National College of Ireland

YARA Based Defence Mechanism Against NFC Based Attacks For Android Devices

MSc Academic Internship
CyberSecurity

Siddesh Ningappa
Student ID: x19141521

School of Computing
National College of Ireland

Supervisor: Michael Pantridge

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Siddesh Ningappa
Student ID:	x19141521
Programme:	CyberSecurity
Year:	2019-2020
Module:	MSc Academic Internship
Supervisor:	Michael Pantridge
Submission Due Date:	17/08/2020
Project Title:	YARA Based Defence Mechanism Against NFC Based Attacks For Android Devices
Word Count:	6293
Page Count:	19

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:	Siddesh Ningappa
Date:	August 17, 2020

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):	

YARA Based Defence Mechanism Against NFC Based Attacks For Android Devices

Siddesh Ningappa
x19141521

Abstract

After the advent of Near field Technology[NFC] in 2003, it has made a revolution in the field of contactless communication. NFC chips have been implemented in ATM cards, Travel cards tags etc. Smartphones have NFC hardware which supports reading and writing operation to these NFC chips. Another major implementation in the field of marketing is NFC tags. These tags are attached to products just like how bar-codes are attached. These tags are lightweight and can carry information such as phone number, URL etc. The URL could be a link to download mobile application or a link to their products page. When an NFC enabled Android device is in the proximity of such tags, browser makes a request to the URL embedded in these tags. Such tags can be overwritten effortlessly. An attacker can stick NFC tags to any product in the supermarket. These tags may contain malicious URLs and might potentially lead to a malware download source. Scanning such tags will result in Phishing attack. These tags increase the attack surface on android platform and introduce challenges in identifying the source of attack. Embedded malicious links to tags may become fatal for the security posture of the Android device and availability of the data to users .In this paper we will discuss about detection and prevention of such NFC based attacks by leveraging Yet Another Recursive/Ridiculous Acronym(YARA) rules and Virustotal's reliable threat databases.

Keywords : NFC tags, Mobile Security, YARA, NFC attacks. NDEF security, Android Security

1 Introduction

Android devices are so popular in the current generation that almost every individual carry minimum one such device in their pocket.Most of the smartphones comes in-built with NFC hardware.Any vulnerabilities or attacks stemming from such source will have vast attack surface to exploit on.Studying the attacks originating from NFC contributes towards valuable research in the field of information security and being cognizant of such attacks will help the researcher to build security measures around that.These attacks must be stopped at initial stage itself.To start with NFC,it is small range wireless communication technology which leverages radio frequency for communication and implements Simple NDEF Exchange Protocol [SNEP] for Data exchange. The communication between two NFC devices happens when they are in close proximity.The unique nature of NFC is, it supports peer to peer communication[1]. During NFC sessions, one device will acts as passive and the other active.The communication operates in two modes.Active mode and passive mode

During passive mode, when passive hardware(respondent) is in proximity of active device, antenna in the passive device receives electromagnetic signals emitted by active devices and processes the data stored in the chip to active device.Active device will use this information to communicate to the external world via internet.The microchip contains NDEF messages which will be used for communication.The NDEF messages can be of several records[2]. One such record is URL and there are no existing security measures implemented to validate the scanned URL in Android devices. Bad NFC tags potentially contain links to download a malware, dropper or command and control server.

[1] https://blog.atlasrfidstore.com/rfid-vs-nfc
[2] https://www.oreilly.com/library/view/beginning-nfc/9781449324094/ch04.html

Implementing a security check for the URL content in NDEF messages will prevent phishing attacks at the initial phase.The research paper is focused on validating the URL in NFC tags for their maliciousness and build solution to prevent attacks originating from NFC tags.This paper is also aimed to performing penetration testing of the android application so that the product itself is safe and secure to use.The research question would be **Is it possible to defend against the malicious content hiding in the URL component of NFC tags using YARA rules?** The maliciousness of URL can be identified by validating them against list of blacklisted URLs, also it can't be ignored that the URLs might change but the structure ideally remains the same. Most malicious URLs lead to malware download. One proven solution which is implemented to defend against malwares is YARA rules.We will be invoking the YARA rules from a server hosted in cloud environment to validate the URLs also we will be validating against virustotal threat database. This thesis aims at developing an android application which will scan the URL from NFC tags, encode it to base64 and send this encoded content to server hosted in cloud environment. The server hosted in cloud will have YARA running and the encoded content will be decoded and be scanned for maliciousness against pre-written rules and also against virustotal's threat database available online

The topic is worth researching considering the exponential increase in usage of NFC tags for advertisement and the increase in usage of NFC enabled android devices. These vectors increase the attack surface.Due to the cost effectiveness and miniature size of the tags, they are being embedded in every single product.Big brands such as NIKE, L'Oréal, LEGO are leveraging NFC tags for marketing purpose exposing millions of customers smartphones to risk[3].There have been several research papers on NFC implementation such as travel cards, postal systems etc . This paper falls on the similar line where the implementation of NFC is advertising or marketing field and entails with the previous researches carried out.

In this paper,the discussion has been seggregated to multiple sections.Introduction(section 1) , we discussed about the background of NFC Technology and their operation. In Section 2 we will be discussing about the previous researches performed on near field communication and security posture of NFC technology also jist of reader/writer mode in NFC. In section 3, we will be discussing about our approach to develop an android application package(APK) which will identify and mitigate phishing attacks originating from NFC tags by developing . Following section will have the information about application design. Different modules which are required to successfully build the APK file .Implementation section will contain actual deployement and walkthrough of application development, server setup,API calls, responses.This will be followed by evaluation of the implementation.Types of testing performed,parameters which are leveraged to measure and quantify the research.Final section will be dedicated to findings and setbacks of this project which can be used for future work and the outcome of our project

2 Related Work

Before we dive deep into the previous research papers, their strength and drawbacks, let's understand what exactly is Near Field Communication. We are aware that it's being used widely in many industries and the compactness of the hardware is making it flexible for its deployment in multiple industries However,NFC based implementations inherit few disadvantages and vulnerabilities. The section is dedicated to understand the nature of NFC, communication protocols involved, previous works on NFC and security challenges identified by researchers in the fields which collaborate NFC.Initial subsection will be dedicated to describe the hardware components involved for a successful communication, operation mode of NFC which our research is focused on and previous works. This discussion will follow Security posture of NFC based implementations and next sub section focuses on Security Factor in NFC Enabled hardware.The last subsection will be dedicated to detection strategy for phishing attacks.

2.1 Anatomy of Near Field Communication

The communication involves two NFC chip enabled hardware in two modes.Active mode and passive mode.In passive mode, one of the device will have power source attached to it and other lack any power source. The communication between these two devices will be triggered by an active device or else called

[3]https://krify.co/what-are-the-best-use-cases-of-nfc

as an initiator.On the flip side, in active mode both initiator and respondent will have the power source of their own. Both initiator and respondent will be composed of antenna,chip,encapsulating layer

In passive mode,when the respondent is in the close proximity of initiator which is upto than 4 cm, electromagnetic field will be generated by Ferrite coils/Antenna by initiator.This electromagnetic field will power the passive device and respondent modulates this energy to send the data stored in the chip back to the initiator or active device.In active mode,the data will be transmitted by respondent to initiator however it will have its own power source rather than relying on the initiator[4].They operate at a frequency of 13.56 MHz and the maximum data exchange speed supported by these devices wouldn't exceed 424 Kbit/s[5].Near field communication origins from Radio frequency identification.Near field communication chips store data in the form of NFC data exchange format(NDEF) messages. The chips memory varies from 48 bytes to 1 Megabytes which makes it difficult for attackers to store binaries or any malicious programs in the chip itself.

Authors Michael Roland et al(2010)'s, mentions that NFC is just a modified version of RFID which in case is an acceptable fact as both RFID and NFC depends on electromagnetic field for communication and passive mode of operation exists in both RFID and NFC technology [1].There is diminutive difference in the interaction between the hardware used except for the type of data stored in RFID and NFC chips. Both have read/write operation mode but NFC stands out in its peer to peer mode and card emulation mode which we will be discussing in the next subsection.These distinctive features makes it effortless for deployment in smartphones or mobile devices. The following Figure 1 illustrates the electromagnetic field generated by Initiator and the data transmitted by respondent back to the initiator

The author Naveed Ashraf Chattha(2014)'s approach was to highlight the vulnerability which could be originated by data corruption.Data corruption described in the paper was nothing but modifying the content in payload field with dummy data or scrambling the existing data.The attack was executed via Man in the middle attack(MiTM) [2].This vulnerability as addressed by Michael Roland et al (2010)'s with a defensive approach which describes that these payloads could be encrypted for security purpose [1].The discussion is further supported by Thomas et al (2016) who proposed that payload in RFC tags could be encrypted by advanced encryption standard (AES) which is one of the secure encryption technique [3].However their approach lacked the fact that even attacker could encrypt the data in a malicious tag and trigger an attack.Simple example would be attacker may use URL shortener such as bit.ly to shorten his malicious URL and write that URL in the NFC tags. How can this be detected?Though URL shortening is actual encryption, it is a way to hide the data. This issue will be addressed in our paper by implementing YARA rules. As discussed earlier in the paper Near field communication is no different from RFID and this paper focuses on a similar attack emerging in the current world but the way how attack executed is quite different by eliminating the MiTM approach.

2.2 Reader/Writer Mode And The Attack Platform

Stephan et al in their paper discussed about NFC based mobile sales assistant which hasn't been implement in the last decade however is gaining popularity in the last 2years. [4] Cheng Hao Chen et al (2014) in their research paper survey and analysisi of NFC attacks clearly explains the DOS vulnerability in reader/writer mode and also the phishing attack which could happen when the device is in read mode.This paper didn't reveal any methods against mitigation of such attacks [5] Feri et al(2016) had also discussed about the attacks originating from different operation modes of NFC.In their paper the main focus was to carry out a research on denial of service(DOS) attacks which could originate from NFC tags.Another prerequisite for the attack is that the active device must be in reader/writer mode. Hence it is significant for a researcher to be cognizant of different modes which NFC operates.NF operates on 3 modes [6]

- Reader/writer mode
- Peer-peer mode
- Card Emulation mode

[4]https://www.ready4s.com/blog/developers-tips-nfc-tags-in-app-development
[5]https://nfc-forum.org/what-is-nfc/about-the-technology/

Our area of research is reader/writer mode .Feri et al(2016) in their research paper clearly concludes that smartphones operating on NFC hardware has security threat during communication. To overcome such threats,our research contributes in developing a middleware/android application package to evaluate the contents in passive tags while the active device/smartphone is in reader/writer mode. [6]

The approach of Charlie Miller(2016) was to target a NFC enabled smartphone and exploit the read mode.This attack was demonstrated in Defcon 2016 .When NFC is enabled in any smartphone, the hardware will enter read mode and when in proximity of a NFC tag or a card, the content gets executed. i.e. if there is contact stored in the tag, contact gets stored in phone-book of smartphone and if there is URL embedded in tag,browser will request the URL.There was no mitigation proposed in the defcon conference[6].

2.3 Security Posture of NFC Implementation

Midhun Kumar Ayyalraj et al (2019)'s approach was to develop a health management system using NFC which involved tagging an individual's health records to an identification number and storing the data in a passive tag.Many researchers identified that this proposal lacked security feature as the identification number could be cloned and malevolent actors could take advantage of such critical data. An individual's health related is categorized as Personally identifiable information (PII) and any leakage or misuse of such data will have unfavourable outcomes for the organisation which stores and processes the data.Hence this implementation wasn't successful. [7]

Author Zhe Lou(2010) proposed a smart postal system which involved attaching a Radio frequency or NFC tag to the envelope. This tag which gets attached to the envelope will behave as stamp which will be pre-written with some identification number using reader mode of the near field communication and a mobile application to scan or read the content of the tag [8].The approach lacked the security mishap which could happen due to tampering of data. What if the sender already has a malicious RFC tag inside the envelope? Such test cases were never considered in the research Our paper focuses on exploitation of similar feature.

Saminger et al (2013)'s proposal was NFC Ticketing System composes a passive NFC tag which stores information such as an identity related to a individual or an identification number and a reader which sends a update request to server, every time passive tag comes in contact. In the paper the author explains how reader/writer mode can be leveraged for implementing NFC in multiple industries [9]but fails to implement any security measures towards the malicious content hiding in passive tags. This justifies the importance of investing time in researching a defensive mechanism for attacks originating from near field communication based products.

Irene Cappiello et al(2014)'s proposed an NFC based grocery shopping process. This approach had its benefits which involved tag usage.A customer can scan the existing NFC tag at home and add it to the virtual shopping cart.This will the user order the same product again using an app. This comes handy for people who aren't familiar with online shopping. [10]This approach eliminates some drawbacks of normal web based shopping however the security of NFC tags has been overlooked and the paper hasn't paid any attention towards NFC based phishing or DOS attacks.

Kimberly Golds et al (2015) describes that NFc tags acan be embedded with a fake poster which has a malicious URL embedded in the poster and when a victim scans such tags, they will be redirected the phishing sites and the attacker might potentially trick the victim by posing as a legitimate website and inturn victim may enter sensitive information such as login credentials of his personal mail id [11].This paper describes how an NFC tag could be programmed such that the default browser on smartphones could make requests to malicious pages.These malicious pages may be phishing or malware download links or Ransomware URLs. The paper didn't highlight any counter measures .Our paper focuses on identification such android based malwares and phishing sites.

[6]https://www.defcon.org/images/defcon-20/dc-20-presentations/Miller/DEFCON-20-Miller-NFC-Attack-Surface.pdf

Vidas et al 2013 has demonstrated a QRphishing attacks on smartphones . Their experiment proved that a Trojan could be installed on smartphones just by scanning a QR code.This installed Trojan can launch phishing activities also send spam messages.However there wasn't any antiphishing mechanism proposed in the paper.QR scanning is no different from NFC scanning as the data which could be place in QR can also be placed in NFC tags as well. Our paper introduces a solution to solve such paradigms of phishing attacks [12].

2.4 Security Factor In NFC Enabled hardware

The approach of Naveed Ashraf Chattha(2014)'s was to propose the prevalent vulnerabilities in NFC environment.Few Vulnerabilities identified in the paper were data corruption,eavesdropping, data modification, data insertion and MiTM [13] but no preventive measures were suggested in their paper.Similar critical vulnerability was identified by Zhao Wang et al(2016) which is called as NFC relay attack but the approach targeted peer to peer mode of NFC technology [14] which is not in scope of this research topic. Security of NFC card could have been bypassed by duplicating the card or by card cloning. However this was mitigated by introducing NDEF Signature which checks the authenticity of the card.

As discussed above,Card Emulation mode is protected from vulnerabilities as there is a distinct chip will be used for transactions and peer to peer mode is protected by considering the trust factor of two devices involved in transaction but the reader writer mode could issue the greatest threat which could originate from NFC tag where URL value could be modified to malicious content. The authentication process of these passive tags could be increased by implementing online authentication which would require users to login to a portal and scan the NFC tag to verify the unique ID stored in tag and process the URL parameter post successful validation of the unique ID.NFC tags are being implemented as smart notice boards [15] and Focusing on implementation on NFC tags in marketing where tags are attached to products.Implementing online authentication in such scenario will increase latency and overhead as the user must have to create account with every vendor he does transaction failing to do so will reject processing the content in NFC tags.This will restrict the vendors in implementation of simple unauthenticated URLs. John et al(2015) suggested attendance model using NFC. Their approach had a second factor for authentication which is one time password(OTP).The NDEF record used in their approach was studentid/identification number.Our project focuses on a different record than the one they used and OTP can be used as a mechanism to protect phishing sites.Their focus was to focus on integrity factor in the CIA (Confidentiality,Integrity,Availability)triad of security. Our project is close to that of their approach but focuses on all 3 factors because a phishing link may compromise all 3 factors [16]

2.5 Detection Strategy For Phishing Attacks

Shraddha Parekh et al (2018) demonstrates a machine learning based approach to identify the phishing URLs. In their paper they leverage feature extraction using different algorithms such as Naive Bayes, Random Forest which is efficient however time consuming and reduces the throughput in the detection [17].FireEye, one of the cyber security security pioneers in one of their blogs demonstrate how phishing attacks could be tackled with the help of YARA rules and also they highlight the importance of lightweight and faster mechanism in detecting the malicious URLs. [7]

3 Research Methodology

Considering that our area of research is android and the project aimed at developing an android application which will act as a middle-ware between the NFC hardware on smartphone and the browser.Radek Fujdiak et al 2019 in their paper describes the need of Secure Software Development Life Cycle(SSDLC) [18] while building any application.Our research paper is developing a middleware for Android platform. Hence SSDLC has been followed to develop the application.This means security has been injected in every stage of the development cycle.

[7] https://www.fireeye.com/blog/products-and-services/2018/16/detect-and-block-email-threats-with-custom-yara-rules.html

3.1 Approach Overview

We identified that the problem statement or threat is URL parameter in NFC tags. Hence identifying a way to validate the URL before it is being requested by the browser and execute phishing attack could potentially solve the problem.Labelling a website as malicious or not could be achieved by comparing it against list of blacklisted addresses.The challenge is the list is never ending and new domains gets registered everyday and identifying the maliciousness of these websites and the content available in the link.This could be tackled by validating the URL content against YARA rules and also against the threat database which is up to date and available free of cost for researchers.The platform leveraged in this researh paper is virustotal.

As we will be following SSDLC cycle, there will be several stages in the application development which are analysis,design,implementation, testing and integration, maintenance. Following figure visualises the SSDLC cycle [8]

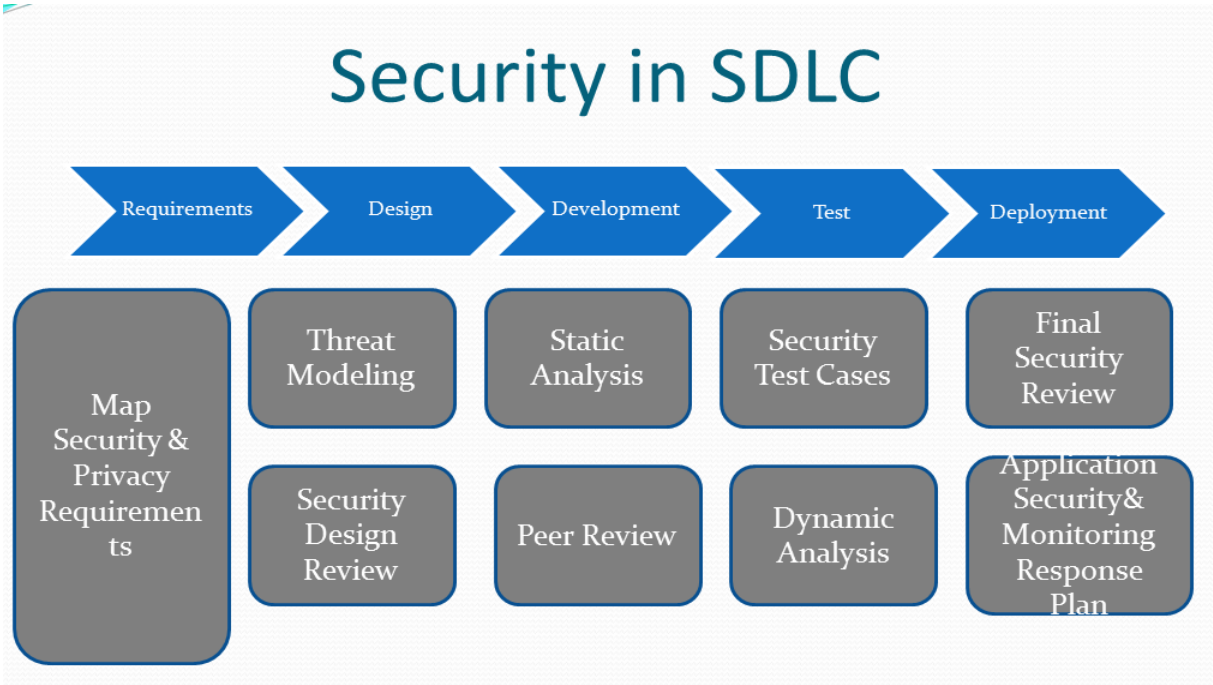

Figure 1: Secure Software Development Life Cycle

3.2 Analysis

This stage is more or less like a requirements gathering stage.Our requirements is to develop an android application package which could read NFC tags. We would need NFC tags for testing and YARA libraries to test for maliciousness of URL . As we were building the android application,we chose JAVA as programming language to code the application but there are no inbuilt YARA libraries in JAVA.So our requirement was extended to host a Ubuntu server and run YARA rules on that server. So our requirement is now to scan the URL from NFC tags and send the URL to Ubuntu server to validate the maliciousness and to respond with a '0' or '9'. if the response is 0, URL will be declared as red(malicious) else it will be sent to virustotal for further validation.I have registered on virustotal website for a free API key and the key has been used in this project to validate the URLs received from NFC tags. If Virustotal's detects as malicious, the URL will be displayed in red else it will be displayed in green

3.3 Architecture Design

Secure designing principles have been implemented while designing the application.Security has been in the front seat throughout the design process.We have designed a simple user interface which will scan

[8] https://www.ecomsecurity.org/web-application-security/six-best-practices-for-security-testing-in-the-sdlc/

the NFC tag, send the request to cloud server and respond with red or green toast message on the app based on maliciousness of the URL.No insecure JAVA libraries have been used during the development and ubuntu server is hosted in Amazon Webs Services(AWS) cloud with security in mind. iptables rules have been written to white-list the ssh access and also the server is running with low privilege user not root.

3.4 Development

As the platform chosen for research is android, we will be building android application in Android Studio.There were two options to code an android application JAVA and Kotlin. JAVA has been chosen because of the familiarity in the language and the broader open source support availability for any issues if encountered during the development. The project skeleton code was downloaded from git repositories. The code downloaded had the capability to read NFC tags. This code has been tweaked to read the URL content in the NFC tags.Simple user interface(UI) has been designed for this purpose For implementation,we will be leveraging open source Android software development kit(SDK).Android Studio will be used as integrated development environment(IDE) for application development.The programming language used will be JAVA.EC2 instance has been spin up on AWS. Nginx server has been hosted on the server . YARA has been installed on the Ubuntu server. A php file on Nginx server is coded to recieve the request from our android app.Decode the URL and validate against the YARA rules.Security factor involved at this stage are YARA and Nginx are running as low privilege user.Virustotal's free API key has been used to validate the URLs which aren't reported as malicious in YARA rules.

3.5 Integration

Security testing has been performed at every stage of the application development.Secure code review has been performed after development of each functionality and once the application is built, dynamic application security testing of the application has been performed also the Ubuntu server has been scanned for unnecessary open ports and has been hardened to make sure that all unnecessary services have been removed.Communication between the android application and Nginx server is unencrypted and running on http for now. This will be considered as limitation of the project. Virustotal's free API key has limitations. The number of requests which could be sent by an user for validation are 4 requests per minute .The application has been integrated successfully such that all 3 (blacklist,YARA,Virustotal)modules work perfectly in sync.

3.6 Maintenance

This will be the last stage in SSDLC cycle and application maintenance will be carried out by fixing any software bugs if occur in future,tuning the application based on users input/feedback, update the web application server to the latest release, Patch any kernel related vulnerabilities as and when they are discovered. Update YARA version on the go and push YARA feeds to the Nginx server frequently so the detection percentage will be increased and also to remove false positives if any identified and reported. Monitoring the logs of the server for any suspicious activity.provide backward compatibility for older android versions when they approach end of life. As the number of users increase, Virustotal API key must be upgraded to premium.More information will be discussed in the design and future work section

4 Design Specification

We have discussed in the previous section abou the vulnerabilities existing in an application or application activity could compromise the whole device itself.The application will be designed to start along with other android services and run in the background.The application will be part of startup services and will become active as soon as the smartphone boots up.The application focuses only on URL parameter of the NDEF record and nothing apart from that.As there is a hardware interaction involved in the process, following is the proposed architecture design for building the solution against phishing attack on NFC enabled Android devices.Android components which play vital role in designing the solution are discussed in detail below.

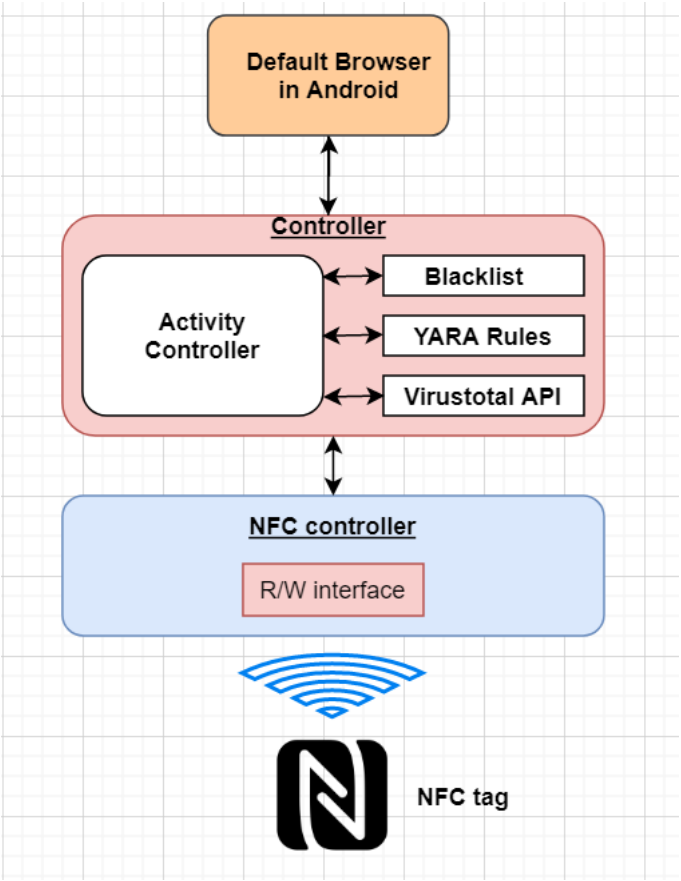

Figure 2: Architecture design for NFC interaction

4.1 Virustotal API

Virustotal has a huge database of malware hashes, blacklisted command and control server domain names, reported phishing websites. It also does the threat hunting when it comes to phishing URLs,if the malicious URL is hidden with the help of URL shorteners such as "www.shortur.at" or "bitly.com", it will still detect the hidden content.The question was how do we leverage this facility in our application. So, We have registered anaccount with virustotal.com and every registered user gets an API key with limited features.This free API key has been used in the project. [9]

4.2 YARA Tool

YARA tool is predominantly used in malware analysis.[10] It is lightweight and easy to deploy as well.IT was challenging to implement YARA in the apk which will be developed as the application being developed is based on java programming language and java does not have any in-built YARA libraries.Design an application with YARA demanded an alternative solution.keeping this in mind application architecture was designed to send an API call for YARA. There are no publicly hosted YARA servers just like Virustotal. Hence YARA tool must be hosted locally or somewhere in the cloud. Hosting the tool locally will work fine during testing but in production, it will create issues Hence AWS's ec2 instance will be used to host YARA.Installing YARA tool will not solve the problem. The tool must be fed with multiple rules .These rules were created based on regular expression validation.

4.3 Cloud Environment

AWS is the cloud service provider which has been leveraged for the purpose of project.Infrastructure as a service(IaaS) is the environment opted for the purpose of thesis.Ubuntu server has been deployed on

[9] https://developers.virustotal.com/reference
[10] https://blog.malwarebytes.com/security-world/technology/2017/09/explained-yara-rules/

AWS environment.Nginx 1.10.3 server has been deployed inside Ubuntu.YARA has been deployed on top of ubuntu. php script has been written to decode the base64 encoded URL and to invoke YARA tool for validating the decoded URL.

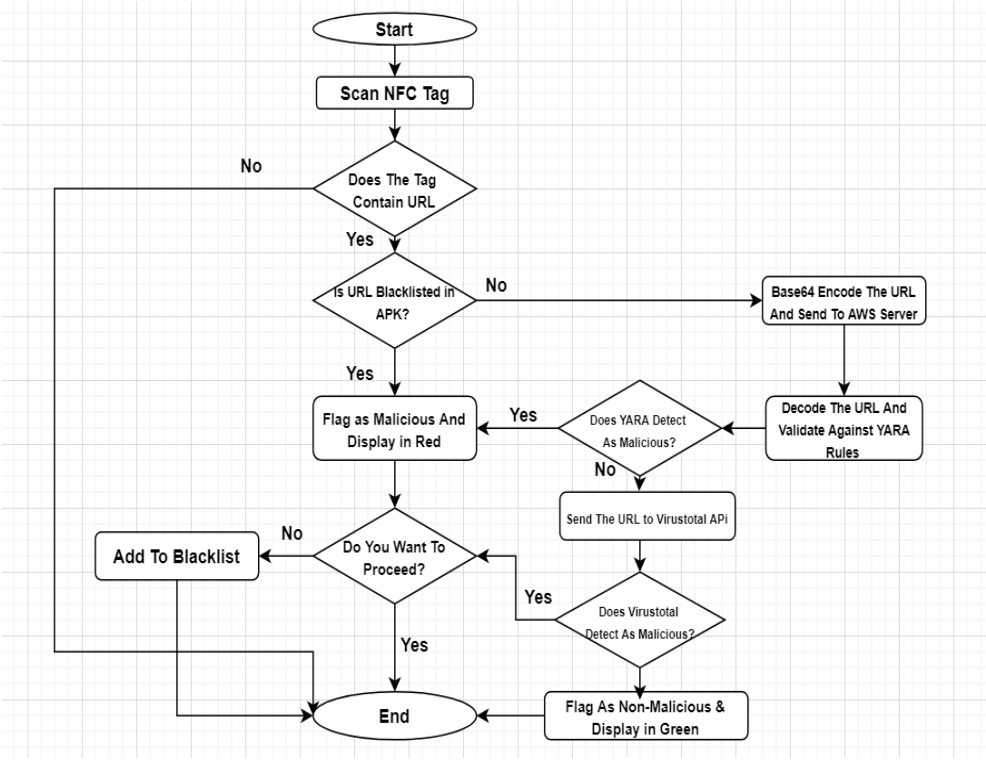

Figure 3: Flow Chart

4.4 Proposed Algorithm

1. Scan NFC tag
2. Invoke NFC hardware module
3. **If**, there is URL parameter in tag, then
4. Parse the data to Activity controller
5. Cross verify the URL with Blacklisted records
6. **If**, there is a match then
7. Tag the URL as malicious and display in Red
8. **Else**, base64 encode the URL
9. Send the URL to YARA client on AWS server
10. Decode the URL content and run YARA
11. **If** ,there is a match then
12. Tag the URL as malicious,display in Red
13. **Else**, send the URL to Virustotal via API call
14. **If** ,there is a match then
15. Tag the URL as malicious,display in Red

16. **Else** Display the URL in Green
17. End **if**
18. End **if**
19. End **if**
20. **Else** Echo "No URL record"

5 Implementation

The implementation was challenging as it involved multi stage parsing of the URL content.Components were built one after the other.Only after successful completion of 1st phase of detection,next module was built. Initially activity controller was built with a skeleton code available on github repository.This code has been modified further.The main modules of the implementation have been explained below

5.1 Activity Controller

When NFC enabled Android device scans an NFC tag,the data/signal is processed by activity controller.i.e.Activity controller decides the next course of action.The normal behaviour is that if an NDEF record contains URL parameter,the URL will be requested in browser.In the implemented architecture,the android app will have permission to access NFC hardware in turn read the contents of NFC tags.if the tag contains URL parameter then the URL will be validate against the list of hard-coded blacklisted addresses and processed from there

5.2 Blacklist

The JAVA code will have list of hard coded website addresses which are labelled as dangerous/malicious websites.This blacklist is a extracted from the list of phishing sites available on internet.If the NDEF record matches any of the blacklisted record, URL will be displayed in red confirming it's maliciousness and the process will halt there. If the URL doesn't match any of the data in blacklist file, then next step will be sending it to cloud server by performing a base64 operation.

5.3 YARA Detection

Continuing from the above step,If the NDEF record doesn't match any of the addresses in blacklist, the URL will be base64 encoded in the application and passed as a url parameter to the variable declared as "yara_url" in the code.Once received at the server,this url will be decoded to plain text by **"yara.php"** file present on nginx server.This decoded will be stored in a file named "**results.txt**" for further processing. YARA tool deployed on the server will have YARA rules written and saved in file named "**test.yar**". These YARA rules are basically regular expressions constructed based on different malwares traffic data. The **Result.txt** file is run against test.yar file for potential malicious content . If there is a match between the content saved in results.txt file and the regular expressions present in test.yar file, the server will return '0' which is termed as bad URL and the URL will be displayed in red and as malicious on the mobile app.This will be the end of process if the data is identified as malicious.

5.4 Virustotal API Validation

If the URL isn't identified as malicious in YARA rules, then the server will respond with '9' and this will trigger an API call to virustotal where the URL will be validated against huge threat database.If the URL is identified as malicious with more than four of the antivirus vendors,this will be flagged as malicious in the application else, it will be displayed in green.Virustotal is a platform to verify malwares, phishing URLs against list of AV vendors and provide results in realtime. Virustotal downloads the binary and run against the list of antivirus for the maliciousness of binary. Virustotal intelligence searches through its datasets for Binary properties, Detection verdicts,Static properties,Behavior patterns and Submission metadata[11]

[11] https://www.incidentresponse.com/wp-content/uploads

```
ubuntu@ip-172-31-42-94:/var/www/html$ cat yara.php
<?php

$a = $_GET['url'];

$file = 'result.txt';
file_put_contents($file, base64_decode($a), LOCK_EX);

$r = shell_exec('yara -s test.yar result.txt');

if (strpos($r, '0x') !== false) {
        echo 0;
        }
else{
        echo 9;
}

?>
```

Figure 4: screenshot shows the code snippet running on YARA server

The plan is to develop an Android application package(apk) an d install on NFC supported android phones. This apk file has 3 views.TextView which will request the user to scan the NFC tag. ImageView is used to display NFC symbol in an Image and doesn't have any role apart from the graphical representation of NFC.Following is the glimpse of the app once installed on android phone. Initial step is to provide the application the permissions to access NFC hardware by defining the following in AndroidManifest.xml file.

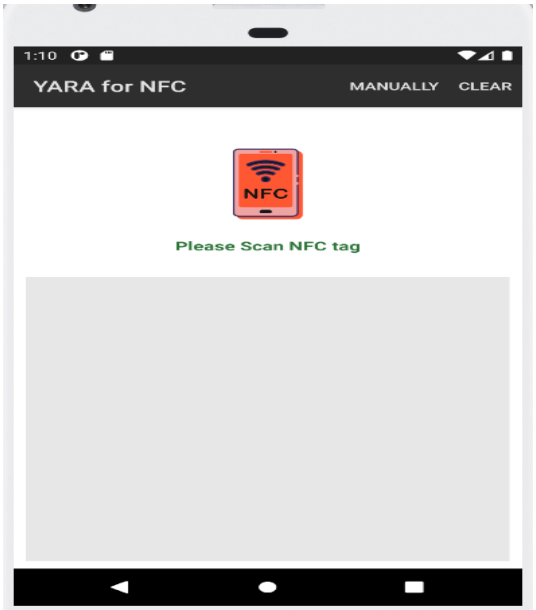

Figure 5: User Interface

<uses-permission android:name="android.permission.NFC" />
<uses-feature android:name="android.hardware.nfc" />

We can leverage "**NdefRecord**" class in JAVA to read the content of tags.When the phone scans the NFC tags, application will read the NDEF record, if it is a URL record, then the content will be 1st validated against the list of blacklisted addresses which will be hard coded in the program.If the URL doesn't exist in the list, then the URL will be base64 encoded and sent as a parameter to following Nginx server hosted on AWS.

http://54.204.6.177/yara.php?url=(base64 encoded url)

AWS Elastic IP has been created and has been attached to the ec2 instance in AWS to make sure that the public IP (54.204.6.177)will not change when the server reboot.

There are 2 files in the project where this URL must be mentioned one is network_security_config.xml file second file is YARA_Api class file URL is handled by yara.php file, decode the base64 data and ran against the YARA rules.

If the URL matches with any of the YARA rules ,it will return '0' and be flagged as malicious and the URL on the mobile app will be appear in red else it will return '9' .Response 9 will trigger a virustotal API request for further validation and the URL will be checked against threat database(virustotal) online

```
<network-security-config>
    <domain-config cleartextTrafficPermitted="true">
        <domain includeSubdomains="true">54.204.6.177</domain> <!-- Yara AWS server -->
    </domain-config>
</network-security-config>
```

Figure 6: network_security_config.xml

```
private static final String TAG = Yara_API.class.getName();
private Activity _activity;
private RequestQueue mRequestQueue;
private StringRequest mStringRequest;
private String yara_url = "http://54.204.6.177/yara.php?url=%s";
private String _analyse_tag;
private YaraLinkAnalysis linkAnalysis;
private ILinkNotify linkNotify;

public Yara_API() {
    this.linkAnalysis= new YaraLinkAnalysis();
}

private String encode_2_base64(String clearText) {
    return Base64.encodeToString(clearText.getBytes(),Base64.URL_SAFE);
}
```

Figure 7: YARA_Api class file

.If there are more than 4 positives detected in the json response from virustotal API, the URL will be tagged as malicious and displayed in red on the mobile app.This decision must be revisited in the future to tune the application performance and detection ratio.

6 Evaluation

The application capability to detect NFC tags has been tested on oneplus 7 android phone.Following is configuration of the smartphone

- Android version 10,
- RAM 8GB,
- 256GB of ROM space
- snapdragon 855

The developed application was successful in detecting the tag and detection time is less than a second.In the next stage, the application has been tested against both benign and malicious URLs.Multiple testing approaches have been taken to evaluate the security posture of the application, performance and usability of the application.All the URLs which were hard coded and tagged as bad in the APK were successfully identified as malicious.YARA rules were successful in detecting the malicious URLs which were extracted from the following website

"https://www.malware-traffic-analysis.net/".

20 different YARA rules were written against 84 different malicious URL structures and had a success rate of 100 percent while detecting the malware traffic for which rules were written. over 50 benign URLs were scanned and passed to the developed application for testing.They passed the initial phase of blacklist validation and were passed to yara tool for further screening. YARA detection was succesful in identifying these URLs as benign and had zero false positives. All the benign URLs were passed to virustotal for further analysis. These URLs were tested against virustotal database as well. several trials were performed to identify the false positives and false negatives.Virustotal identified many websites as malicious as it rates based on the Antivirus vendors.Few websites which are built for educational learning purpose and are vulnerable in nature have been tagged as malicious.But YARA rules came out successful in evaluating such domains.The response time of YARA server was faster compared to that of virustotal response.YARA server took 138 milliseconds to detect the URL as malicious but virustotal took almost thrice the time which is 415 milliseconds.

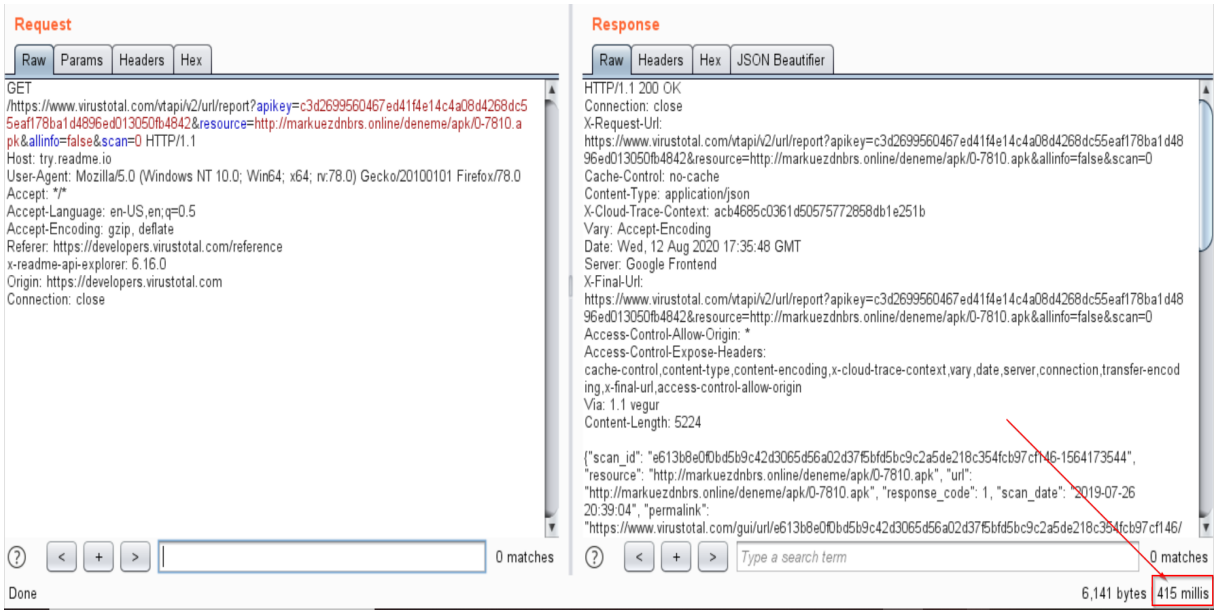

Figure 8: Screenshot shows the response time from Virustotal

Virustotal API calls will be served with Javascript object notation(JSON) response and the maliciousness of the URL is decided on the number of positives reported by the virustotal.This parameter is set to 4 for the purpose of the project and was changed to 2 and 3 during the testing.This change increased the number of false positives being reported.Hence after testing the threshold level has been set to 4. Following snapshot shows the parameter **"positives"** which has been leveraged for this project

Static and dynamic security testing has been performed on the application.Mobsf has been used as static application security tool. It has identified the below vulnerabilities related to permissions in androidmanifest.xml file.From the figure,it is evident that there are 3 permissions enabled for the app .All the above permissions are necessary to run the application . Internet access is required to validate the URL against virustotal database and against YARA rules.NFC is required to scan the NFC tags and read/modify/delete SD card contents to update the blacklist .APK file has been reversed using **dex2jar**

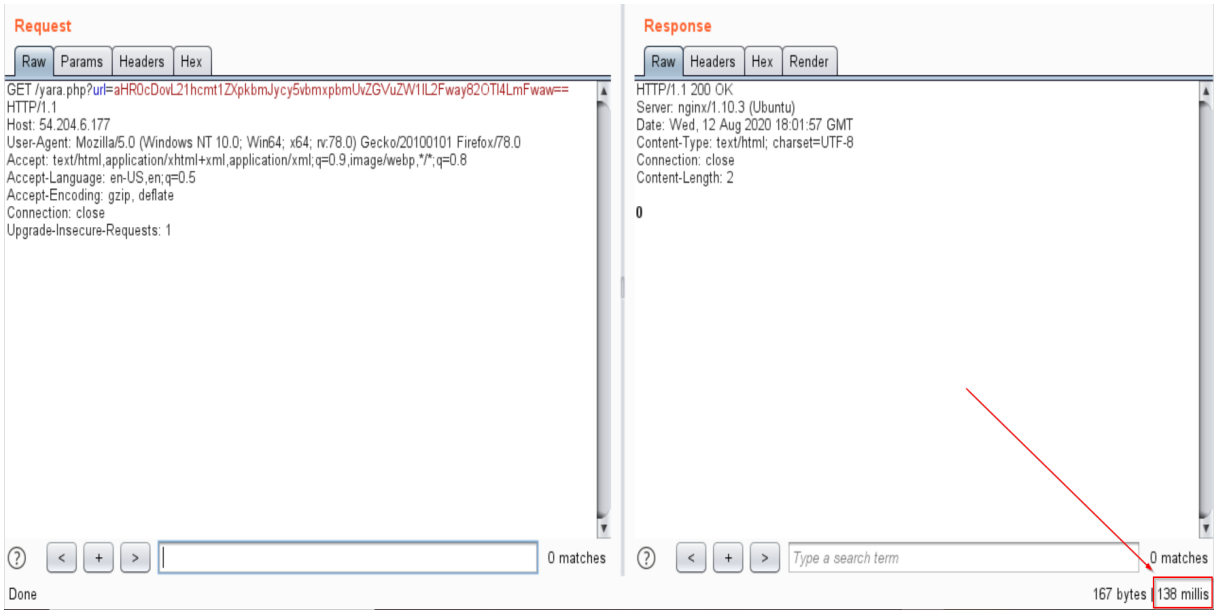

Figure 9: Screenshot shows the response time from YARA server

Figure 10: JSON response for Virustotal API request

and **jd-gui** has been used to view the structure of the application and verified that there is no sensitive information leakage.screenshot of jd-gui has been attached in the appendix.

Drozer has been leveraged for extended part of security testing. Initially **genymotion** has been installed on the host machine using oracle virtualbox as hypervisor. Google Pixel 3 is the virtual android device which has been spun on genymotion emulator.Android debugging bridge(adb) and drozer server has been installed on kali linux. drozer agent and our application has been installed on google pixel3 virtual device. Following commands have been run after establishing connection with the emulator via drozer.application package listing, attack surface listing and connection to the activities listed from drozer has been attempted . To begin with the testing, our application was listed with the help of command **"run app.package.list"** our application package name was identified as com.iot.nfcreader.More information about the package was extracted by running the command **"run app.package.info –a com.iot.nfcreader"** further activities within the package were enumerated as well using the command During the testing process, each and every activity was started to see if there is any information leakage but nothing found and permission was denied for the google firebase activity. The command to start the activity is as follows **"run app.activity.start –component com.iot.nfcreader com.google.firebase.auth.internal.FederatedSignInActivity"**[12] [13] The above results showcase that the application is secure but still contains sensitive information such as API keys.The limitation of project is http connection for YARA detection.As the data is being sent AWS is in cleartext an attacker can perform man in the middle (MiTM)attack by sniffing the traffic.MiTM might potentially result in malfunctioning of the application and end up producing huge number of false positives. We will be considering this aspect in our future work.

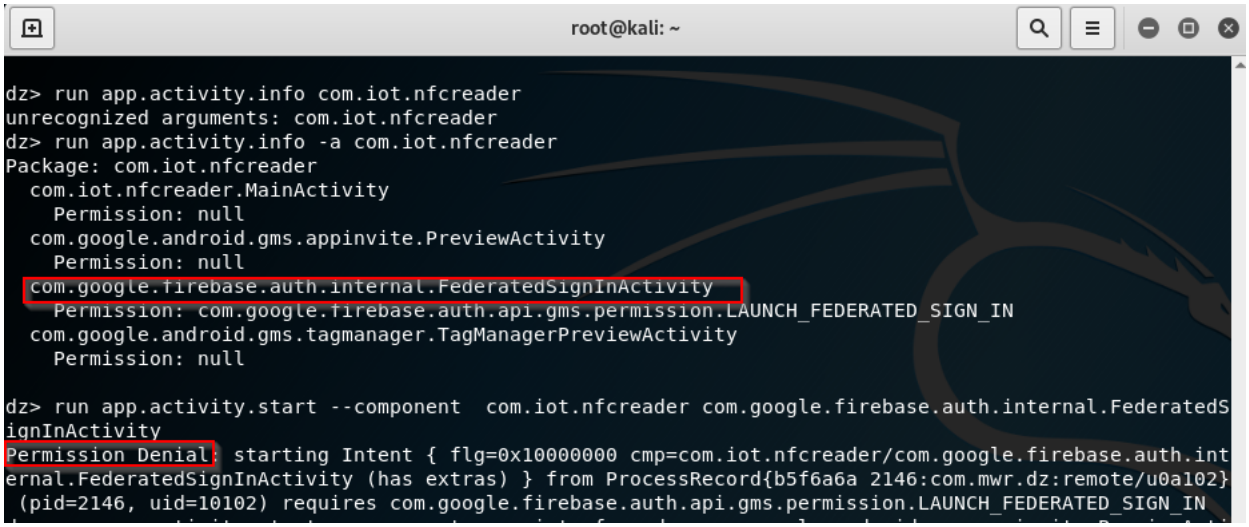

Figure 11: drozer Activity exploit

7 Conclusion and Future Work

As discussed in the introduction, we have developed an android middle-ware which will scan the NFC tags for URL content. The scanned data will undergo multiple checks to identify potential phising links or malicious data.This action is performed with the help of hardcoded blacklists,YARA rules and virustotal database which is a efficient and reliable source.We have understood the significance and the swiftness of YARA rules in responding to the malicious payloads.The paper has focused on SSDLC to build the application, YARA rules to detect the malicious content, leveraged cloud environment which in a way is secure by nature and also performed security testing on the developed application.This concludes that the paper has contributed more towards the security aspect.

The project is mainly focused only on the URL record of NFC tags. The tag may contain other information as well.The YARA tool must be fed with updated YARA rules to detect new malware/phishing links. The application could be enhanced to identify malicious content hiding behind all other NDEF records.The server where YARA is hosted must be protected by distributed denial of service(DDOS)

[12] https://blog.dixitaditya.com/android-pentesting-cheatsheet/
[13] https://resources.infosecinstitute.com/android-penetration-tools-walkthrough-series-drozer/

attacks. This could be achieved by implementing or signing up for AWS shield from amazon. The detection per minute is limited to 4 request per minute for virustotal lookup. This could be enhanced by subscribing for virustotal premium API key.The API key has been hardcoded in the source coded for the purpose of the project.In future this API key can be stored in cloud key management systems such as Hashicorp key vault. Azure key vault or AWS key management system. The communication between YARA server and the application is in cleartext (http).This could be enhanced to https connection by purchasing secure socket layer(SSL) certificate.The Ubuntu server is running with 3GB random access memory(RAM).The RAM must be increased in the future.The YARA request is happening through the public IP address which is not advisable and as part of future work,a domain name must be purchased and PTR record could be added so that the IP will be mapped to the purchased domain name and future requests can be made through the domain name.

References

[1] M. Roland and J. Langer, "Digital signature records for the nfc data exchange format," in *2010 Second International Workshop on Near Field Communication*, pp. 71–76, 2010.

[2] N. A. Chattha, "Nfc — vulnerabilities and defense," in *2014 Conference on Information Assurance and Cyber Security (CIACS)*, pp. 35–38, 2014.

[3] T. Plos, M. Hutter, M. Feldhofer, M. Stiglic, and F. Cavaliere, "Security-enabled near-field communication tag with flexible architecture supporting asymmetric cryptography," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 11, pp. 1965–1974, 2013.

[4] S. Karpischek, F. Michahelles, F. Resatsch, and E. Fleisch, "Mobile sales assistant - an nfc-based product information system for retailers," in *2009 First International Workshop on Near Field Communication*, pp. 20–23, 2009.

[5] C. H. Chen, I. C. Lin, and C. C. Yang, "Nfc attacks analysis and survey," in *2014 Eighth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pp. 458–462, 2014.

[6] F. Fahrianto, M. F. Lubis, and A. Fiade, "Denial-of-service attack possibilities on nfc technology," in *2016 4th International Conference on Cyber and IT Service Management*, pp. 1–5, 2016.

[7] M. K. Ayyalraj and S. A. A. Balamurugan, "Patient health description using nfc-tag-m-health," in *2019 Amity International Conference on Artificial Intelligence (AICAI)*, pp. 642–646, 2019.

[8] Z. Lou, "Nfc enabled smart postal system," in *2010 Second International Workshop on Near Field Communication*, pp. 33–38, 2010.

[9] C. Saminger, S. Grünberger, and J. Langer, "An nfc ticketing system with a new approach of an inverse reader mode," in *2013 5th International Workshop on Near Field Communication (NFC)*, pp. 1–5, 2013.

[10] I. Cappiello, S. Puglia, and A. Vitaletti, "Design and initial evaluation of a ubiquitous touch-based remote grocery shopping process," in *2009 First International Workshop on Near Field Communication*, pp. 9–14, 2009.

[11] K. Gold, S. Shetty, and T. Rogers, "A testbed for modeling and detecting attacks on nfc enabled mobile devices," in *MILCOM 2015 - 2015 IEEE Military Communications Conference*, pp. 635–640, 2015.

[12] T. Vidas, E. Owusu, S. Wang, C. Zeng, L. Cranor, and N. Christin, "Qrishing: The susceptibility of smartphone users to qr code phishing attacks," pp. 52–69, 04 2013.

[13] M. Roland, J. Langer, and J. Scharinger, "Security vulnerabilities of the ndef signature record type," in *2011 Third International Workshop on Near Field Communication*, pp. 65–70, 2011.

[14] Z. Wang, Z. Xu, W. Xin, and Z. Chen, "Implementation and analysis of a practical nfc relay attack example," in *2012 Second International Conference on Instrumentation, Measurement, Computer, Communication and Control*, pp. 143–146, 2012.

[15] G. P. Rajesh, P. Pattar, M. N. Divya, and V. Prasad, "Near field application: Nfc smart notice board," in *2016 Thirteenth International Conference on Wireless and Optical Communications Networks (WOCN)*, pp. 1–5, 2016.

[16] J. Jacob, K. Jha, P. Kotak, and S. Puthran, "Mobile attendance using near field communication and one-time password," in *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, pp. 1298–1303, 2015.

[17] S. Parekh, D. Parikh, S. Kotak, and S. Sankhe, "A new method for detection of phishing websites: Url detection," in *2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT)*, pp. 949–952, 2018.

[18] R. Fujdiak, P. Mlynek, P. Mrnustik, M. Barabas, P. Blazek, F. Borcik, and J. Misurec, "Managing the secure software development," in *2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, pp. 1–4, 2019.

8 Appendix

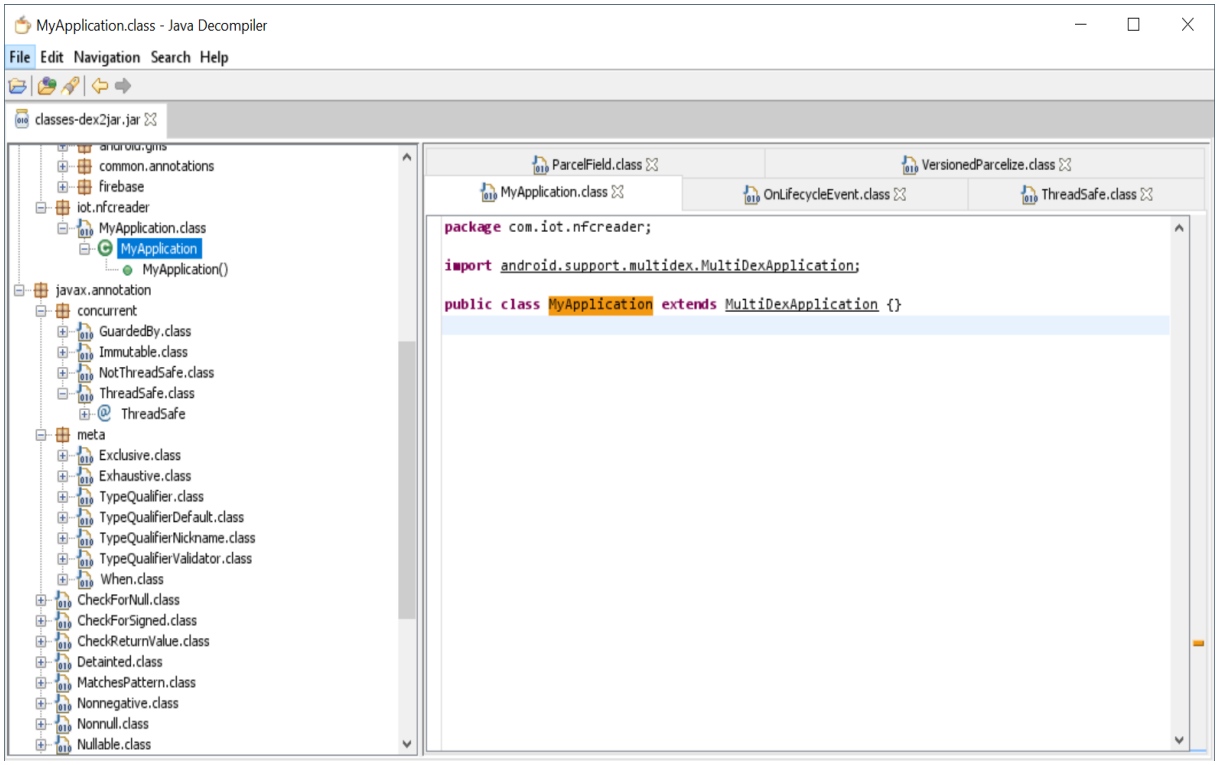

Figure 12: screenshot shows the reversed apk file using the tool JD-GUI

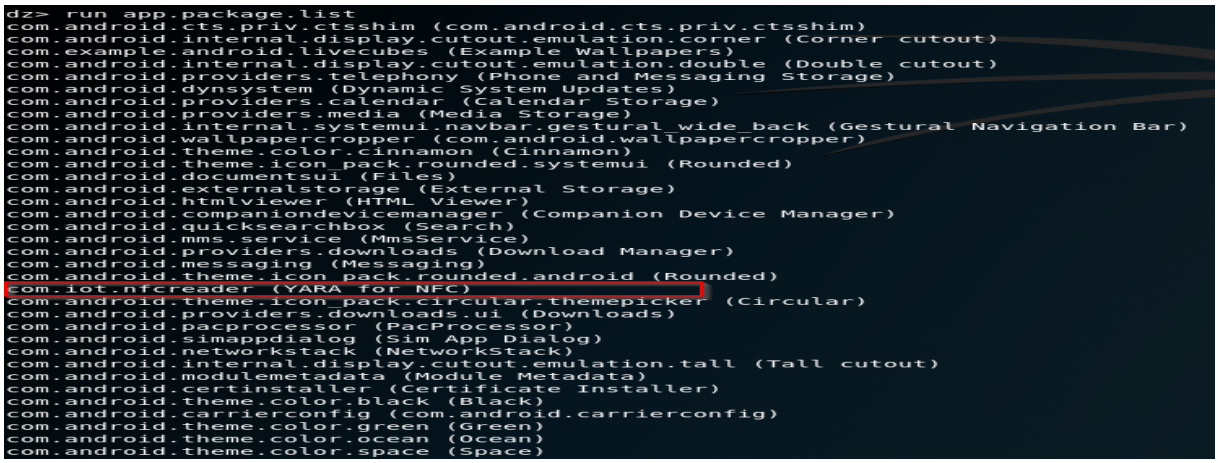

Figure 13: screenshot shows the drozer's package listing command

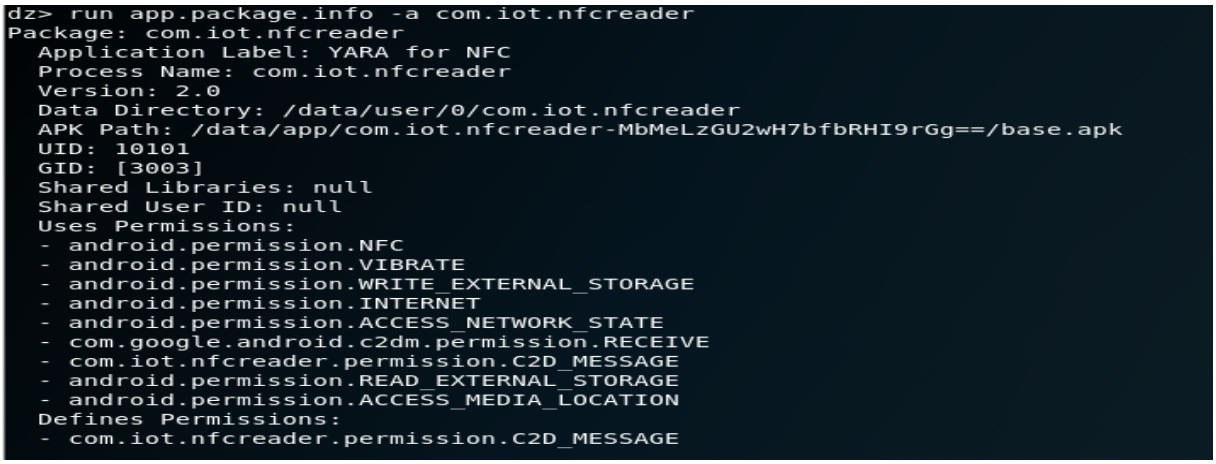

Figure 14: screenshot shows the drozer's permission listing command

Figure 15: screenshot shows the drozer's attack surface enumeration command