Practical NFC Peer-to-Peer Relay Attack using Mobile Phones

Lishoy Francis, Gerhard Hancke, Keith Mayes, and Konstantinos Markantonakis

Information Security Group, Smart Card Centre,
Royal Holloway University of London,
Egham Hill, TW20 0EX, Surrey, United Kingdom.
(L.Francis;Gerhard.Hancke;Keith.Mayes;K.Markantonakis)@rhul.ac.uk
http://www.scc.rhul.ac.uk/

Abstract. NFC is a standardised technology providing short-range RFID communication channels for mobile devices. Peer-to-peer applications for mobile devices are receiving increased interest and in some cases these services are relying on NFC communication. It has been suggested that NFC systems are particularly vulnerable to relay attacks, and that the attacker's proxy devices could even be implemented using off-the-shelf NFC-enabled devices. This paper describes how a relay attack can be implemented against systems using legitimate peer-to-peer NFC communication by developing and installing suitable MIDlets on the attacker's own NFC-enabled mobile phones. The attack does not need to access secure program memory nor use any code signing, and can use publicly available APIs. We go on to discuss how relay attack countermeasures using device location could be used in the mobile environment. These countermeasures could also be applied to prevent relay attacks on contactless applications using 'passive' NFC on mobile phones.

1 Introduction

Near Field Communication (NFC) [1] is intended as a short-range standardised technology for providing contactless communications for mobile devices. NFC is intended to be an intuitive method of establishing ad-hoc connections, simply requiring that two NFC-enabled devices are brought in close physical proximity to each other. NFC also allows for devices to interact with existing contactless/RFID (Radio Frequency Identification) systems. In 'passive' communication mode NFC allows devices to emulate passive contactless smart cards, while 'active' mode allows for devices to act as contactless smart card readers or to communicate with each other. Although the use of NFC-enabled devices in contactless systems has received much publicity, the use of NFC to support peer-to-peer services is less well covered.

One of the earliest specified uses of active NFC was to pair Bluetooth devices by facilitating the exchange of information needed to setup the Bluetooth

communication channel [2]. NFC can also be used for sharing data and content between mobile devices, such as digital business cards and social networking details, although the data rates are currently not best suited to high-bandwidth transfers. Mobile payments are becoming increasingly popular and there are a variety of schemes using a range of data bearers. NFC is seen as an ideal technology in this area with its ability to interact with existing contactless systems and facilitate peer-to-peer transactions between mobile phones [3].

In short range communication systems, it is usually assumed that the devices are actually in close physical proximity when successfully communicating. However, in a relay attack the communication between two devices are relayed over an extended distance by placing a proxy device within communication range of each legitimate participant and then forwarding the communication using another communication channel. The two legitimate participants receive valid transmissions from each other and therefore assume that they are in close physical proximity. In some systems, especially in smart token environment, this could lead to serious security vulnerabilities [4]. The risk posed to near-field channels by relay attacks and the possibility of using NFC-enabled mobile phones as a relay attack platform have been discussed [5] [6], but a practical relay attack using this platform has not been demonstrated.

In this paper, we describe how a relay attack against peer-to-peer NFC system could be practically implemented. The novelty of this attack is that it also uses available NFC-enabled mobile phones as attack platforms, providing the attacker with off-the-shelf proxy device. The attacker's mobile phones are of acceptable (non-suspicious) form factor, unlike custom built emulators used in other relay attacks. The attack functionality can also be implemented using only software via publicly available APIs in a standard MIDlet (Mobile Information Device Profile or MIDP application) using JSR 118 API [7]. The resources and technical skill required of the attacker are therefore greatly reduced. In Section 2 we provide a brief introduction to NFC communication and relay attacks. We describe the attack implementation in Section 3 and discuss current countermeasures that could be used to mitigate relay attacks in Section 4.

2 Background

NFC technology allows for the integration of contactless technology into active devices, such as mobile phones. NFC operates within the 13.56 MHz Radio Frequency (RF) band and has an operating distance up to 10cm. A NFC-enabled device can act as both a "contactless card" and a "contactless reader". NFC-enabled devices, as specified in ISO-18092/ECMA-340 [1] and ISO-21481/ECMA-352 [8], are compatible with existing contactless systems adhering to ISO 14443 [9], ISO 15693 [10] and FeliCa [11]. The NFC standards also define a communication mode for peer-to-peer (P2P) or 'active' communication, with the purpose of facilitating communication between two NFC-enabled devices. In 'active' NFC, the participants communicate in a "client-server" model. The device that starts the data exchange is known as the Initiator and the recipient is known as the Target. In 'active' mode, the Initiator and Target uses their own generated RF

field to communicate with each other. First the Initiator transmits an RF carrier, which it uses to send data to the Target. Once an acknowledgment for the data sent has been received from Target (by modulating the existing field), the Initiator switches the carrier off. The original Target then reprises the role of Initiator, switching on its carrier, and transmits a response to the original Initiator. For the purposes of reader's clarity, we call the NFC enabled mobile phone configured as the Initiator to be in "writing" mode and the phone configured as the Target to be in "reading" mode.

On NFC-enabled mobile phones the Secure Element (SE) provides the security means to establish trust between service provider and the device. The SE also provides a secure environment for hosting sensitive applications and storing cryptographic keys. Currently there are three main architectures for NFC. The first involves an SE that exists as an 'independent' embedded hardware module, i.e. a stand-alone IC (Integrated Chip) is built into the phone. In the second option, the SE is implemented within the UICC (Universal Integrated Circuit Card) [12]. Of the existing Subscriber Identity Application (SIA) modules such as the Subscriber Identity Module (SIM) [13], Universal Subscriber Identity Module ((U)SIM)[14] and Removable User Identity Module ((R)UIM) [15]. The third option implements the SE on a removable memory component such as a Secure Multi-Media Card (Secure MMC) or Secure Digital card (Secure SD) [16]. The discussion comparing the advantages and disadvantages of the above mentioned architectures is beyond the scope of this paper. It is important to note that the NFC standards does not specify any security services apart from the Signature Record Type Definition [17], leaving the security design to the application developer. The Signature RTD specifies how data is to be signed to ensure data integrity and provide data authentication. This standard is currently being reviewed by the NFC Forum [18].

The handsets that were used in our practical experiments implemented independent SEs, which supported Java Card 2.2.1 [19] (Java Card Open Platform [20]), Global Platform 2.1.1 [21] and Mifare Classic [22] emulation. To implement 'passive' emulation of a contactless token an application must be installed in the secure program memory of the SE. Currently, it is possible to unlock the SE in 'independent' architectures and to install custom applications [23]. However, this would be controlled when the UICC is used as the SE, as access to the UICC is strictly managed by the mobile operator [24]. When implementing 'active' NFC communication the functionality can be entirely controlled via a MIDlet installed in the non-secured application memory of the mobile phone. A developer using the mobile phone as a platform for 'active' communication does therefore not need to gain access to any secure parts of the NFC architecture. The MIDlet can implement the 'active' NFC functionality using standard functions available within the extensions of the public JSR 257 Contactless Communication API [25].

2.1 Relay Attack Theory

The Grand Master Chess problem as discussed in [26] provides a classic example of relay attack. In this scenario a player who does not know the rules of Chess can

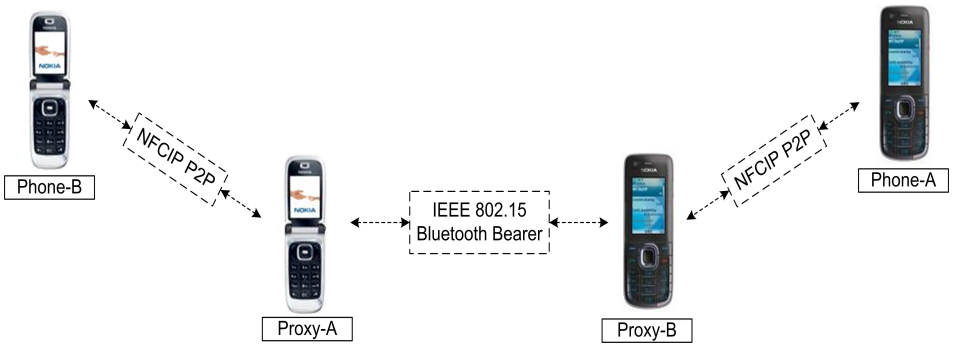


Fig. 1. P2P Relay Setup using Bluetooth.

simultaneously play against two grand masters. The player starts a postal game of Chess with each grand master and subsequently forwards the moves originating from one grand master to the other. Although each grand master thinks that they are engaging the player they are essentially playing against each other. The application of this scenario to security protocols was first discussed in [27] as 'mafia fraud'. Subsequently this attack has also been referred to as a 'wormhole attack' [28] or as a 'relay attack' [29]. Using this attack an attacker is able to bypass security protocols by only relaying challenges and responses between two legitimate entities. As the attacker is always in the position to provide the correct reply, which he/she learned from the other party by forwarding the original message and recording the response, the security protocol is executed successfully and both parties will consider the attacker to be a legitimate participant in the protocol. In this scenario the attacker never needs to know any detail of the information he/she relays, i.e. he/she does not need to know the structure of the protocol, the algorithms used, the plain text data sent or any secret key material. The attacker must only be in a position where he/she can continue to relay the communication for the entire duration of the protocol. Earlier practical examples of relay attack in the contactless environment using custom-built hardware or using NFC-enabled contactless readers can be found in [30], [31] and [32].

3 Relay Implementation

We implemented the relay attack against two NFC enabled mobile phones operating in peer-to-peer mode and participating in a legitimate transaction. As illustrated in the Figure 1, Phone-A intends to interact with Phone-B to perform a legitimate peer-to-peer transaction. The attacker introduces two additional mobile phones into the transaction setup, namely Proxy-A and Proxy-B, to relay the communications between Phone-A and Phone-B. In our proof-of-concept attack experiment, we practically implemented the relay attack using four NFC enabled mobile phones, as shown in Figure 2.

Fig. 2. Devices used in the relay attack – Device B (Nokia 6131 NFC), Proxy A (Nokia 6131 NFC), Proxy B (Nokia 6212 Classic NFC) and Device A (Nokia 6212 Classic NFC).

3.1 Phone-A and Proxy-B

The role of Proxy-B, as the name suggests, is to represent Phone-B and to relay communications to and from Phone-A. To realise both Phone-A and Proxy-B we used two Nokia 6212 Classic NFC mobile phones which are based on Symbian S40 5^{th} Edition FP1 platform. On Phone-A, a MIDlet (MIDP 2.1 application [7]) was implemented (3 kilobytes in size) that utilised the JSR 257 extensions API to realise NFC peer-to-peer communications. Phone-A is designed to switch between "reading" and "writing" modes as required.

On Proxy-B, a MIDlet (MIDP 2.1 application) implemented (14 kilobytes in size) the JSR 257 extensions for NFC peer-to-peer and JSR 82 API [33] for IEEE 802.15 (Bluetooth) communications. By default, Proxy-B was configured in "reading" mode and also supports "writing" mode. The NFC platform of Phone-A and Proxy-B supported the active peer-to-peer mode of operations for both Target and Initiator. Hence these devices performed "reading" and "writing" in active mode.

3.2 Phone-B and Proxy-A

Phone-B and Proxy-A were realised on two Nokia 6131 NFC mobile phones, based on Symbian S40 3^{rd} Edition FP1 platform. Proxy-A represented Phone-A in the transactions and relayed messages with Proxy-B. Similar to Phone-A, on Phone-B a MIDlet (MIDP 2.0 application [7]) was implemented (3 kilobytes in size) that utilised JSR 257 extensions API to realise NFC peer-to-peer communications. Phone-B is designed to switch between "reading" and "writing" modes as required.

On Proxy-A, a MIDlet (MIDP 2.0 application) implemented (14 kilobytes in size) the JSR 257 extensions API [25] for NFC peer-to-peer and JSR 82 API [33]

for IEEE 802.15 (Bluetooth) communications. By default, Proxy-A was configured in "writing" mode and also supported "reading" mode. The NFC platform of Phone-B and Proxy-A supported active peer-to-peer mode of operation for Initiator, but only passive for Target. Hence these devices performed "reading" in passive mode and "writing" in active mode. Using JSR 257 extensions API, the connection open and receiving data in "reading" mode were made as follows,

```
private static final String TARGET_URL
= "nfc:rf;type=nfcip;mode=target";
NFCIPConnection conn = (NFCIPConnection) Connector.open(TARGET_URL);
byte[] data = conn.receive();
byte[] ack = {(byte) 0xFF,(byte) 0xFF};
conn.send(ack);
```

Similarly, the connection open and sending data in "writing" mode were made as follows,

```
private static final String INITIATOR_URL
= "nfc:rf;type=nfcip;mode=initiator";
NFCIPConnection conn = (NFCIPConnection) Connector.open(INITIATOR_URL);
byte[] cmd = {(byte) 0x9A,(byte) 0xED};
conn.send(cmd);
conn.receive();
```

3.3 Relay Bearer

Proxy-A and Proxy-B established the relay channel using Bluetooth, where Proxy-B acted as the "server" and Proxy-A act as the "client". Bluetooth is a short range radio technology developed by Bluetooth Special Interest Group (SIG) and utilises unlicensed radio in the frequency band of 2.45GHz. The supported data speed is approximately 720kps. Bluetooth communication range from 10 metres to 100 metres. The MIDlets implemented the Bluetooth communication on Proxy-A and Proxy-B using JSR 82 API. We used L2CAP (Logical Link Control and Adaption Protocol) available within the host stack of the Bluetooth protocol. L2CAP is layered over the Baseband Protocol and operates at the data link layer in the OSI (Open System Interconnection) Reference Model. It supports data packets of up to 64 kilobytes in length with 672 bytes as the default MTU (Maximum Transmission Unit) and 48 bytes as the minimum mandatory MTU. For creating an L2CAP server connection, a btl2cap scheme (url), a 16 byte service UUID (Universally Unique Identifier), a friendly name (device name) and other parameters were provided as follows,

```
String service_UUID = "00000000000010008000006057028A06";
String url = "btl2cap://localhost:" + service_UUID + ";ReceiveMTU
=672;TransmitMTU=672;name=" + deviceName;
L2CAPConnectionNotifier notifier =
(L2CAPConnectionNotifier) Connector.open(url);
conn = notifier.acceptAndOpen();
```

For creating an L2CAP client connection, a btl2cap scheme, unique Bluetooth MAC address (6 bytes) of the server device, protocol service multiplexer (port) for the remote device and other parameters were provided as follows,

```
String url = "btl2cap://00226567009C:6001;authenticate=false;
encrypt=false;master=false;ReceiveMTU=672;TransmitMTU=672";
conn = (L2CAPConnection) Connector.open(url);
```

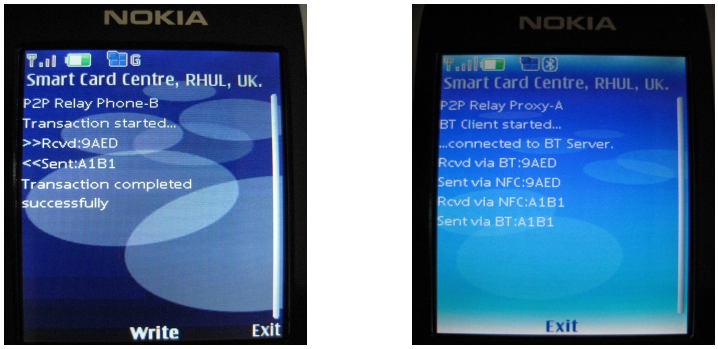
The MIDlets used the DiscoveryAgent class to perform Bluetooth device and service discovery. The DiscoveryListener interface was implemented to handle the notifications of devices and services. When devices and services are discovered, the DiscoveryAgent notifies the MIDlet by invoking callback methods such as deviceDiscovered() and serviceDiscovered(). The methods such as retreivedDevices() and searchServices() caches the previously discovered devices and services, in order to reduce the time needed for discovery. The client MIDlet on Proxy-B listened for the registered service and when available, connected to the server. The sending and receiving of data were achieved by using an L2CAPConnection.

We note that the MIDlets implementing the Bluetooth bearer did not require any code signing [34] and executed in the untrusted 3^{rd} party security domain. More information on Java security domains can be found in [35]. Currently the attacker just needs to enable the application access privilege [36] [37] for connectivity to be set as "always allowed" and enable Bluetooth to be used on the mobile phone. The device specific API access rights can be found in [38] for a Nokia 6131 and in [39] for a Nokia 6212.

Alternatively, Bluetooth can be replaced with SMS (Short Messaging Service) or mobile Internet [using intermediate server(s) via HTTP (Hyper Text Transfer Protocol) over GPRS/E–GPRS (Enhanced General Packet Radio Service)] as the data bearer in order to increase the relay range. These options were only briefly considered, but we found that these bearers introduced additional overheads. In SMS, the user interaction (key press to confirm the message sending) was required even when the MIDlet was signed in the trusted 3^{rd} party security domain. Additionally, SMS is considered to be not so reliable due to its inherent nature such as low bandwidth and variable latency. The additional key presses would also introduce suspicion on the attacker by the victim. Even though, the mobile Internet bearer does not have user interaction overhead similar to SMS, it does need signed code operating within trusted 3^{rd} party security domain.

3.4 Relay Experiment

To demonstrate a proof-of-concept attack, against peer-to-peer NFC transactions using enabled mobile phones, we implemented a simple application in a controlled laboratory environment. Phone-A would act as the Initiator and 'write' a 2-byte message to Phone-B, who would then become the Initiator and answer by "writing" a different 2-byte message back to Phone-A. Our goal was simply to demonstrate that it is possible to relay this exchange or transaction with the mobile-based proxy platforms.



(a) Device B (b) Proxy A

(c) Proxy B (d) Device A

Fig. 3. Relay attack data flow on each device involved.

To start the relay experiment, Proxy-A and Proxy-B negotiates and establishes the Bluetooth channel. In order to simplify the experiment, a 2-byte command message was input by the user in Phone-A configured in "writing" mode. The Phone-A exchanged the command with Proxy-B which then relayed to Proxy-A over the Bluetooth data bearer. Proxy-A being in "writing" mode transfered the payload onto Phone-B. The response for the command message was transferred by Phone-B to Proxy-A in "writing" mode and then relayed over to Proxy-B. The Proxy-B switched to "writing" mode and transfered the response message to Phone-A completing the relay process. In effect, Phone-A was made to believe that the response message to the sent command message was originating from Proxy-B. Similarly, Phone-B was made to believe that command message was originating from Proxy-A, which was actually originating from Phone-A via Proxy-B. Therefore it is reasonable to conclude that, in the real world the security of applications using NFC peer-to-peer communication can be potentially be bypassed using the method presented. The data flow of the relay, as processed by each device, is shown in Figure 3.

4 Using Location Information as Relay Attack Countermeasure

Protecting a system against a relay attack is difficult because this attack circumvents conventional application layer cryptography. Additional security measures are therefore needed to supplement existing authentication or encryption mechanisms. Several countermeasures to relay, or wormhole, attacks have been proposed. These countermeasures are intended for wireless sensor network and smart token environments. Although there are differences between these environments and mobile peer-to-peer services, there are aspects of these countermeasures that could be applied. This section provides a short overview of methods proposed for detecting relay attacks and we refer to an entity verifying the proximity of another entity as the "verifier" and the entity proving its proximity as the "prover".

One way of detecting relay attacks is to monitor any additional delay in the propagation time, which could be caused by the attacker when forwarding the data over a longer distance. Simply placing a time-out constraint on the round-trip time of a challenge-response exchange is not practical as the processing time of the prover can vary and a small error in the expected processing time could have a large influence on the round-trip time [4]. Distance-bounding protocols are designed to enable the verifier to accurately determine the round-trip time of a chosen cryptographic challenge-response [40], thereby eliminating the variability in the time taken to calculate the response. The security of distance-bounding protocols, however, are dependent on the underlying communication channel [41] and conventional channels similar to those used in NFC have been shown to be unsuitable for distance bounding [42]. Furthermore, more computationally advanced devices such as mobile phones can outperform legacy tokens and therefore circumvent systems implementing timing-based countermeasures that allow for processing time of legacy tokens.

The verifier could authenticate the prover based on physical characteristics of the communication channel. Using RF 'fingerprints' has been proposed as a method for source authentication in sensor networks [43] and for smart tokens [44]. An attacker would not be able to relay the communication as his proxy will not have the same RF fingerprint as the prover. This method requires that each device is 'fingerprinted', which is likely to be difficult in practice, especially for a large number of legitimate devices of varying ages (different versions of hardware platform) that are owned, managed or manufactured by multiple entities.

Relay attack could also be mitigated by asking the user to perform additional verification of the transaction. The user could be shown additional details of the underlying transaction by a trusted component to check that the transaction was executed faithfully [45], although this would place a significant responsibility on the user, reduce transaction throughput, and require that he/she has sufficient knowledge of how the transaction should be concluded. Another approach involving the user is multi-channel communication, where the user also verifies additional audio or visual channels [46]. This complicates the relay process, and the attacker must relay multiple channels, some of which might require high

bandwidth. This approach is promising although it might increase the transaction time and reduce simplicity of operation, which are aspects of NFC that are attractive to both users and service providers.

A popular approach in wireless sensor networks is to verify proximity by comparing the location of the nodes [47]. A network-wide localization algorithm is used to determine the relative or absolute location of the nodes. Nodes participating in relay attacks will cause anomalies in the network topology and the attempted attack will be detected. This method usually requires multiple nodes to collaborate in the localization process and constructing a topology, and in the mobile environment a transaction only involved two entities. Location services are, however, becoming prominent in mobile environment providing a basis for implementing a countermeasure that is practical, effective and remains transparent to the end user.

4.1 Integrating Location into NFC Transactions

There are a number of robust Location Based Services (LBS) already deployed in the mobile environment. These services are based on the fact that the location of the handset can be accurately determined, either by the network operator, a third-party or the handset itself. It is therefore feasible that location information could be incorporated into peer-to-peer transactions in order to provide relay-resistant communication. This section briefly discusses two methods for determining handset location that are currently used in LBS.

A simple, and widespread, method of retrieving and handling location information starts with obtaining the information of the cell broadcast tower (or sector) identifier or the Cell-ID, and attaching it with other parameters in the network broadcast such as Mobile Country Code (MCC), Mobile Network Code (MNC) and Location Area Code (LAC). For instance, one can compute a Location Code (LC) as follows,

```
LC=‘‘23415431824422847’’, where MCC=234 | MNC=15 | LAC=43182 | Cell-ID=4422847
```

This information forms the basis for a simple/crude location of the user device, i.e. calculating the position based on the above mentioned location parameters; primarily the tower (or sector) identifier or Cell-ID which the device last accessed. This is applicable to most traditional handsets (e.g. GSM/UMTS). The advanced information obtained from the device can be shown on commercial mapping services, e.g. Google Maps [48]. An important point to note is that the accuracy of this form of deriving the location of a mobile device is very limited and dependent on the radius of the cell coverage, which can range from tens of metres to tens of kilometres. Global Positioning System (GPS) technology is becoming available in an increasing number of mobile handsets and it is a more accurate method of deriving the location information from the mobile phone itself. Providing the handset can detect the satellite signals (which is not always the case), the GPS-enabled handsets supply the application with the location co-ordinates giving a precise position for the device. The co-ordinates consist of Latitude (LAT) and longitude (LNG) information. For example,

```
LAT=51.42869568, LNG=-0.56286722
```

The location information maybe generated by the participants or generated and attested by a trusted $3rd$ party. An example of XML representation of location proof for mobile phones similar to that presented in [49] is given below.

```
<location proof>
<issuer>Issuer's Public Key</issuer>
<recipient>Recipient's Public Key</recipient>
<location information>
<gps><lat>51.42869568</lat><lng>-0.56286722</lng></gps>
<mcc>234</mcc><mnc>15</mnc><lac>43182</lac><cellid>4422847</cellid>
</location information>
</signature></signature>
</location proof>
```

4.2 Preventing Relay Attacks with Location

Examples of enabling mobile applications with 'location proofs' can be found in [49] and in [50]. Assuming that location information is available within mobile phones, the transaction data could be modified to enable the entities involved to verify their proximity. Hu et. al. [51] proposed a simple method using location information to prevent wormhole attacks in wireless sensor networks. This method required that the verifier and prover know their locations and that they have loosely synchronised clocks. The prover would basically add its location and a timestamp to the transmitted data. An additional authentication mechanism, such as a digital signature, is then used to verify that the packet was constructed by the prover. The verifier compares the prover's location to its own and confirms that the prover is in close proximity. If an attacker relays the data then the prover's location should in theory be further away from the verifier's location and the attack would be detected. The attacker cannot modify the data, or construct a new data packet as it does not know the prover's key material. The timestamp prevents an attacker recording a valid transaction and using it at a later stage at the same location. A simple application of this principle is shown in Figure 4. Implementing such a security mechanism in mobile environment is practically feasible and the success of the protocol would simply rely on the accuracy and reliability of the location information available. Using a digital signature scheme, for the required cryptographic data authentication mechanism, is also feasible as mobile phones have sufficient processing resources. The use of digital signatures in NFC applications is also in the process of being standardised in the Signature RTD candidate specification currently being reviewed by the NFC Forum. Also, current work by the authors discusses a secure authentication method for proximity communication channels based on location information. The idea is to attest physical proximity of devices by using location information as an additional security metric. With the identity of the device or subscriber or token bound to the location information it is possible to restrict potential security threats to geographic areas or proximity zones. The method provides a means to test relative and absolute location, and to determine the proximity as well as provide non-repudiation for the devices involved in the transaction.

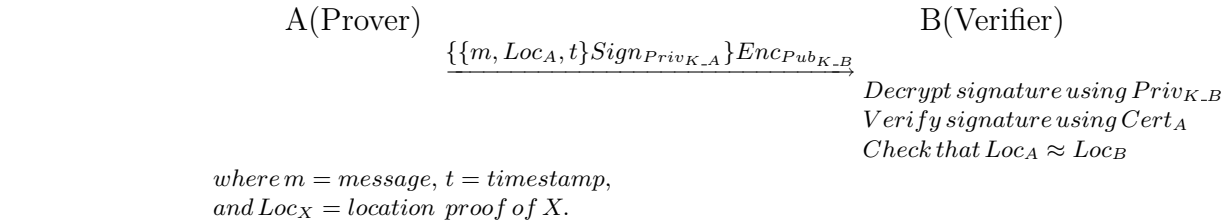


Fig. 4. An example of using location proof for preventing relay attacks in P2P NFC Transactions.

4.3 Limitations of Location Based Security

In this paper, we do not attempt to provide a comprehensive discussion on the issues surrounding the security of location based services, and it should be noted that some location based services have been shown to be insecure, e.g.[52]. However, we do wish to briefly discuss the practical issues in implementing a location-based countermeasure related to the work presented. The limitations of a location-based countermeasure depends upon how the location information is obtained and used within the application. There are external factors, such as host network policy, governing the availability of location information. Some operators may not be prepared to share this information nor to confirm its accuracy unless for legal or investigative reasons. As a result, the application designers would need to consider how the accuracy and detail of available location information could best be exploited. The application design should also take into account the differences in handsets owned by the legitimate participants. The two parties involved in the transaction might subscribe to two different networks. Hence, care needs to be taken in design of the application that generates the location information and its verification, as CellIDs and LACs will vary for different network operator. GPS provides more "independent" access to location information as the mobile phone could determine its own location and not depend on information from the network operator. The downside of this is that the devices would need to support GPS functionality and may not be able to derive GPS co-ordinates indoors. In practice, any application relying on location information to provide security would need to use a combination of location services to best suit the operational environment when a transaction takes place.

5 Conclusion

Peer-to-peer transactions in NFC is being considered for a range of applications including payments. Relay attacks are a threat in contactless and networked environments, and may bypass the security measures employing temporal contracts and cryptography. Our contribution in this paper included a practical demonstration of a first relay attack implementation using NFC-enabled mobile phone platform. We showed that with NFC an attacker can create and introduce proxies by software development (no hardware modification) of suitable MIDlets for the mobile device. The attack did not require any code signing, and did not

need software to be installed in secure program areas such as the SE. It also used standard, easily available APIs such as JSR 257 and JSR 82. The need for countermeasures should therefore be taken seriously, and as discussed the use of location based solution to verify proximity is seen as a promising approach.

References

1. ISO/IEC 18092 (ECMA-340), Information technology Telecommunications and information exchange between systems Near Field Communication Interface and Protocol (NFCIP-1). 2004. http://www.iso.org/, Cited 31 March 2010.
2. Bluetooth Core Specification Version 2.1. + EDR. Volume 2, July 2007.
3. Lin, G., Mikhak, A.A., Nakajima, L.T., Mayo, S.A., Rosenblatt, M.: Peer-to-peer Financial Transaction Devices and Methods. Apple Inc. Patent Application WO/2010/039337, April 2010.
4. Hancke, G.P., Mayes, K.E, Markantonakis, K.: Confidence in Smart Token Proximity: Relay Attacks Revisited. Elsevier Computers & Security, Vol. 28, Issue 7, pp 615–627, October, 2009.
5. Anderson, R.: RFID and the Middleman. Conference on Financial Cryptography and Data Security, pp 46–49, December 2007.
6. Kfir, Z., Wool, A.: Picking Virtual Pockets using Relay Attacks on Contactless Smartcard Systems. Proceedings of IEEE/CreateNet SecureComm, pp 47–58, 2005.
7. Sun Microsystems, JSR-000118 Mobile Information Device Profile 2.0, http://jcp.org/aboutJava/communityprocess/final/jsr118/index.html.
8. ISO/IEC 21481 (ECMA-352), Information technology Telecommunications and information exchange between systems Near Field Communication Interface and Protocol (NFCIP-2). 2005. http://www.iso.org/, Cited 31 March 2010.
9. ISO/IEC 14443, Identification cards – Contactless integrated circuit cards – Proximity cards, http://www.iso.org/, Cited 31 March 2010.
10. ISO/IEC 15693, Identification cards – Contactless integrated circuit cards – Vicinity cards, http://www.iso.org/, Cited 31 March 2010.
11. FeliCa, http://www.sony.net/Products/felica/, Cited 31 March 2010.
12. European Technical Standards Institute (ETSI), Smart Cards; UICC-Terminal interface; Physical and logical characteristics (Release 7), TS 102 221 V7.9.0 (2007-07), http://www.etsi.org/, Cited 31 March 2010.
13. Third Generation Partnership Project, Specification of the Subscriber Identity Module-Mobile Equipment (SIM - ME) interface (Release 1999), TS 11.11 V8.14.0 (2007-06), http://www.3gpp.org/.
14. Third Generation Partnership Project, Characteristics of the Universal Subscriber Identity Module (USIM) application (Release 7), TS 31.102 V7.10.0 (2007-09), http://www.3gpp.org/.
15. Third Generation Partnership Project 2 (3GPP2), Removable User Identity Module (RUIM) for Spread Spectrum Systems, 3GPP2 C.S0023-C V1.0' (May 26 2006), http://www.3gpp2.org/.
16. SD Card Association, http://www.sdcard.org/, Cited 31 March 2010.
17. Candidate Technical Specification: Signature Record Type Definition. NFC Forum. October 2009.
18. Near Field Communication (NFC) Forum, http://www.nfc-forum.org, Cited 31 March 2010.

19. Sun Microsystems, Java Card Platform Specification v2.2.1, http://java.sun.com/products/javacard/specs.html, Cited 31 March 2010.
20. NXP, Java Card Open Platform, http://www.nxp.com/, Cited 31 March 2010.
21. Global Platform, Card Specification v2.1.1, http://www.globalplatform.org, Cited 31 March 2010.
22. NXP Semiconductor. Mifare Standard Specification, http://www.nxp.com/acrobat_download/other/identification/, Cited 31 March 2010.
23. Francis, L., Hancke, G.P., Mayes, K.E., Markantonakis, K.: Potential Misuse of NFC Enabled Mobile Handsets with Embedded Security Elements as Contactless Attack Platforms. Proceedings of the 1st Workshop on RFID Security and Cryptography (RISC'09), in conjunction with the International Conference for Internet Technology and Secured Transactions (ICITST 2009), pp 1-8, November 2009.
24. Mayes, K.E., Markantonakis, K. (Eds.): Smart Cards, Tokens, Security and Applications, Springer Verlag, 2008. ISBN: 978-0-387-72197-2.
25. Sun Microsystems, JSR-000257 Contactless Communication API 1.0, http://jcp.org/aboutJava/communityprocess/final/jsr257/index.html.
26. Conway, J.H.: On Numbers and Games. Academic Press, 1976.
27. Desmedt, Y., Goutier, C., Bengio, S.: Special Uses and Abuses of the Fiat-Shamir Passport Protocol. Advances in Cryptology (CRYPTO), Springer-Verlag LNCS 293, pp 21, 1987.
28. Hu, Y.C., Perrig, A., Johnson, D.B.: Wormhole Attacks in Wireless Networks. IEEE Journal on Selected Areas in Communications (JSAC), pp 370–380, 2006.
29. Hancke, G.P., Kuhn, M.G.: An RFID Distance Bounding Protocol. Proceedings of IEEE/CreateNet SecureComm, pp 67–73, September 2005.
30. Hancke, G.P., Practical Attacks on Proximity Identification Systems (short paper). Proceedings of IEEE Symposium on Security and Privacy, pp 328–333, May, 2006.
31. Libnfc.org, Public Platform Independent Near Field Communication (NFC) Library, http://www.libnfc.org/documentation/examples/nfc-relay, Cited 31 March 2010.
32. RFID IO Tools. rfidiot.org. Cited 31 March 2010.
33. Sun Microsystems, JSR-000082 Java API for Bluetooth 2.1, http://jcp.org/aboutJava/communityprocess/final/jsr082/index.html, Cited 31 March 2010.
34. Sun Microsystems, Java Code Signing for J2ME, http://java.sun.com/, Cited 31 March 2010.
35. Nokia Forum, Java Security Domains, http://wiki.forum.nokia.com/index.php/Java_Security_Domains Cited 31 March 2010.
36. Nokia Forum, MIDP 2.0 API Access Rights, http://wiki.forum.nokia.com/index.php/MIDP_2.0_API_access_rights Cited 31 March 2010.
37. Nokia Forum, MIDP 2.1 API Access Rights, http://wiki.forum.nokia.com/index.php/MIDP_2.1_API_access_rights Cited 31 March 2010.
38. Nokia Forum, Nokia 6131 API Access Rights, http://wiki.forum.nokia.com/index.php/API_access_rights_on_phones,_Series_40_3rd_FP1 Cited 31 March 2010.
39. Nokia Forum, Nokia 6212 API Access Rights, http://wiki.forum.nokia.com/index.php/API_access_rights_on_phones,_Series_40_5th_FP1 Cited 31 March 2010.

40. Brands, S., Chaum, D.: Distance Bounding Protocols. Advances in Cryptology, EUROCYPT '93, Springer-Verlag LNCS 765, pp 344–359, May 1993.
41. Clulow, J., Hancke, G.P., Kuhn, M.G., Moore, T.: So Near and Yet So Far: Distance-Bounding Attacks in Wireless Networks. European Workshop on Security and Privacy in Ad-Hoc and Sensor Networks (ESAS), Springer-Verlag LNCS 4357, pp 83–97, September 2006.
42. Hancke, G.P., Kuhn, M.G.: Attacks on Time-of-Flight Distance Bounding Channels. Proceedings of the First ACM Conference on Wireless Network Security (WISEC'08), pp194–202, March 2008.
43. Rasmussen, K.B., Čapkun, S.: Implications of Radio Fingerprinting on the Security of Sensor Networks. Proceedings of IEEE SecureComm, 2007.
44. Danev, B., Heydt-Benjamin, T.S., Čapkun, S.: Physical-layer Identification of RFID Devices, Proceedings of USENIX Security Symposium, 2009.
45. Anderson, R.J., Bond, M.: The Man-in-the-Middle Defense. Presented at Security Protocols Workshop, March 2006. http://www.cl.cam.ac.uk/ rja14/Papers/Man-in-the-Middle-Defence.pdf
46. Stajano, F., Wong F.L., Christianson, B.: Multichannel Protocols to Prevent Relay Attacks. Conference on Financial Cryptography and Data Security, January 2010.
47. Boukerche, A., Oliveira, H.A.B., Nakamura, E.F., Loureiro, A.A.F.: Secure Localization Algorithms for Wireless Sensor Networks. *IEEE Communications Magazine*, Vol. 46, No. 4, pp 96–101, April 2008.
48. Google Maps, Google Inc., http://www.googlemaps.com/ Cited 31 March 2010.
49. Saroiu, S., Wolman, A.: Enabling New Mobile Applications with Location Proofs. In Proceedings of the 10th Workshop on Mobile Computing Systems and Applications, Santa Cruz, California, February 23 - 24, HotMobile 2009, ACM, New York, USA, 1-6, 2009.
50. Luo, W., Hengartner, U.: Proving your Location without giving up your Privacy. In Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications, Annapolis, Maryland, February 22 - 23, HotMobile 2010, ACM, New York, USA, 7-12, 2010.
51. Hu, Y.C., Perrig, A., Johnson, D.B.: Packet leashes: A Defense Against Wormhole Attacks in Wireless Networks. Proceedings of INFOCOM, pp 1976–1986, April 2003.
52. Tippenhauer, N.O., Rasmussen, K.B., Pöpper, C., Capkun, S.: Attacks on Public WLAN-based Positioning. Proceedings of the ACM/Usenix International Conference on Mobile Systems, Applications and Services (MobiSys), 2009.