# Lightweight Security Middleware to Detect Malicious Content in NFC Tags or Smart Posters

Sufian Hameed*, Bilal Hameed‡, Syed Atyab Hussain*, Waqas Khalid*
*CS Department, National University of Computer and Emerging Sciences (FAST-NUCES), Pakistan
(sufian.hameed@nu.edu.pk, k102230@nu.edu.pk, m.waqas.khalid@hotmail.com)
‡IPVS - University of Stuttgart, Germany (bilal.hameed@ipvs.uni-stuttgart.de)

***Abstract*—NFC, similar to RFID, is a contactless identification technology that has gained widespread adoption in application domains such as contactless transactions, micropayments, identity management and mobile health IDs etc. However, the NFC standard itself does not provide built-in security features. This means that each and every developer would need to implement security features in his NFC application on his own. This in turn results in users being as secure from NFC-based security vulnerabilities as the developer chooses to implement. Clearly, this environment is a major impediment in the widespread adoption and deployment of NFC applications. In this paper, we propose the concept of a light weight security middleware that could implement the different security safeguards needed by a wide array of applications. The applications in turn can then implement the security features relevant to their scenarios. Furthermore, we implemented security safeguards to deal with malicious tag content in NFC applications. Our evaluations showed that the security middleware is both lightweight and has low functional latency.**

***Index Terms*—Near Field Communication; NFC Security; Malicious URL Filtering; Security Middleware**

## I. Introduction

Near Field Communication (NFC) is a standards-based, wireless communication paradigm that enables simple and safe two-way interactions between electronic devices over a short-range (a few centimeters). NFC enabled devices can establish communication with each other by a simple touch or by coming into proximity with each other. Current and envisioned applications for NFC include contactless transactions, micro payments with smartphones, data exchange, and simplified setups for more complex communications such as Bluetooth and Wi-Fi.

NFC devices can be used in arbitrary applications. In particular NFC enabled devices have huge potential in mobile payments [7], [12], retail systems [6], [19] and ticketing system deployed for public transport [27]. NFC also provides non-payment opportunities in advertising, consumer electronics, gaming, healthcare and wellness [15], [17], and social networks [11], [25].

Many of the key ingredients for global adoption of NFC are already in place. NFC-forum [2] aggregated statistics from major technology research and advisory firms like ABI Research and Gartner. According to these statistics, NFC enabled smartphone shipments are anticipated to increase by 481% from 2012 to 2015. Further by 2015, 50% of smart-phones will have NFC capability and NFC will be the most used solution for mobile payments, enabling worldwide transactions totaling about $151.7 billion. Based on these statistics it is evident that there is a consensus on the market growth of NFC devices which is very encouraging for NFC application developers.

When it comes to security, NFC technology has had a dubious history. Unlike RFID, the communication range of NFC is limited to a few centimeters; still NFC does not enhance any inherent security features or in other words NFC alone does not ensure secure communications. Due to this short coming, attackers can easily exploit the NFC technology. Eavesdropping practices can easily be observed in applications that use NFC technology [14]. With eavesdropping an attacker can listen into NFC transactions and access victim's credentials such as his financial and personal information and use it for malicious intent. Apart from eavesdropping, data corruption and manipulation is also becoming an increasingly common threat [18]. Lack of efficient security mechanisms will be, without doubt, the limiting factor in the wide spread adoption of NFC based applications.

Eavesdropping, however, is not the only security concern plaguing NFC applications. Presence of malicious content i.e. malicious code execution through command or malicious URLs in NFC tags is still an open concern. NFC tag authentication is a practical solution against tag content spoofing only for NFC based financial and ticketing services [22]. In this regard, S-SPAN [28] also proposed a framework for secure active-passive pairing between NFC tags and Android devices in a smart poster setting. S-SPAN used a very involving approach to eliminate tag spoofing and cloning. Their idea is to store no information other than a string of random bytes in the tag for identification. Whenever a user interacts with a smart poster using his smartphone, an authenticated query is generated to fetch the content of the tag from a web server. The users are required to be registered and logged-in at the time of interaction. This registration requirement will create serious usability issues in the NFC Tap-N-Go communication and further it would be very difficult to authenticate arbitrary tags dispersed with smart posters.

Keeping in view these security concerns, we propose a lightweight security middleware for NFC applications. In principle, the middleware would be capable of providing security solutions for problems such as eavesdropping, identity theft, malicious content etc. However, for the purpose of this paper, we would concentrate only on the implementation of safeguards against malicious tag content i.e. malicious URLs. The core idea in our implementation is to observe the content of the tag to detect malicious element at early stages before they get a chance to execute and compromise the security of the NFC smart device. The basic task of the middleware is to prevent the user from accessing the malicious or spoofed URLs while sharing data with NFC tags. These malicious URLs can compromise the security and enable drive-by-downloads leading to all sorts of exploits on a victim's smartphone. There are several antivirus applications that provide safe web browsing functionality; however, there are still a large number of smartphone users without any antivirus application installed on their devices. This calls for an efficient lightweight plug-and-play middleware application that can be natively installed on NFC-enabled phones to protect against malicious content (URLs) in NFC tags.

Our proposed middleware framework acts as an intermediate layer, between the data read and the execution of desired functionality, to verify the discrepancies in the NFC Data Exchange format (NDEF[1]). The framework is integrated with native NFC reader applications such that the detection of malicious element functionality is available to all NFC applications without rooting the mobile device. When a user will swipe his phone on a smart poster, the transmitted data will be passed to the middleware. At this stage the middleware will inspect the URL data according to the NDEF format and essentially apply three different techniques i.e. whitelisting, blacklisting and trust and reputation to detect malicious URLs. If the middleware detects malicious URL it will raise a suspicion flag. This will prompt the user to decide whether he wants to continue or abort. If no suspicious activity is detected, all the operations will be performed transparently.

We also evaluate the performance of URL filtering with different techniques deployed in the middleware. We demonstrate that the detection of malicious elements in URLs is efficient and it also scales with increasing size of white and black lists. The evaluation show that the memory footprint of the security middleware remained as low as 2.3 % (23.5MB) with white and black list size of 50K entries each. The CPU usage on the other hand remained under 4% when tested with HTC One X. We also demonstrate that the middleware has reasonably low latency with a worst case URL filtering latency being less than 1 second.

## II. Discussion on State of the Art

The recent advances in NFC technology and wide scale proliferation of NFC enabled smart devices have resulted in an increasing interest in the security aspect of NFC based applications. This section highlights various security attacks, threats, solutions and open problems put forward by the research community.

The authors in [10] gave a comprehensive analysis of various aspects of security whenever an NFC interface is used. The paper highlighted different threats, for instance eavesdropping, data corruption, man-in-the-middle attack, data insertion and data modification, which can compromise NFC interactions. The paper concludes with a very brief discussion on various possible solutions that can be used to circumvent these threats.

Collin Mulliner [18] was among the first few security researchers to analyze the security of NFC-enabled mobile phones. Mulliner developed tools for security testing of NFC-enabled mobile phone, based on which he was able to detect a number of previously unknown vulnerabilities. These vulnerabilities can easily be exploited for spoofing of tag content, an NFC-based worm, and even for denial-of-service attacks.

In [14] Renaud Lifchitz, a researcher from British Telecom (BT), developed a proof of concept Android and Desktop application to hack NFC-based credit cards. This work was presented in Hackito Ergo Sum in April 2012. Apart from the attacks discussed above, there is a thorough discussion on NFC attack surface [16] including attacks related to Google Wallet and secure element [21], [23], [26] along with couple of solutions to mitigate them [5], [13], [20].

[8] propose a mobile payment system for merchant micropayments based on the existing GSM and NFC infrastructure. The proposed m-payment system uses existing SIM authentication and identification capabilities and leverages GSM cryptographic primitives. This results in easy integration into existing GSM infrastructure and already deployed POS systems. The system provides reasonable security for low value micropayments.

SemTag [9] provides secure and quick access to medical and emergency data of patients stored in German electronic Health Card (eHC). SemTag makes the information of eHC available to care givers in emergency situations. This was previously not possible for unconscious patients, since the eHC card cannot be duplicated, customized or read without physical contact.

Despite all the security related solutions discussed above, presence of malicious content i.e. malicious code execution through command or malicious URLs in NFC tags is still an open concern. In this paper we have designed and developed an efficient technique to detect malicious element, malicious URLs in particular, at early stages before they get a chance to execute and compromise the security of the NFC smart devices.

## III. Lightweight Security Middleware for Malicious Content Detection

Security can be the Achilles' heels in the wide spread adoption of NFC based applications. If security and privacy is not considered right from the beginning of R&D, it results in sub-optimal security solutions [24]. A right approach would be to consider security and privacy in the earlier stages as it will enable developers to avoid risks from the very beginning. Retrofitting security always tends to be more challenging, costly and often results in sub-optimal solutions. If usability factors are important for any technology then it will be

[1]NDEF is a lightweight, binary message encapsulation format to exchange information between two NFC devices or an NFC device and a tag.

even more indispensable to consider security early in the development [10].

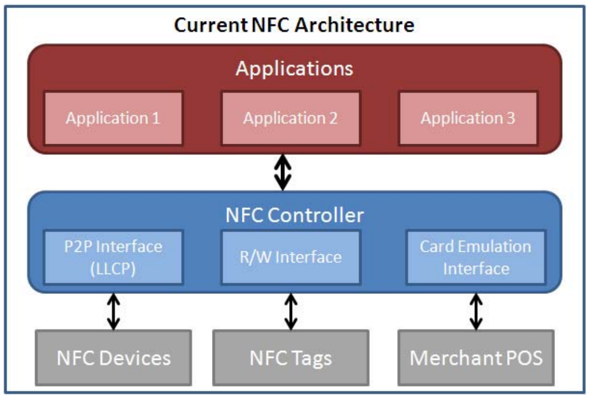

Fig. 1. Current NFC Application Architecture

NFC smart poster streamlines the user experience and simply takes the user to a website. The user does not have to waste time and type out the URL. However, this extremely beneficial functionality can also be easily abused. An attacker can very easily write a malicious URL on the NFC smart poster and deceive a victim into loading a malicious website. A part from inserting malicious URL an attacker can also abuse the NFC technology to secretly install malicious software or worm on the victim's phone. Current NFC reading applications are not capable of detecting presence of any malicious content or URL in the NFC-tags.

NFC tag authentication is a practical solution against a tag's content spoofing of NFC-based financial and ticketing services [22]. On the other hand it would be very difficult to authenticate arbitrary tags dispersed with smart posters. Unfortunately, current NFC specifications do not provide for built-in security features. Fig. 1 presents the current NFC communication architecture. Now in the absence of built-in security services, each and every application would have to come up with its own security features, which is pretty much a redundant work and is akin to re-inventing the wheel again and again. Furthermore, application developers could frequently forget on implementing some safeguards, thereby rendering their applications vulnerable to that kind of attack.

Keeping the risk of malicious tags in mind, we propose a lightweight security middleware that will observe the content of the tag to detect malicious element before it can be executed. The core objective of the security middleware, within the scope of this paper, is to prevent the user from accessing the malicious or spoofed URIs in NFC tags. Basically the middleware acts as an intermediate layer between the data read and the execution of desired functionality, to verify the discrepancies in the NDEF format.

### A. Middleware Component Design

We design the middleware application to run as an Android service component to perform the long running operations in

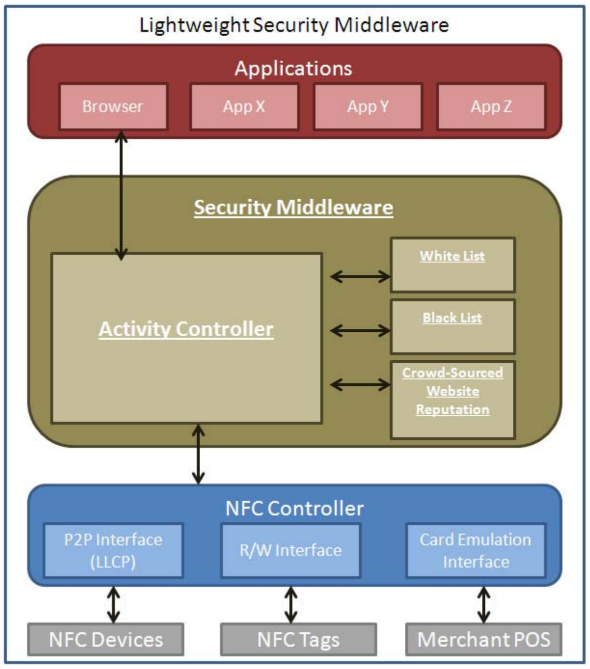

Fig. 2. NFC Application Architecture with Lightweight Security Middleware

the background. The middleware application is programmed to start as soon as the smartphone boots successfully and load all the necessary components in the starting. The service of the lightweight security middleware is invoked whenever there is any interaction between the phone and the NFC tags. Furthermore, the middleware only focuses on the URL NDEF record and guarantees safe interaction with smart poster tags. Hence the goal of the middleware is to provide the user with non-malicious or non-blacklisted content. In order to detect malicious URL tags, the middleware applies three different tests on the extracted tag content. Fig. 2 shows the basic communication architecture of NFC with the addition of security middleware. In the sections below, we will take a look at the functionality of each individual component.

*1) Activity Controller:* Activity controller, as the name indicates liaison all the functionalities of the security middleware with the NFC controllers on the lower layer and applications (browser) on the top. On tag read, if the NFC controller identifies the NDEF record to be a URL, the tag content will be pushed to activity controller of the middleware. At this point the activity controller will execute three different test using i) white listing, ii) black Listing and iii) crowd-sourced Internet website reputation ratings. Based on the response, the activity controller will either invoke the browser to transparently load the URL or prompt the user that the tag contains malicious content.

*2) White Listing:* After performing initial truncation on the URL, the activity controller performs white list verification on the URL. A white list acts as a register of

URLs that are approved to be legitimate. For our prototype implementation we have used Alexa [1] top 100,000 websites in the white list. These URLs are stored in simple text format. During tag interaction, if the read URL gets a match with the white list, the activity controller will transparently trigger the browser to open the URL. If there is no match, the URL is further forwarded to be tested against the blacklist.

*3) Black Listing:* During black list testing, the URL that fails the white list test is checked against the black list. The black list is a list of websites that are deemed to contain malicious content. Similar to the white list, the black list website URLs are stored in a simple text format. If the tag's URL finds a match with the black list, the security middleware application will show a prompt (alert) message to the user. The user then has the option to navigate the URL anyways or cancel his request and walks away from the malicious website. If the user decides to visit the URL in spite of the alert, the URL is removed from the black list and is added to the white list. If the tag's URL fails to match the black list entries, the activity controller will further pass the URL for the next test to get a crowd-sourced reputation rating on the URL.

For our prototype implementation we have used a URL blacklist service [3] that provides free download of 166,000 plus up-to-date blacklisted URLs.

*4) Crowd-sourced Internet Website Reputation Ratings:* If the URL is not available in both the white and the black list, the activity monitor will use the services of crowd-sourced Internet website reputation ratings as a last resort. In our solution we have used Web Of Trust (WOT) [4], which is a free website reputation, rating and review tool that aggregates crowd-sourced ratings and reviews for websites from over 115 million internet users. When the activity monitor passes a URL to the public API of WOT, the WOT reputation system returns the reputation ratings computed based on the user feedback and information from third-party sources. These ratings are computed for each URL and the WOT system computes two values: a reputation estimate and the confidence in the reputation.

- The reputation $R$, is the aggregate trust for any URL. The higher value of $R$ indicates higher trust of the community users for that particular URL. The definitions for the reputation values are $>=$ 80 is Excellent, $>=$ 60 is Good, $>=$ 40 is Unsatisfactory, $>=$ 20 is Poor and $>=$ 0 is very poor.
- The confidence $C$, is an estimated indicator of reliability for the reputation $R$ of a URL. Again, the higher the value of $C$, the more the system considers $R$ to be reliable.

In our prototype we have set a threshold of 60 for both $R$and $C$. This means that if the values of any of $R$ and $C$ are less than 60, the activity controller will alert the user, else the URL will be loaded in the browser application. Kindly refer to [4] for further details on the different matrices WOT uses to compute the reputation ratings.

In order to use the public API of WOT, we need an API key. For that you need to create a WOT account and request an access to API key. The following example shows how we can request reputation ratings for http://rfidsec2014.cis.uab.edu/ from WOT API.

```
http://api.mywot.com/0.4/public_link_json2
       ?hosts=rfidsec2014.cis.uab.edu/
       &callback=process&key=<your API key>
```

## B. Prototype Environment and Permission Settings

We have used Android platform for the development of the NFC security middleware. For the transparent operation of the middleware application, the following permissions and features will be added to its manifest file.

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

This will take permission from the user to read and write using NFC. This permission is essential as the middleware is intended to be the first point of access to the content fetched from the tags. By defining this permission in the manifest file the middleware can intercept all the NFC communication streams.

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

This will acquire the permission from the user to access Wi-Fi state i.e. whether Wi-Fi is available or not. This state information is needed since our application uses WOT API which requires Wi-Fi connection

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

This will take permission from the user to use internet in the application.

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

This will take permission from the user to use external storage to write URLs to the white and black lists.

```
<uses-feature android:name="android
.hardware.nfc" android:required="true"/>
```

This feature request will make sure that the application will only be available for the users who have NFC enabled smart phones when downloading is requested via the Google play store.

## IV. Evaluations

This section presents the performance evaluation of the security middleware to detect malicious URLs in NFC tags or smart posters. Our evaluations mainly focused on the extra delays incur with the addition of the middleware in NFC application architecture. It is very important that the effect of middleware is negligible in terms of latency; else the usability aspect of this tap-and-go technology will be effected. We also run some experiments on the CPU and memory usage to show that the middleware has a reasonable CPU and memory footprints.

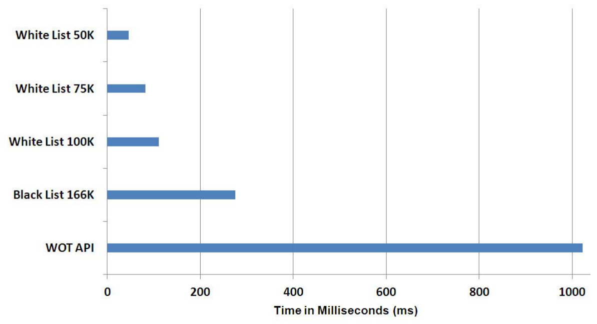

Fig. 3. Latency with white list, black list and WOT API with HTC Amaze 4G

## A. Evaluation Settings and Results

Evaluating the interaction latency for a NFC tag and a smartphone requires a lot of patience. The NFC tag is manually read and the latency is then noted by inserting time stamps at different function points. For evaluating the effect of whitelisting we use three different sizes for the white lists i.e. white list with 50,000, 75,000 and 100,000 entries and we place the URL written on the tag at the bottom of the list. Similar settings are used for Blacklisting, except we do not vary the size of the blacklist and keep it constant at 166,000, just the way we downloaded it from [3]. Finally we also evaluated the time it takes to analyze a URL by using crowd-sourced reputation ratings using WOT tool. For all the three white list samples, black list and WOT invocations we run 15 NFC tag interactions each (reading of tags with URL) and recorded the time taken for a URL to pass through each step of the application.

We used two different smartphones with different configurations with a 2 Mbps Internet connection. The key attributes of the smartphones used are as follows:

- HTC Amaze 4G: Android (2.3.3) Gingerbread with 1 GB RAM and a 1.5 GHz Dual Core CPU.
- HTC One X: Android (4.1.1) Jelly Bean with 1 GB RAM and a 1.5 GHz Quad Core CPU.

Fig. 3 and Fig. 4 show the average results of different experiments spread over the NFC tag interactions with HTC Amaze 4G and HTC One X respectively. It takes at an average of 46.3 ms, 82.1 ms and 110.9 ms to validate a URL read from an NFC tag against the white lists on the HTC Amaze. On the other hand, with HTC One X, it takes at an average of 35.4 ms, 66.5 ms and 100.3 ms to validate a URL read from an NFC tag against the white list.

The blacklist validation takes a bit longer and requires on average 275.7 ms with HTC Amaze and 205.9 ms with HTC One X. This is obvious since the blacklist validation also

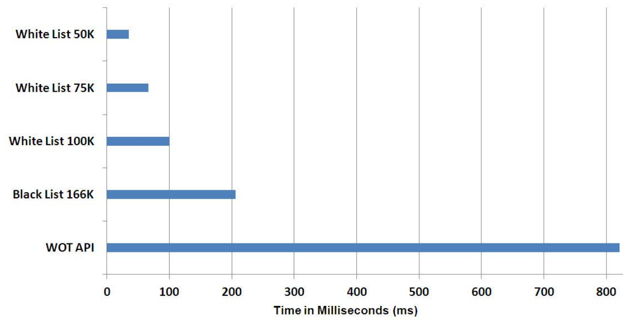

Fig. 4. Latency with white list, black list and WOT API with HTC One X

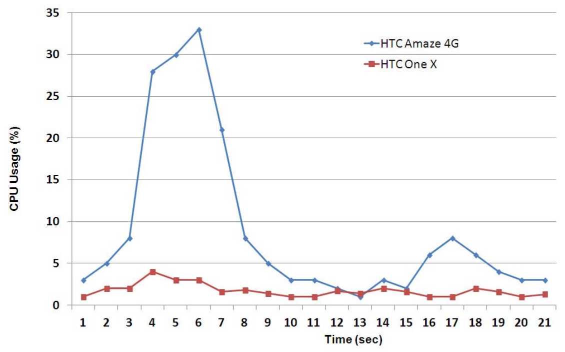

Fig. 5. The CPU usage during NFC tag interactions

includes the white list failure. The URL analysis that involves the WOT reputation ratings to judge the legitimacy of any URL requires on average 1021 ms with HTC Amaze and 821 ms with HTC One X. The latency in this case is a bit higher since it involves both white list and black list failure and WOT API invocation and response. These results show that the maximum time that a user will require to interact with the new NFC application architecture (with our security middleware) will be 1 second at max. This will be highly efficient since NFC applications are not intended to make their user wait for the response and act accordingly.

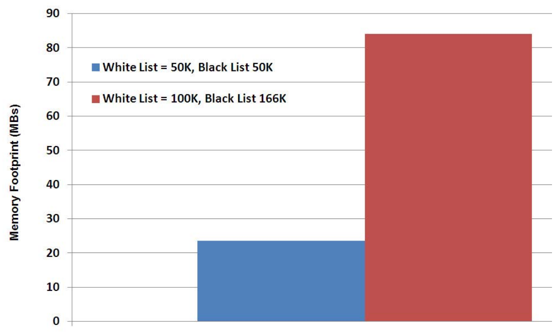

Fig. 6. Memory footprint of the Middleware with large white and black lists

We analyze the effect of the security middleware on CPU over a short time span, while reading and interacting with the

NFC tags or smart poster. The CPU usage is almost identical during and in between the interactions on the HTC One X and the overall usage remained between 1-4% (see fig. 5). The spikes on the low computational CPU of HTC Amaze are assumed to be caused by system processes independent of the tag interactions. To interpret these results it can be stated that the security middleware does not have a significant impact on the CPU load.

As discussed before, we implemented the middleware application as an Android service component to perform long running operations in the background. The middleware application is programmed to start as soon as the smartphone boots successfully and load white and black lists with huge entries in the starting. Whenever an event (tag read) is detected the middleware application retrieves pre-loaded entries quickly and will not load the white and black lists for each poster interaction. Fig. 6 show the results (approximately same for both smartphones) related to memory footprints of the middleware application. According to the results, the memory footprint of the middleware application running as an Android service is 23.4 MB i.e. 2.3 % when white and black lists are maintaining 50K entries each. When we increase the size of white and black lists to 100K and 166K respectively, the memory footprint also increased to approximately 83.5 MB i.e. 8.3 %. Since the size of list is directly related to the memory footprint, it would be efficient to keep the size of white list in couple of hundreds and few thousands for black list. Failure of URL detection in the lists will invoke the WOT API with maximum latency of just 1 second.

## V. Conclusion and Future Work

In this paper, we presented the concept of a light weight security middleware that would implement security safeguards against most of the well known security vulnerabilities in NFC-based applications. For the purpose of this paper, the middleware implemented security measures against malicious tag content behavior in NFC-based smart posters. Our evaluations showed that the security middleware has reasonable memory and CPU footprints. In addition to this, the middleware has reasonably low latency with a worst case latency being less than 1 second. This would ensure that users would not face a high degree of usability degradation even with the middleware implemented natively on NFC devices.

In future, we would like to extend the middleware with security safeguards for most of the well known security vulnerabilities such as identity theft, man-in-the middle attack, tag spoofing etc. The idea is to provide NFC application developers with a one stop shop for all of their security scenarios.

## References

[1] Alexa. www.alexa.com.

[2] Nfc forum. www.nfc-forum.org.

[3] Url black list. www.urlblacklist.com.

[4] Web of trust (wot). www.mywot.com/wiki/API.

[5] V. Alimi. An ontology-based framework to model a global platform secure element. In *4th International Workshop on NFC*, March 2012.

[6] G. Broll, H. Palleis, H. Richter, and A. Wiethoff. Exploring multimodal feedback for an nfc-based mobile shopping assistant. In *5th International Workshop on NFC*, Feb 2013.

[7] U.B. Ceipidor, C.M. Medaglia, A. Opromolla, V. Volpi, A. Moroni, and S. Sposato. A survey about user experience improvement in mobile proximity payment. In *4th International Workshop on NFC*, March 2012.

[8] W. Chen, G. P. Hancke, K. E. Mayes, Y. Lien, and J.-H. Chiu. Nfc mobile transactions and authentication based on gsm network. In *Second International Workshop on NFC*, 2010.

[9] Sebastian Dunnebeil, Felix Kobler, Philip Koene, Jan Marco Leimeister, and Helmut Krcmar. Encrypted nfc emergency tags based on the german telematics infrastructure. In *Proceedings of the 2011 Third International Workshop on Near Field Communication*, NFC '11, pages 50–55, Washington, DC, USA, 2011. IEEE Computer Society.

[10] Haselsteiner E and K. Breitfu. Security in near field communication (nfc). In *Proceedings of the International workshop on RFIDSecurity*, RFIDSec '06, 2006.

[11] A. Fressancourt, C. Herault, and E. Ptak. Nfcsocial: Social networking in mobility through ims and nfc. In *Near Field Communication, 2009. NFC '09. First International Workshop on*, pages 24–29, Feb 2009.

[12] Omkar Ghag and Saket Hegde. Article: A comprehensive study of google wallet as an nfc application. *International Journal of Computer Applications*, 58(16):37–42, November 2012. Published by Foundation of Computer Science, New York, USA.

[13] M. Hutter and R. Toegl. A trusted platform module for near field communication. In *Proceedings of the Fifth International Conference on Systems and Networks Communications (ICSNC),*, pages 136–141, Aug 2010.

[14] Renaud Lifchitz. Hacking the nfc credit cards for fun and debit. In the Hackito Ergo Summit, 2012.

[15] A. Marcus, G. Davidzon, D. Law, N. Verma, R. Fletcher, A. Khan, and L. Sarmenta. Using nfc-enabled mobile phones for public health in developing countries. In *Near Field Communication, 2009. NFC '09. First International Workshop on*, pages 30–35, Feb 2009.

[16] Charlie Miller. Exploring the nfc attack surface. In the Black Hat Convention, 2012.

[17] J. Morak, D. Hayn, P. Kastner, M. Drobics, and G. Schreier. Near field communication technology as the key for data acquisition in clinical research. In *Near Field Communication, 2009. NFC '09. First International Workshop on*, pages 15–19, Feb 2009.

[18] Collin Mulliner. Vulnerability analysis and attacks on nfc-enabled mobile phones. In *ARES*, pages 695–700, 2009.

[19] Denise Paradowski and Antonio Kruger. Modularization of mobile shopping assistance systems. In *Near Field Communication (NFC), 2013 5th International Workshop on*, pages 1–6, Feb 2013.

[20] Marie Reveilhac and Marc Pasquet. Promising secure element alternatives for nfc technology. In *Proceedings of the 2009 First International Workshop on Near Field Communication*, NFC '09, pages 75–80, Washington, DC, USA, 2009. IEEE Computer Society.

[21] M. Roland, J. Langer, and J. Scharinger. Practical attack scenarios on secure element-enabled mobile devices. In *4th International Workshop on NFC*, March 2012.

[22] Michael Roland and Josef Langer. Digital signature records for the nfc data exchange format. In *Second International Workshop on NFC*, 2010.

[23] Michael Roland, Josef Langer, and Josef Scharinger. Applying relay attacks to google wallet. In *Proceedings of 5th International Workshop on Near Field Communication (NFC)*, pages 1–6, Feb 2013.

[24] Peter Schoo and Massimo Paolucci. Do you talk to each poster? security and privacy for interactions with web service by means of contact free tag readings. In *First International Workshop on NFC*, 2009.

[25] E. Siira and V. Tormanen. The impact of nfc on multimodal social media application. In *Second International Workshop on NFC*, April 2010.

[26] Keith Wagstaff. Google wallet hack shows nfc payments still aren't secure. http://techland.time.com, 2012.

[27] R. Widmann, S. Grunberger, B. Stadlmann, and J. Langer. System integration of nfc ticketing into an existing public transport infrastructure. In *4th International Workshop on NFC*, March 2012.

[28] J. Wu, Lin Qi, R.S.S. Kumar, N. Kumar, and P. Tague. S-span: Secure smart posters in android using nfc. In *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–3, June 2012.