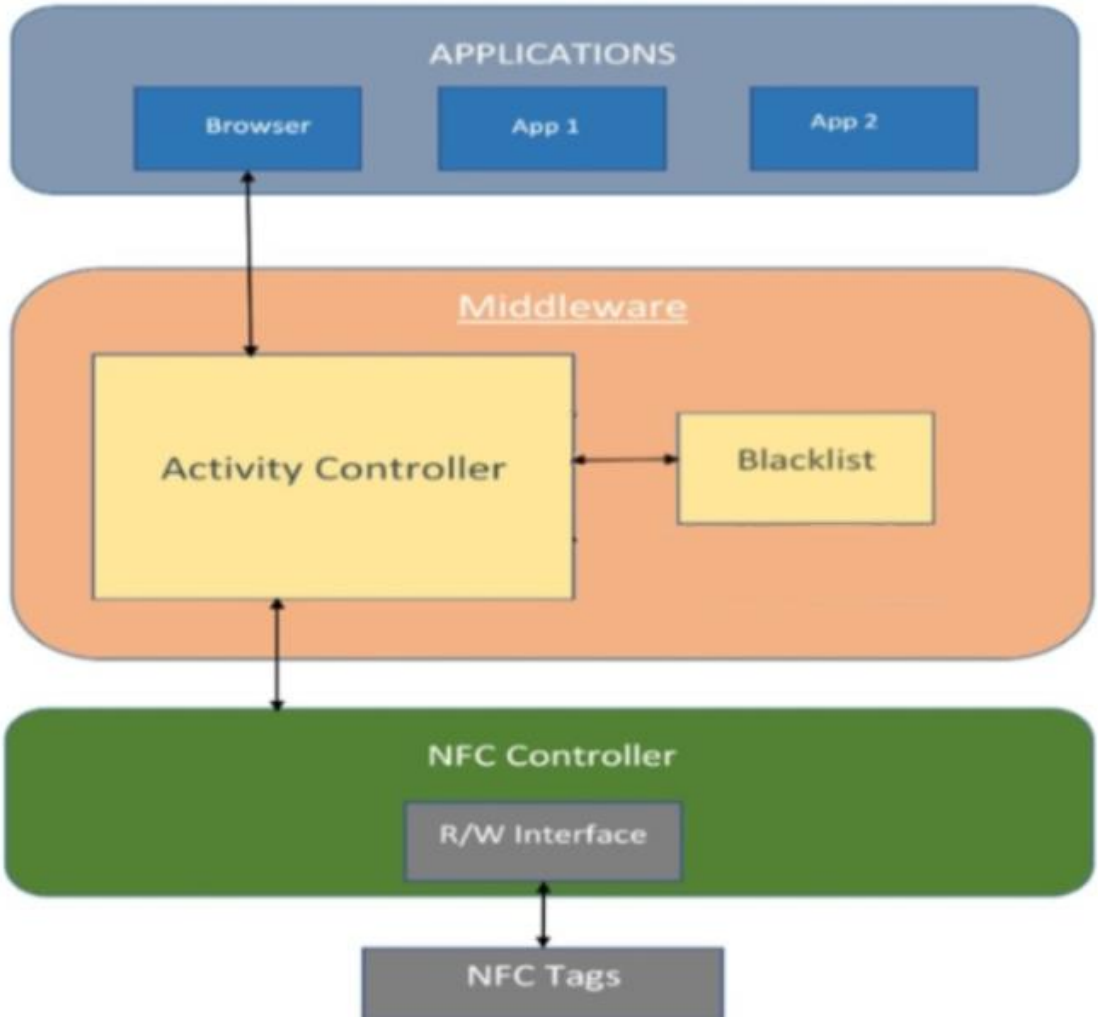
Table 2: Advantages and Disadvantages of Waterfall Model

Advantages	Disadvantages
A methodological and sequential approach	Every step is final. Developer can't go back to previous step and make changes.
Has a methodological documentation and record keeping	Depends completely on the initial requirements. If the initial requirements are faulty , it can severely impact the resulting application.
The developer knows exactly what to perform at every stages of the development.	The testing can only be done after full development of the application.
Suitable for smaller projects with very well-understood requirements	Not suitable for application where the needs keep on changing.

4 Design Specification

From the literature review, we have discussed how Mulliner[11] was able to take control of an NFC enabled phone. His approach not only focussed exploiting the NFC subsystem but also controlling various software applications within the devices. Taking this into matter our proposed application will be in the form of an android application because this is the easiest way of managing both the NFC subsystem and interaction with other applications. After installing our application, every time when the device comes in contact with an NFC tag, it will get initialised. The main focus of the application is to identify malicious URLs contained within the tag to ensure a safer communication with the tag. We will be using the blacklisting method to achieve this. Figure 3 illustrates the remodelled design of the NFC architecture after incorporating our middleware component. Our application will act as an intermediary between the NFC controller and the application that it will trigger based on the content within the tag (in this case the browser) to identify malicious web addresses present in the tag. We will discuss about some of the main components of the middleware application in the section below.

Figure 3:Modified NFC Architecture with security middleware



Activity Controller

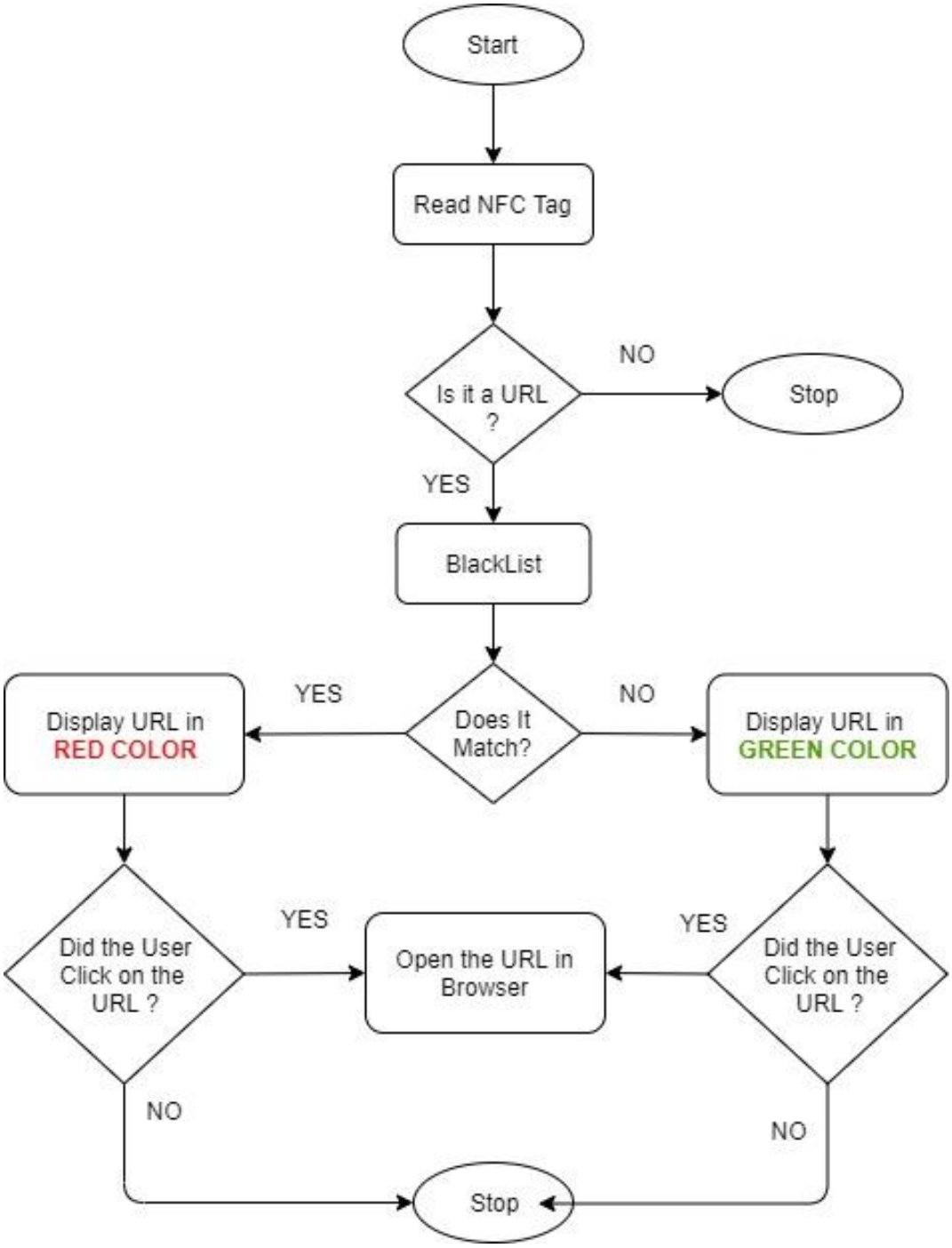
This is the main part of the middleware that manages all of its functionalities. It is positioned ideally between the Application Layer and the NFC controller that handles the read/write interface. When the read/write interface discovers a valid URL from an NFC tag, it is transferred to the activity controller instead of directly opening the browser and loading it. The activity controller uses blacklisting mechanism to check the legitimacy of the URL. Based on the result, it will display whether the URL is malicious or not with the help of colour code. Malicious URLs are displayed in RED colour whereas non-malicious URLs are by GREEN colour.

Blacklisting

In order to check the credibility of the URL retrieved from the tag, we will use the blacklisting mechanism. During this testing, the URL is tested against a list of blacklisted domain names that is encoded within the application. If a match is found, then the URL is displayed within the textarea of the User Interface (UI) with the background colour being RED. On the other hand, if the match is not found, that means the URL is not blacklisted and the text area around

the URL will display GREEN indicating that the URL is safe. This colouring scheme will help the users to identify whether the URL is safe to visit. Thus, our application will add an additional security layer when dealing with malicious URLs.

4.1 Activity Diagram of the proposed Middleware



4.2 Proposed Middleware Algorithm

1. Read Tag
2. If NDEF Record Type = URL then
3. Push URL to Activity Controller;
4. Backlist(URL);
5. If URL Matches with Blacklist then
6. Display URL with **RED COLOR**;
7. Else
8. Display URL with **GREEN COLOR**;
9. End if
10. Else
11. Echo "Not able to detect the URL"
12. End if

5 Implementation

For building the middleware framework, we are using Android platform. Being an open source platform, android provides support and compatibility for the development and deployment of the application. We used Android Studio as the Integrated Development Environment (IDE) for creating the application. The development was done by modifying an existing open source android project[11] taken from GitHub. This project is a simple NFC reader that display basic details about the tag once it gets scanned. We modified it to detect only the URL type content and apply blacklisting before displaying the result in the ScrollView.

5.1 User Interface and Permissions

In order to keep things simple, we have created a simple layout with an ImageView, TextView and a ScrollView. The NFC phone Logo is placed by using the ImageView. The TextView is used for displaying the instruction on how to use the app. ScrollView is where the actual URL will get displayed once the application reads the URL from the tag.

Figure 4 :UI Design

Since our application needs to access the NFC hardware, we need to declare certain permissions within the AndroidManifest file. This file outlines the primary information about the application to the operating system. Therefore, the following

[11] https://github.com/legenddcr/MyNFCReader

permissions needs to be added into the manifest file in order for the application to work properly.

- `<uses-permission android:name="android.permission.NFC" />`

  This will allow the application to require the user's permission to access the NFC hardware for the purpose of reading and writing. Our application will be able to intercept all the NFC communication streams by defining this in the manifest file.

- `<uses-permission android:name="android.permission.VIBRATE"/>`

  This permission is necessary for notification configuration. The application will use this when ever there is an interaction between the tag so that the user can feel it.

- ```
  <uses-permission
  android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
  ```

  This allows the application to get the consent from the user in order to access the external storage for storing the application data.

5.2 Blacklisting

The application is using blacklisting in order to identify the malicious domain names from the NFC tag. The blacklist consists of a series of websites that are considered to be malicious material. We have used squidblacklist[12] which is a URL blacklist service for getting the blacklist for embedding into our prototype application. The picture below shows a small code snippet from the application which is responsible for checking the blacklist.

```
private static String check_safe_or_not(TextView text) {

    List<String> blacklist = new ArrayList<>();

    blacklist.add("https://maliciousssite101.000webhostapp.com/");
    blacklist.add("http://accounts-googlc.com");
    blacklist.add("http://account-spotifyupdt.com");

    for(int i=0;i<blacklist.size();i++){
        if(text.getText().toString().trim().equals(blacklist.get(i))){

            return "B";

        }

    }

    return  "W";
}
```

Figure 5:Blacklisting Function

[12] http://www.squidblacklist.org/downloads.html
```

In Figure 5, initially, we have created an array list to store the blacklist URLs. We used a simple for loop to check if the URL obtained from the tag matches with the URLs in the array list. If a match is found, the function will return “B”, else it will return “W”.

In the figure below (Fig.6), we can see the function being called by the main program. If the return value is “W”, it means the URL is safe, and therefore the background colour of the URL will change to #B0FC58 (hex color code for green). Otherwise, the background colour of the text will get changed to #FF9595(hex color code for red).

```
if(check_safe_or_not(text).equals("W")){

//   text.setCompoundDrawablesWithIntrinsicBounds(0,0,R.drawable.ic_check_mark,0);
   text.setBackgroundColor(Color.parseColor( colorString: "#B0FC58"));

}
else {

   text.setBackgroundColor(Color.parseColor( colorString: "#FF9595"));

}
return text;
```

Figure 6:calling the function in main program

6 Evaluation

6.1 Functional Testing

This testing is done to verify whether the application is performing all the functionalities that we have specified during our requirement analysis stage. The main tasks we want our application to perform are:

1. Detect the NFC tag
2. Read the URL from the tag
3. Display the URL in GREEN colour if it is safe
4. Display the URL in RED colour if it is blacklisted

Functionality 1 – PASSED

We have used the toast functionality[13] to get a simple popup notification whenever a functionality gets executed. This notification will disappear automatically after the timeout that we can set. This is an excellent way to check whether the status of the operation is working.

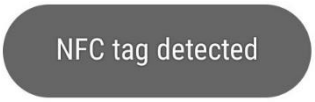


Figure 7:Toast popup when the device detects an NFC tag

[13] https://developer.android.com/guide/topics/ui/notifiers/toasts

Functionality 2 – PASSED

We have used the same toast functionality for checking when the application reads the URL from the tag.

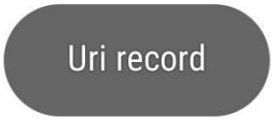


Figure 8:Toast notification when the App reads the URL

Functionality 3 – PASSED

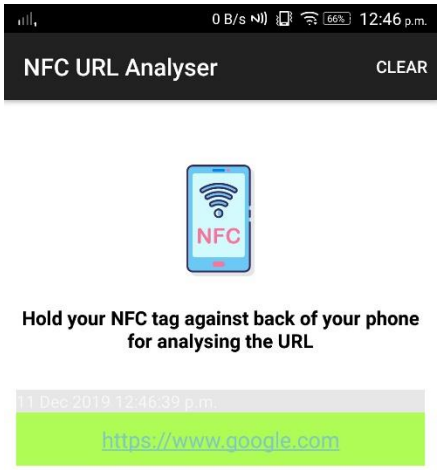


Figure 9:Displaying URL in GREEN color

Functionality 4 – PASSED

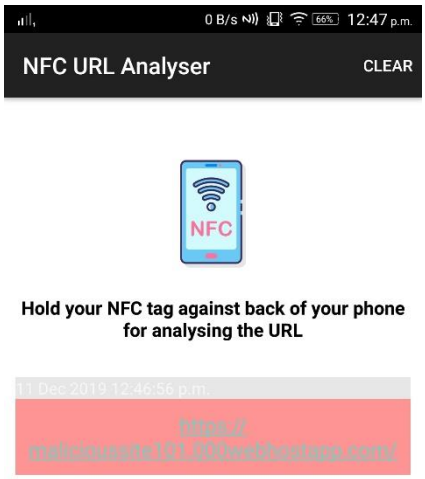


Figure 7: Displaying URL in GREEN color

6.2 Performance Testing

This section consists of the performance assessment of the proposed application for identifying malicious URL from the NFC tag. Our analysis is concentrating mainly on the usage of primary memory (RAM) and the internal storage to see if our application consumes too much system resource. The performance assessment of the application was a tedious task. It involved running the application and reading the tag while extracting the log information generated to find out the time it took to read the tag. We repeated this several times while increasing the number of domain names in the blacklist. The testing URL is placed at the bottom of the list so that it will take the program to go through all the other link before detecting the URL. The result of the test and its impact on memory and storage are mentioned in Table 1.

Some of the main specifications of the device that is used for experimenting is described below.

Model – Lenovo Viber P1 Turbo
OS Version – Android 6 (marshmallow)
RAM -3 GB
Processor – Snapdragon 615 Octa-core processor @1.5GHz

Table 1: Memory, storage consumption based on the size of Blacklist

Blacklist Size	10	50	100	200	300	400	500	1000	2000	4000
Memory usage (MB)	4.9	5.7	5.8	5.9	6.3	6.4	6.6	8	8.1	8.6
Storage (MB)	59.24	59.27	59.27	59.28	59.30	59.32	59.32	59.38	59.49	59.72

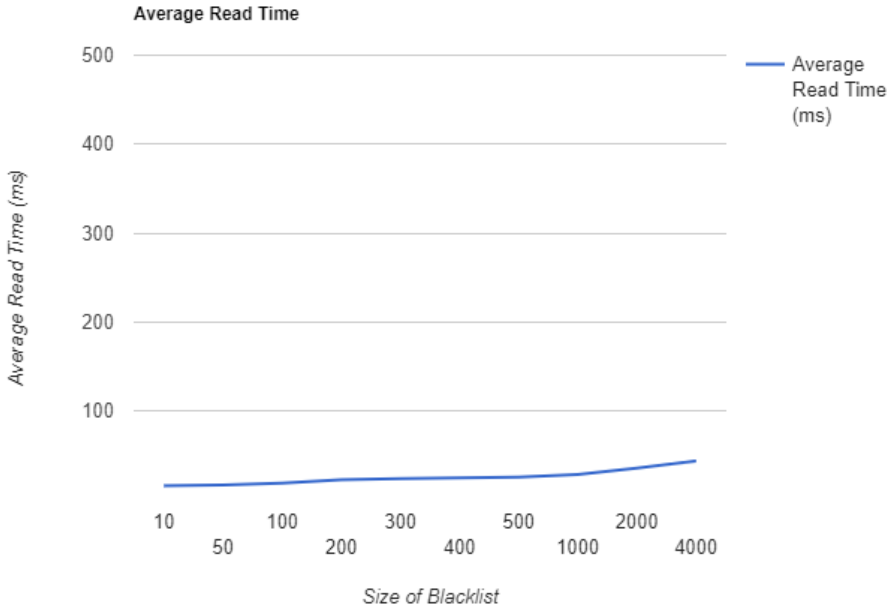


Figure 8: Average Blacklist read time

Table 1 represent the result of the performance testing done with the application. We have performed the test 10 times, each time increasing the number of the Blacklist URLs. Looking into the table in detail, it is clear that as we increase the size of the blacklist, the memory consumption and space the application occupy within the internal storage also increased but at an insignificant level. The average level of memory consumption is 6.63 MB and that of storage is 59.35 MB. Figure 8 shows the average read time of the application which is calculated by analysing the log generated by the Android Studio. On taking a closer it is observed that the average read time is merely 26.8 millisecond. This means that even after performing 10 test cases with different size blacklist, the average read time is always less than one-tenth of a second which is a positive result since NFC applications are not supposed create significant delay while exchanging information. One important thing to notice here is that the effectiveness of the blacklisting depends on the length and quality of the blacklist. The longer the list , the better the chance of detecting malicious URL. However, this method by itself won't completely serve the purpose of identifying bad URL because every day thousands of malicious domain names are generated worldwide[14] and it a difficult job to keep up with this without using a

[14] https://arstechnica.com/information-technology/2012/06/google-detects-9500-new-malicious-websites-daily/

dynamic mechanism such as an Application Programming Interface (API). Because of the time constrain we were not able to implement this mechanism along with the blacklisting however, if given more time we could have integrated them together to improve the current capability of the application.

7 Conclusion and Future Work

In this paper, we introduced the idea for an efficient intermediate application to improve the security of NFC enabled phones by providing protection against malicious NFC tags. A series of papers were reviewed that discussed numerous vulnerabilities and attacks discovered by various researchers in the field of NFC security. After going through some of these works, we realised the lack of security in the NFC tags and the ways it can be modified to exploit a smartphone. This empowered us to propose and develop a secure middleware for facilitating a safer environment for tag interaction. Through our analysis based on functional and performance testing, we identified that the proposed application passed all the required functionalities that we stated during the requirement analysis stage. Furthermore, the level of resource consumption was also fairly low based on the application's evaluation result. However, there is always a trade-off when it comes to performance and security. We can improve the security by making a substantially large blacklist, but in long run it could result in diminishing the performance aspect of the application.

In future, we can integrate the functionality of API along with the present blacklisting technique. This allows in fetching constantly updated blacklisted domain list from third party providers for a real-time comparison. This will help in identifying malicious URLs in an efficient manner with a better rate of detection. It is also possible to extend the functionality of the current application to detect more than just a URL. The NFC tags can have contents other than URL like adding contact information, making a call, SMS, opening custom applications, modifying the device state etc. All these functionalities can also be misused to create security risks as well. Having a single application to identify threats from all these sources will certainly help in maintaining the security of the NFC enabled smartphones.

References

[1] S. Burkard, "Near Field Communication in Smartphones," *Abc*, 2020.

[2] V. Coskun, B. Ozdenizci, and K. Ok, "The survey on near field communication," *Sensors (Switzerland)*, 2015.

[3] N. A. Chattha, "NFC - Vulnerabilities and defense," in *Conference Proceedings - 2014 Conference on Information Assurance and Cyber Security, CIACS 2014*, 2014.

[4] M. Roland and J. Langer, "Digital signature records for the NFC data exchange format," in *Proceedings - 2nd International Workshop on Near Field Communication, NFC 2010*, 2010.

[5] J. Wu, L. Qi, R. S. S. Kumar, N. Kumar, and P. Tague, "S-SPAN: Secure smart posters in android using NFC," in *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM 2012 - Digital Proceedings*, 2012.

[6] M. Gossar, M. Stark, M. Gebhart, W. Pribyl, and P. Söser, "Investigations to achieve very high data rates for proximity coupling devices at 13.56 MHz and NFC

applications," in *Proceedings - 3rd International Workshop on Near Field Communication, NFC 2011*, 2011.
[7] M. Stark and M. Gebhart, "How to guarantee phase-synchronicity in active load modulation for NFC and proximity," in *2013 5th International Workshop on Near Field Communication, NFC 2013*, 2013.
[8] U. Azad and Y. E. Wang, "Direct antenna modulation (DAM) for enhanced capacity performance of near-field communication (NFC) link," *IEEE Trans. Circuits Syst. I Regul. Pap.*, 2014.
[9] Renaud Lifchitz, "Hacking the NFC credit cards for fun and debit."
[10] M. Roland, J. Langer, and J. Scharinger, "Security vulnerabilities of the NDEF signature record type," in *Proceedings - 3rd International Workshop on Near Field Communication, NFC 2011*, 2011.
[11] C. Mulliner, "Vulnerability analysis and attacks on NFC-enabled mobile phones," in *Proceedings - International Conference on Availability, Reliability and Security, ARES 2009*, 2009, pp. 695–700.
[12] W. Issovits and M. Hutter, "Weaknesses of the ISO/IEC 14443 protocol regarding relay attacks," in *2011 IEEE International Conference on RFID-Technologies and Applications, RFID-TA 2011*, 2011.
[13] C. Miller and P. Paganini, "Near Field Communication (NFC) Technology, Vulnerabilities and Principal Attack Schema - InfoSec Institute," *Proc. Blackhat*, p. 2016, 2013.
[14] M. Q. Saeed and C. D. Walter, "Off-line NFC tag authentication," in *2012 International Conference for Internet Technology and Secured Transactions, ICITST 2012*, 2012.
[15] M. Q. Saeed, Z. Bilal, and C. D. Walter, "An NFC based consumer-level counterfeit detection framework," in *2013 11th Annual Conference on Privacy, Security and Trust, PST 2013*, 2013.
[16] M. Roland, J. Langer, and J. Scharinger, "Practical attack scenarios on secure element-enabled mobile devices," in *Proceedings - 4th International Workshop on Near Field Communication, NFC 2012*, 2012.
[17] M. Roland, J. Langer, and J. Scharinger, "Applying relay attacks to Google Wallet," in *2013 5th International Workshop on Near Field Communication, NFC 2013*, 2013.
[18] M. Hutter and R. Toegl, "A trusted platform module for Near Field Communication," in *Proceedings - 5th International Conference on Systems and Networks Communications, ICSNC 2010*, 2010.
[19] M. Reveilhac and M. Pasquet, "Promising secure element alternatives for NFC technology," in *Proceedings - 2009 1st International Workshop on Near Field Communication, NFC 2009*, 2009.
[20] I. Sommerville, *Software engineering (10th edition)*. 2016.