A Practical Generic Relay Attack on Contactless Transactions by Using NFC Mobile Phones

Lishoy Francis
Trust Team, Orange Labs UK
Orange, France Telecom R&D
Building 10, 566 Chiswick High Road
Chiswick Park, W4 5XS
London, United Kingdom

Gerhard Hancke
Department of Computer Science
83, Tat Chee Avenue
Kowloon
Hong Kong

Keith Mayes
Information Security Group
Smart Card Centre
Royal Holloway University of London
Egham Hill, TW20 0EX
Surrey, United Kingdom

***Abstract*—Contactless technology is widely used in security sensitive applications, including identification, payment and access-control systems. Near Field Communication (NFC) is a short-range contactless technology allowing mobile devices to act primarily as either a reader or a token. Relay attacks exploit the assumption that a contactless token within communication range is in close proximity, by placing a proxy-token in range of a contactless reader and relaying communication over a greater distance to a proxy-reader communicating with the authentic token. It has been theorised that NFC-enabled mobile phones could be used as a generic relay attack platform without any additional hardware, but this has not been successfully demonstrated in practice. We present a practical implementation of an NFC-enabled relay attack, requiring only suitable mobile software applications. This implementation reduces the complexity of relay attacks and therefore has potential security implications for current contactless systems. We also discuss countermeasures to mitigate the attack.**

I. Introduction

Radio Frequency Identification (RFID) technology has become increasingly prevalent in everyday applications. Contactless technology is a subset of RFID systems operating at 13.56 MHz, with an operating range of up to 10 cm. This technology comprises mature standards and industry specifications and is widely used by the smart card sector in security sensitive systems. Contactless technology is currently used in credit card payment [1], [2], [3], e-ID and e-passport systems [4], [5], transport ticketing [6], [7] and access control systems [8], [9]. The practical security of contactless systems is therefore an active research area, both in terms of the actual channel [10], [11], [12] and deployed applications [13], [14], [15].

Relay attacks are especially of interest with regards to contactless application security [16]. Contactless systems, as a result of the limited operational range, operate on the implicit assumption that successful communication with a token proves that the token is in close proximity of the contactless reader. Therefore, once authentication has been achieved at the application layer, the reader will approve a transaction or render a service as it believes that the legitimate token is in its presence. A relay attack exploits this assumption by placing a proxy-token within the communication range of the reader, which communicates with a proxy-reader located in close proximity to the legitimate token. The proxy-token is always able to answer with a valid response to any reader command because it simply forwards the command to the proxy-reader, which in turn sends it to the legitimate token and returns the valid response from the legitimate token to the proxy-token. For the duration of the relay attack the proxy-token exhibits the same behaviour as a legitimate token from the reader's perspective. This attack effectively circumvents application layer security mechanisms. For example, an attacker can circumvent an authentication protocol by simply relaying a challenge to the real token, which will provide him with the correct response, which can then be relayed back to the reader via the proxy-token. It does not matter what application layer protocols or security algorithms are used, as the attacker just relays all the application layer data, thereby ensuring that both the legitimate reader and the legitimate token always receive the data they expect.

Near Field Communication (NFC) is a short-range RFID technology intended to equip mobile devices with a contactless communication channel compatible with existing contactless technology. An NFC-enabled device is able to act like a passive contactless token, which can be read by contactless readers. Alternatively, an NFC-enabled device can act as a contactless token reader. NFC-enabled devices can also speak to each other by using a specified 'peer-to-peer' mode. NFC is not a new technology, having been invented almost a decade ago and actively promoted by the NFC Forum [17] since 2004. NFC has been the focus of numerous worldwide trials and proof-of-concept demonstrations, but large scale deployment was hampered by disagreement regarding NFC-device architecture, application management and the resultant lack of NFC devices. In 2011 NFC has, however, become increasingly prominent with a number of phone manufacturers releasing NFC-enabled smart phones, such as the Nokia C7 [18], RIM Blackberry 9900/9930 [19] and Google Nexus S [20]. At the same time, NFC has also made rapid strides in enabling mainstream applications, as illustrated by the release of Google Wallet [21] and Orange Quick Tap [22] payment

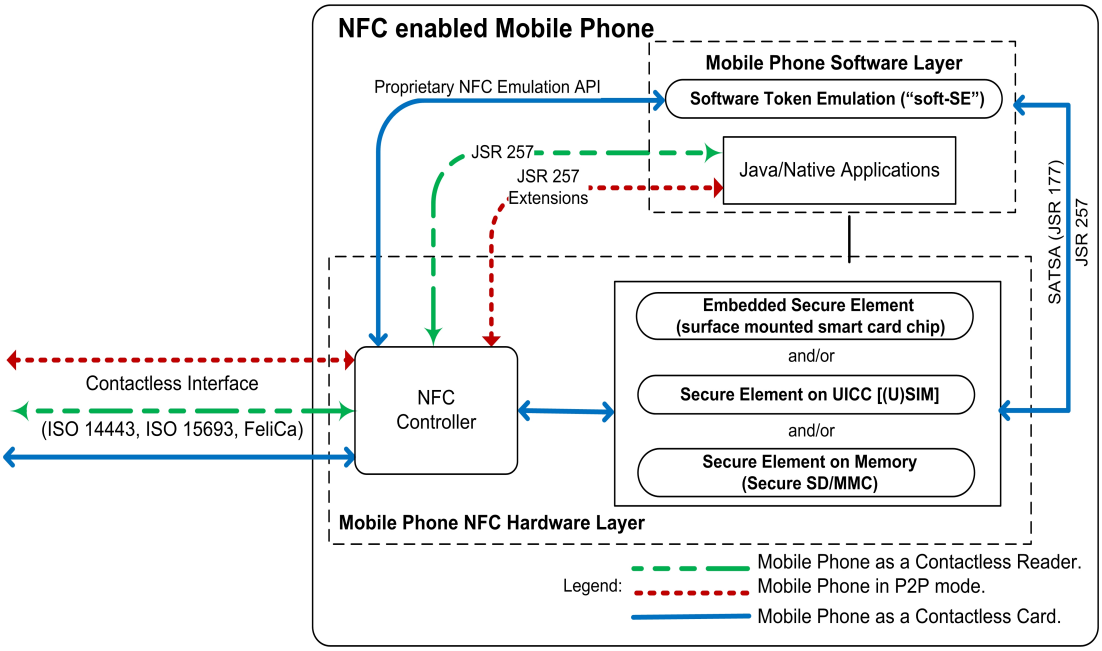


Fig. 1. NFC architecture options with SE available as a software emulation ("soft-SE") via mobile phone APIs.

systems in the USA and UK respectively.

As the deployment of NFC gathers speed the security of NFC devices and applications becomes increasingly important [23], and in addition the security implications of providing access to what is essentially a programmable contactless reader and token emulation platform should also be considered. For example, it has been shown that an NFC-enabled mobile phone can be used as an effective token skimming and cloning platform [24]. The ability of an NFC-enabled device to act as both a token and a reader potentially makes such device an ideal platform for implementing software relay attacks, as theorised in multiple publications [25], [26], but this has not been proven to be practically possible in a generic manner. This paper describes a relay attack implementation using unmodified NFC-enabled mobile phones, which only requires an attacker to write suitable mobile platform applications using publicly available APIs. Our relay attack implementation significantly demonstrates a reduced complexity of attack as it does not require special attack hardware, as in some previous relay attack experiments [27]. This implementation also results in an attack that cannot be visibly detected in contrast with attacks with PC-controlled NFC-enabled devices acting as proxy-token [28], [29] since an NFC phone is (or will soon be) an accepted token form factor. The attack implementation is application independent and works against widely deployed, conventional contactless system configurations, i.e. a reader and a passive contactless token, and not only against the NFC peer-to-peer communication mode [30]. The practical simplicity of such a relay attack implementation increases the likelihood of this exploit being used in practice and places real-world systems at risk. Modern contactless credit card and m-wallet payment systems, ticketing, access control and electronic identification schemes are vulnerable to relay attacks, and an attack that can effectively be executed by unskilled attackers using off-the-shelf hardware represents a credible threat. This work could potentially change system implementers' view of preceding work on relay attacks, which is mostly dismissive and can be summarised by the quote "There's been no example of it happening in the real world, and we find it highly unlikely that it will happen" [31]. This paper challenges the currently held opinion that relay attacks require advanced skill and custom hardware that is unlikely to transition from a laboratory to the real world [32]. Apart from the risks a relay attack poses, practically implementing a 'proof-of-concept' attack using NFC mobile phones serves to emphasise the current weaknesses in NFC architecture that would need addressing.

This paper starts with a general discussion of NFC technology and relay attacks in Section II. Our relay attack implementation on mobile phones, its effectiveness and experimental observations are discussed in Section III. Finally, potential countermeasures against relay attacks are discussed and analysed in Section IV.

II. Background

In this section, we present a brief overview of NFC technology. We then go on to discuss relay attacks and related work.

A. NFC Technology

NFC facilitates the integration of contactless technology into active device platforms, such as mobile phones. NFC is a short-range RFID technology operating at the 13.56 MHz radio frequency (RF) band and is described in the ISO 18092/ECMA 340 [33] and in ISO 21481/ECMA 352 [34] standards. NFC

is specified to be compatible with existing contactless systems adhering to ISO 14443 [35], ISO 15693 [8] and FeliCa [36]. The standards specify both 'passive' and 'active' operation. Passive operation corresponds to the operation of conventional contactless systems. The NFC device can therefore either act like a contactless token, interacting with a reader, or act like a reader, powering and interacting with a contactless token. Two NFC devices can also interact with each other in active, or peer-to-peer (P2P) mode, when brought in close proximity. In this active mode, devices take turns to transmit an RF field, e.g. device 1 turns on its RF field and transmits data to device 2, followed by device 1 turning off its field and device 2 turning on its field and transmitting data to device 1. Peer-to-peer relay has been covered in a previous publication and is not the focus for this paper [30]. It is expected that NFC will be deployed in existing contactless applications, such as payments, ticketing, access control, identification and logistics. NFC in conjunction with the additional functionality of its host platform could also enable additional applications, such as one of the early proposals of using NFC for quickly pairing Bluetooth devices [37].

Today, there are a number of NFC-enabled devices available but mobile phones are the main focus of industry and this paper. More details of the NFC phone platform as relevant to our implementation are discussed in Section III. There are three main components that comprise an NFC-enabled phone platform [38] (an overview is shown in Figure 1):

- Application Execution Environment (AEE): The general application area of the mobile phone providing data storage and processing capabilities alongside basic mobile phone services.
- Trusted Execution Environment (TEE): The TEE is usually realised through the use of a secure element (SE) and provides secure data storage, execution and application management. A SE is essentially a smart card supporting Java Card 2.2.1 [39] (Java Card Open Platform [40]), Global Platform 2.1.1 [41] and selected legacy products such as the Mifare Classic [42] emulation. An SE is most commonly implemented as an embedded module, i.e. a surface-mounted module soldered into the phone, as an integrated component on the (U)SIM (Universal/Subscriber Identity Module) [43], or as a removable secure memory token [44]. A new development is the concept of a "soft-SE" located within the mobile phone application area. The "soft-SE" is open for development, in contrast to earlier SE modules that had to be unlocked for development use. For example, using an "unlock" application supplied by the phone manufacturer. Once unlocked, an SE is forever considered as untrusted and can subsequently be used only for development purposes. An NFC phone will contain one or more of these SE implementations.
- NFC Controller (low level stacks): The NFC Controller handles the physical transmitting and receiving of data over the RF interface. The card emulation stack, reader/writer stack and peer-to-peer stack allow for communication between the controller and the AEE/TEE as required by the respective mode of operation. Reader and peer-to-peer operations are generally controlled via applications in the AEE, with card emulation being controlled via applications in the TEE, i.e. executing within an SE.

With the exception of the application management on the SE and the Signature Record Type Definition (SRTD) [45], which aims to provide data authentication for data in NFC Data Exchange Format (NDEF) [46], the NFC specifications and standards leave application security in the hands of the developer. There have been several research papers discussing NFC security, such as [23]. Research work has been published both on vulnerabilities in the specifications, such as the vulnerabilities in the SRTD [47], and NFC software stacks allowing tags to redirect to spoofed web addresses or load malicious software [48], [49]. Given the computational capabilities of the phone platform, and the added capabilities of NFC to act as a reader and a token, the possibility of using an NFC phone as a platform for contactless "skimming" and "cloning" platforms has also been [24] considered.

B. Relay Attack

A relay attack can be best explained conceptually with the help of the Grand Master Chess problem as discussed in [50]. In this scenario, a person who does not know the rules of chess could play against two grand masters by challenging both of them to a postal game. The player would then simply forward the move received from one grand master to the other, effectively making them play against one another. Each grand master would think that they are playing said person, but in reality they are playing against each other. The application of this scenario to security protocols was first presented and discussed in [51]. In the literature, this attack has subsequently been referred to as a 'wormhole attack' [52] or as a 'relay attack' [53].

A relay attack has serious security implications as the attacker is able to bypass any application layer security protocol, even if such protocols were based on strong cryptographic principles. For example, an attacker can circumvent an authentication protocol by simply relaying a challenge to a legitimate token, which will provide him with the correct response, which can then be relayed back to the verifier. It does not matter what application layer protocols or security algorithms are used, in fact the attacker requires no prior knowledge about the data he is relaying, as the attacker just relays all the application layer data, thereby ensuring that both the reader and the token always receive the data they expect. If the overarching protocol contains a security vulnerability the attacker could also modify the relayed data in real time to exploit this vulnerability, an action often referred to as an 'active' relay [16].

To execute a relay attack, the adversary needs two devices, which act as a token and a reader respectively. These devices are connected via a suitable communication channel in order to relay information over a greater distance. The proxy-reader

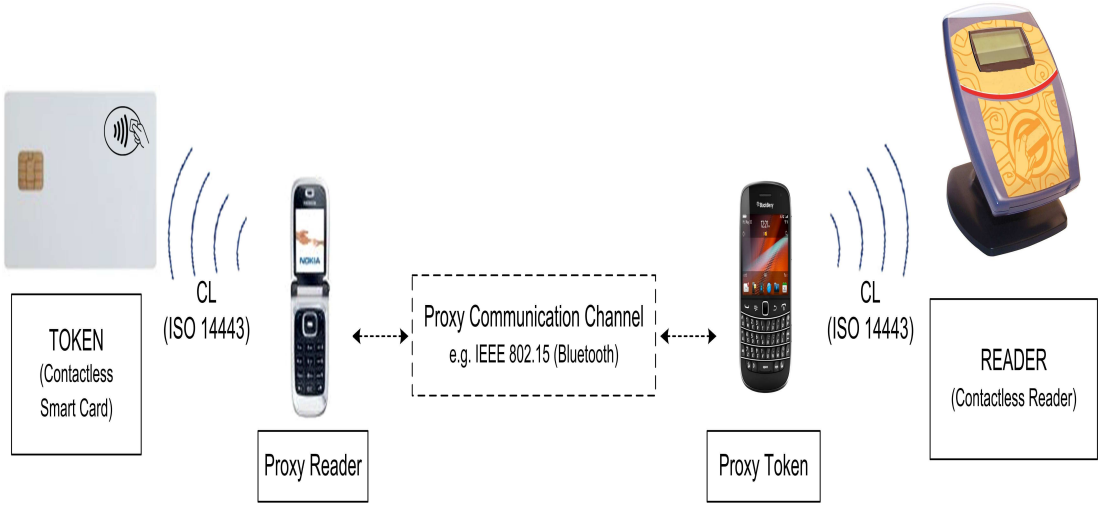


Fig. 2. Practical relay setup using only NFC mobile phones.

is used to communicate with the real token, while the proxy-token is placed near the real reader. Any information transmitted by the reader is received by the proxy-token and relayed to the proxy-reader, which will transmit the information to the token. The token assumes that it is communicating with the reader and responds accordingly. The token's response is then relayed back to the proxy-token, which will transmit the information to the reader. The intention of the attacker is to ensure that the reader is unable to distinguish between the real token and the proxy. If he succeeds the reader will assume that the token and its associated owner are in close proximity and grant access to the attacker.

Several practical implementations of relay attacks in the contactless environment have been published. The earliest impactful implementation was a demonstration of a relay attack against EMV payment systems using contact-based cards [56], which illustrated the vulnerability of deployed systems to relay attacks and showed that even real-world systems that are engineered to be secure contain no countermeasures to attacks of this type. These implementations often required custom-built hardware [27], [54] or the use of NFC-enabled contactless readers controlled by a host computer [28], [55]. In some cases, the use of custom hardware is not a negative. In certain systems readers are unattended, or as in the case of [54] the use of custom hardware is part of the attack's success as the system does not use technology used widely in other applications. The drawbacks are that these implementations yield proxy-tokens that can easily be spotted as out of the ordinary, in the case of [56] some social engineering and coordination was required as the attacker has wire running down his sleeve to the card presented to the vendor. The complexity of such attacks have been argued to potentially limit their widespread use in exploiting current systems [31], [32]. In contrast, an attack implemented entirely on an NFC-enabled phone, requiring an attacker to only download and install suitable applications, is more likely to become a practical threat. The scenario of a relay attack implemented against conventional contactless systems using only mobile phones, envisaged in [25], has not been practically demonstrated but has been the target of some research initiatives. In [29] a phone was used as the proxy-reader and an NFC-enabled reader acted as the proxy-token. In [30] it was shown that the communication between two NFC devices communicating in P2P mode could be relayed using two NFC phones. In both these cases the authors did not succeed in implementing a proxy-token acting as a passive contactless token as would be required when relaying conventional contactless transactions.

III. Practical Relay Implementation

In this section, we describe the practical implementation of a relay attack using only off-the-shelf NFC mobile phones. We implemented the attack with two commercially available NFC-enabled mobile phones and conducted several controlled relay experiments to verify the effectiveness of the attack. Both the proxy-token and proxy-reader mobile phones are configured simply by installing mobile phone applications that we developed. The attack implementation requires no unlocking of devices or secure elements, no hardware or software modification to the phone platform, and minimal knowledge of the data that is to be relayed. We also chose to implement the relay channel in such a way that it could be set up between the two phones without the need for relying on access to a mobile network. The relay setup for attacking a contactless system, as implemented in this paper, is shown in Figure 2.

A. Proxy Communication Channel using NFC Mobile Phones

In a relay attack, the attacker and his/her accomplice uses proxy-devices that communicate over a proxy channel. The relay experiment thus requires a high-speed and reliable communication link between the two NFC mobile phones implementing the proxy-reader and proxy-token.

Bluetooth was chosen as communication channel for our relay experiments. Bluetooth, or IEEE 802.15, is a short-range radio technology developed by the Bluetooth Special Interest Group (SIG). It utilises unlicensed radio spectrum in the frequency band of 2.45 GHz, offering bandwidth in the range of 720 kilobits per second and an effective operating range typically in the region of 10 m to 100 m. Point-to-point Bluetooth is simple to set up and the communication latency offered by the channel is relatively low. Although the communication range offered by Bluetooth could be seen as a limitation our aim was to demonstrate the feasibility of establishing a relay channel between two mobile phones. In reality, the proxy channel could be realised via other technologies such as IEEE 802.11 or mobile Internet over GPRS/E-GPRS (Enhanced General Packet Radio Service). Relaying data via mobile Internet requires no user-interaction and potentially offers both increased bandwidth and low latency if good network coverage is available. It would therefore appear to be a good alternative channel option but it introduces a reliance on the mobile network, i.e. the attack is only effective if there is network coverage and a reliable data service. It could also be argued the when using mobile Internet data leaves an audit trail of relayed data, whereas the use of Bluetooth channel does not relay on third party infrastructure.

The mobile applications installed on the proxy-reader and proxy-token implemented Bluetooth communication using the JSR 82 API. We used L2CAP (Logical Link Control and Adaptation Protocol) that is available within the host stack of Bluetooth protocol. L2CAP is layered over the Baseband Protocol, and operates at the data-link layer within the OSI (Open System Interconnection) Reference Model. The supported data capacity of the channel, for individual packets, is up to 64 kilobytes in length. The default Maximum Transmission Unit (MTU) is 672 bytes, and 48 bytes is the minimum mandatory MTU. More details for implementing the Bluetooth API can be found in [30], [37], [57].

B. NFC Mobile Phone as Proxy-Reader

A Nokia 6131 NFC phone was configured as a proxy-reader (controlled with an MIDP/J2ME Application [58]) capable of interacting with contactless tokens. This involved developing a MIDP 2.0 application (which is commonly known as a MIDlet) to emulate a contactless reader using a standard NFC contactless communication API - JSR 257 [59]. This application was developed by using a freely available Nokia NFC Software Development Kit (SDK) [60]. The MIDlet was designed to exchange ISO 14443-4 based Application Protocol Data Unit (APDU), such as those received from a proxy-token over the relay communication channel, with external contactless smart cards. For using the JSR 257 API and JSR 82 Bluetooth API in the MIDlet, it did not require any code signing [61] in order to install and execute the application.

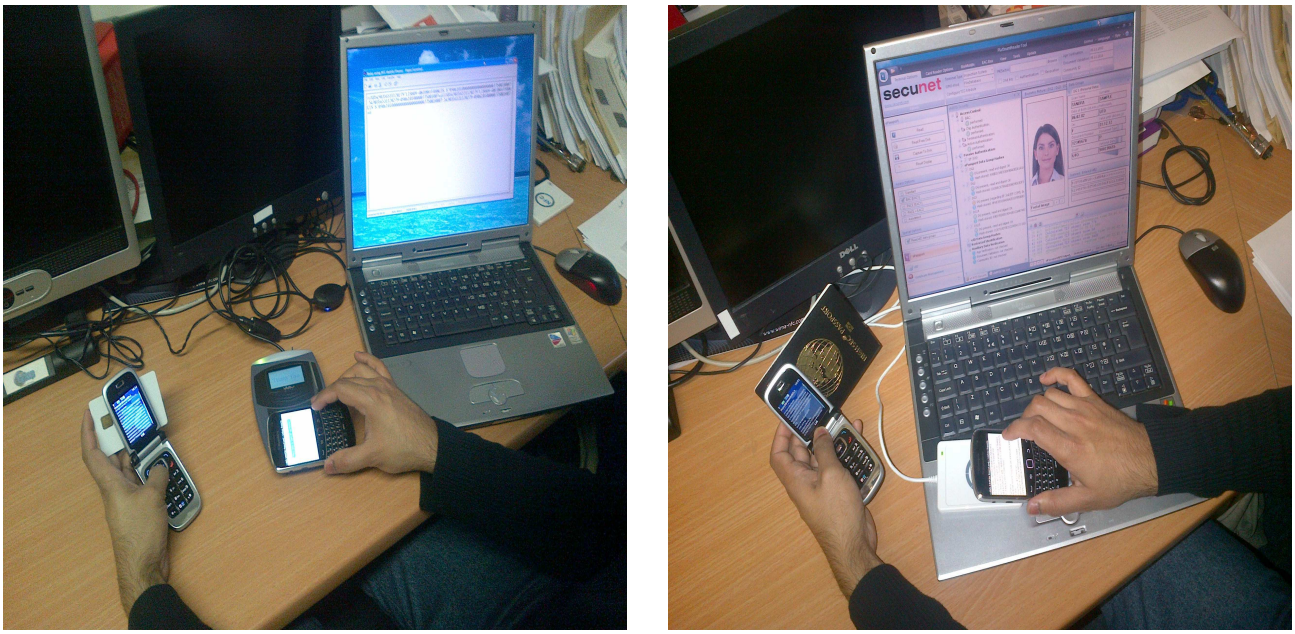
C. NFC Mobile Phone as Proxy-Token

In a relay system with only NFC enabled mobile phones, the main challenge is to configure the phone as a controllable proxy-token. A proxy-token needs to receive command messages from the reader, relay them to a proxy-reader and then present the relayed responses back to the reader, all in an orderly and timely fashion. During previous development work on specific legacy NFC phones we found that the embedded SE could not support multiple communication sessions. This meant that once an SE emulating a token received a command from a reader it was bound to that communication session and unable to send the received command to the relay channel. This is an observation subsequently also made in [29]. In contrast, we found that (U)SIM SEs were capable of maintaining multiple sessions, potentially making a relay attack possible. (U)SIMs are however tightly controlled by mobile network operators and obtaining such SEs for development is difficult, which inherently limits the appeal of such an attack implementation.

The release of the NFC-enabled Google Nexus S and BlackBerry 9900/9930 phones has provided more freedom in developing applications requiring card emulation functionality. Although the Nexus S does not yet support card emulation functionality as standard, it has been shown that it is possible modify the phone firmware to allow for user controlled card emulation as a result of the open nature of the Android OS [62]. The BlackBerry phones, running Blackberry OS v7.1, allow user-controlled card emulation without modification. We therefore chosen to implement the proxy-token on a BlackBerry 9900 phone, keeping with our goal of demonstrating a simple, software-only NFC relay attack.

The BlackBerry v7.1 NFC API[1] provides for the emulation of contactless applications based on a “soft-SE”. This approach offers greater flexibility in application development but also increases the likelihood that the phone could be used as an attack platform. To start with, we tested whether it was possible to create a contactless application with a reserved Application Identifier (AID), i.e. an AID associated with a sensitive application such as credit card payments [24]. The ability to set the AID in such a way provides an ideal entry point for the relay process, as the reader would inherently select and start communicating with the relay application. We found that no security controls were in place to prevent spoofing a legitimate reserved AID and also that this emulation method allowed the emulation application to be in session with the reader while also accessing other system components, thereby making it possible to relay received commands. The BlackBerry NFC mobile phone was thus configured as a proxy using a BlackBerry Java Application we developed that utilised the BlackBerry-specific NFC emulation API v7.1 [63], and implemented Bluetooth communication using the JSR 82 API [57]. The NFC emulation API [63] did not require mandatory code-signing, although the underlying Runtime API was required to be signed in order to install and run the application on the device. The registration process and subsequent acquiring of the signing certificate did however not involve any formal organisational or personal vetting [64].

[1] net.rim.device.api.io.nfc.emulation



(a) Contactless credit card transaction

(b) e-Passport transaction

Fig. 3. Testing relay attack implementation on real systems.

D. Proof-of-Concept Relay Experiment

Initially, we simply tested whether our implementation would relay a single command response transaction using a test setup involving a contactless reader and a token containing a simple Java Card [39] applet. The proxy-token was presented to the contactless reader and the legitimate contactless token was presented to the proxy-reader and it was determined that the reader obtained an acceptable response (correct content and adequate response timing). Subsequently we managed to perform a relay involving multiple commands from the reader, reliably completing a full legitimate contactless transaction during each run. Using the Bluetooth channel in a non line-of-sight environment the attack worked up to a range of 15m. In an open plan room with some minor obstacles the attack worked up to a range of 35m.

We also set up two additional laboratory controlled experiments. The first was a test payment system based on first generation contactless credit cards, i.e. static authentication credentials, using a contactless point-of-sale (POS) terminal and a 'card' we constructed using a valid card data profile. Cards using static authentication in this way are no longer best-practice but the objective of the experiment was more focused on whether the POS would accept a delayed relayed response as valid. Newer cards using dynamic authentication protocols are equally vulnerable to relay attacks as the dynamic challenge and response is as easily relayed. The relay experiment on a payment transaction using a POS reader, and a contactless smart card with a sample payment application installed is shown in Figure 3(a). In the second experiment, we tested the relay against an e-passport demonstration system using a sample passport and authentic reader software. The emphasis in this case was once again whether the reader would accept a relay response, and in addition this setup also tested whether longer data APDUs, such as the passport record including a JPEG picture, could be reliably relayed. The test setup is shown in Figure 3(b). Additionally, we tested our attack on a system based on MIFARE DESFire EV1 [65], an ISO 14443 based smart card technology that is widely used in public transport schemes, access management, closed-loop e-payment, and contactless ticketing applications. In all cases the relay attack executed successfully and the reader/POS accepted the proxy-token's responses. We would like to highlight the fact that these systems were not chosen because they are known to be vulnerable to relay attacks, these are just systems we had access to. The attack as implemented would work on any system with communication fully compatible to NFC or ISO 14443 contactless technology, which includes most payment and m-wallets, electronic identity, ticketing and access control systems deployed today. Some legacy contactless products that are only partially compatible with the standard, and use proprietary APDUs, might be resistant to this attack implementation. Relaying contactless transaction data relies on the attack application receiving the received data from the NFC communication module, which is responsible for demodulation, decoding and stripping off frame information such as CRCs, parity bits and stop/start patterns and providing data left to the application layer. Some contactless card systems use proprietary framing, which means that the NFC module would not be able retrieve the data in the normal way. Additional detail on this attack restriction is given in Section IV-B2.

TABLE I
TIMING MEASUREMENTS OF A SAMPLE TRANSACTION (APDU COMMAND/RESPONSES) (IN MILLISECONDS)

	(a) Contactless Smart Card	(b) Embedded SE	(c) "soft-SE"	(d) Relay (Proxy-Token on "soft-SE")
Command 1	113.3917	190.5490	181.1386	246.0
Command 2	12.1902	20.2325	12.8123	114.0
Command 3	4.1948	5.2614	10.7270	109.0
Command 4	17.0420	18.5147	38.3187	118.0
Command 5	4.8180	5.8939	15.4194	78.0
Total	151.6367	240.4515	258.416	665.0

E. Experimental Analysis and Further Tests

The attack parameter of most interest is the round-trip time required by the relay process. The main delay is caused by the relay communication channel. The Bluetooth channel introduces approximately 50 ms into the the round-trip-time of the challenge-response. Although all the readers we tested accepted the delayed responses of the proxy-token, we wanted to quantify this delay and compare the performance of a proxy-token to other emulation implementations as a matter of scientific interest. Table I shows the response times of several command and response message sequences on different emulation platforms, with respect to the payment test system discussed in the previous section. The times shown in the Table is 'best-case', the shortest times observed over several measured transaction runs. In the worst case about 30 ms is added to response times. We measured the response time of a sample payment application implemented on a programmable contactless smart card, on an embedded SE and on a "soft-SE". Although timing measurements varied between protocol runs, these implementations were in general all significantly quicker than the response time of the proxy-token, but this is to be expected taking into account the overhead involved with the relay process. We believe that the response time for the proxy-token could potentially be made faster, as there is some room for optimisation of our application with regards to the implementation of the relay communication channel.

We also wanted to determine what the maximum time is that an attacker has to relay transactions before the reader refuses the response. We programmed a variable timer routine into the proxy-token application and systematically increased the time until the attack failed. In the case of the POS reader the allowable attack time was up 35000 ms and for the passport system reader the allowable attack time was up 5200 ms. This has significant implications, as 35 and 5.2 second is a long time in terms of modern communication systems. This could potentially allow an attacker to extend the effective range of the attack. We tested the latency of a potential relay channel implemented using a WiFi access point and estimate that such a channel, including initial connection and session setup, would introduce about a 1.5 second delay to the attack. This time increase is still acceptable to the readers we tested, with the implication that the proxy-reader and proxy-token can now be situated anywhere in the world and still relay acceptable transactions.

IV. SECURITY COUNTERMEASURES FOR RELAY ATTACK

In this section we discuss potential security countermeasures and their effectiveness in mitigating the relay attack presented in Section III. We only consider countermeasures that can be implemented without degrading the user experience, which is one of main advantages of contactless technologies. We therefore do not discuss measures that shift the responsibility of security to the end user, such as shielding tokens or performing two-factor authentication with a PIN. The countermeasures can be divided into two main categories:

- Contactless Platform Countermeasures: countermeasure proposals for treating the phone as a resource-limited contactless token, i.e. simple mechanisms implemented by the reader and back-end infrastructure.
- Mobile Phone Platform Countermeasure: countermeasures leveraging the capabilities of the mobile phone platform to enhance the security of the contactless transaction.

A. Contactless Platform Countermeasures

This section briefly examines security countermeasures proposed for making contactless systems resistant to relay attacks.

1) Timing: One of the intuitive countermeasures is enforcing stricter timing restraints on responses. This is based on the valid observation that a relayed transaction will have an increased response time in comparison to a legitimate transaction. It is, however, difficult to implement this in practice. Firstly, obtaining accurate transaction timing information on current readers is a challenge, considering the number of underlying process components adding overhead. Accurate response timing would likely require dedicated hardware that directly monitors the RF channel. In real world systems there is also a need to accommodate a variety of contactless tokens, which vary in terms of performance, so setting a restrictive timeout value could lead to valid transactions being rejected. The method that ISO 14443 (which is the contactless standard used in the majority of security sensitive contactless applications and serves as the basis for NFC) mandates for negotiating communication parameters between the reader and token also negates the use of timeouts. ISO 14443 Part 4 specifies a Frame Waiting Time (FWT) variable that sets the time within which a token shall start its response after the end of the reader's data. FWT is defined as $(256 \cdot 16/f_{\text{carrier}}) \times 2^{FWI}$, where FWI is a value from 0 (FWT = 300 μs) to 14 (FWT

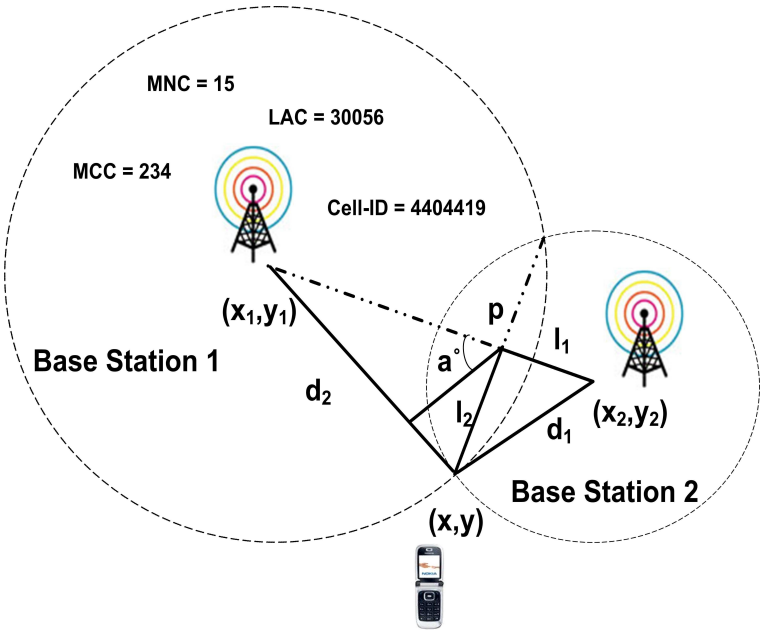


Fig. 4. Network cell broadcast based location sensing and triangulation.

= 5 s) with a default of 4 (FWT = 4.8 ms). The value of the Frame Waiting Integer FWI is defined by the token in the *ATS* response. If implemented, the Frame Waiting Time defines an upper bound on the relay delay. Even though this value is set it is seldom enforced by the reader, as was seen in our experimented, and instead replaced by a much longer timeout. Even if a reader did enforce the FWT it is not a suitable countermeasure because it is the token that specifies the Frame Waiting Time (FWT) during the communication setup. As the token specified the time within which it shall start its response a proxy-token could simply specify a FWT of up to 5 seconds [16], more than enough time to complete the relay process.

2) Distance Bounding: Distance-bounding protocols determine an upper bound for the physical distance between two communicating parties based on the Round-Trip-Time (RTT) of cryptographic challenge-response pairs [66], and it has been proposed that these are suitable for relay-resistant RFID systems [53]. Distance bounding is in theory the most effective countermeasure but this approach requires special communication channels to facilitate accurate and secure distance estimates, since conventional RF channels have been shown inadequate for implementation of secure distance bounding [67], [68]. Although much progress has been made on practical distance bounding implementations for smart tokens [56], [69] the integration of such channels into NFC-enabled devices has not been an industry priority.

B. Mobile Phone Platform Countermeasures

Previous work on relay resistant systems often operated under the assumption that the contactless token was a resource-limited device that relied almost entirely on the reader to function. In comparison, an NFC-enabled mobile phone platform acting as token has relatively abundant resources, such as its own power supply, additional communication links, increased processing capability and a selection of hardware peripherals. The resource-limited paradigm should therefore no longer be a constraining factor when considering relay countermeasures.

1) Location as Security Metric: Even though the use of location information in mobile network access systems has given rise to many applications and services, the capabilities of mobile phones to deduce both absolute and relative location are not utilised for verifying the proximity of devices conducting a transaction. Reliable and accurate location information is an effective countermeasure against relay attacks, e.g. location information could be simple appended to a transaction that is then signed by the legitimate sender [52], and as has also been shown to enable other security services [70], [72]. In fact the use of location information available in the mobile environment to provide security services is not new [71], [73], and could serve as an ideal countermeasure in NFC systems, which as intrinsically linked to mobile. In this section, we discuss the potential role of mobile location-based services in preventing relay attacks on transactions between NFC-enabled phones, or an NFC-enabled mobile phone and a reader with knowledge of its own location.

a) Network Cell Broadcast: The simplest method of retrieving mobile location information is using metrics from the cell broadcast towers or base stations. These include a Cell-ID identifier associated with parameters such as Mobile Country Code (MCC), Mobile Network Code (MNC) and Location Area Code (LAC). The cell broadcast information can be retrieved by using location APIs from the mobile software platform or from the (U)SIM. This approach is applicable to most traditional mobile phones used in mobile network access systems such as GSM and UMTS.

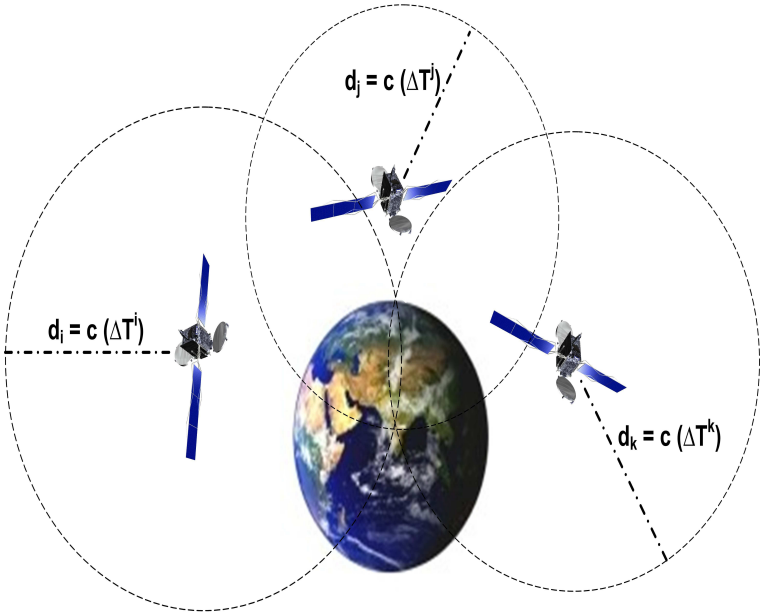


Fig. 5. GPS based location sensing and triangulation.

Figure 4 shows and example of a cell broadcast location sensing and triangulation method. The Location-Code (LC) for Base Station 1 can be constructed by the mobile phone as,

`LC = 23415300564404719, where MCC = 234, MNC = 15, LAC = 30056, Cell-ID = 4404719`

According to [74], if the locations of the towers and base stations are known then the most probable position (x, y) of the mobile phone can be calculated based on the received signal strength, to be either one of the two values represented by equations (6) or (7) as derived below. In Figure 4, (x_1, y_1) and (x_2, y_2) represents the coordinates of two base stations. Their mean distances to the mobile phone are d_1 and d_2 respectively. The distance between the two base stations, d_{bts}, can be derived as,

$$d_{bts} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{1}$$

$$\left.\begin{aligned} l_1 &= (d_1 + d_2) - d_{bts}, \\ l_2 &= \sqrt{d_2^2 - l_1^2} \end{aligned}\right\} \tag{2}$$

$$sin(a) = \frac{(y_2 - y_1)}{d_{bts}} \tag{3}$$

$$cos(a) = \frac{(x_2 - x_1)}{d_{bts}} \tag{4}$$

The point where l_1 and l_2 meets P, (x_p, y_p), can be obtained as,

$$\left.\begin{aligned} x_p &= x_2 - l_1(cos(a)), \\ y_p &= y_2 - l_1(sin(a)) \end{aligned}\right\} \tag{5}$$

Then we get,

$$\left.\begin{aligned} x &= x_2 - l_1(cos(a)) - l_2(sin(a)), \\ y &= y_2 - l_1(sin(a)) + l_2(cos(a)) \end{aligned}\right\} \tag{6}$$

$$\left.\begin{aligned} x &= x_2 - l_1(cos(a)) + l_2(sin(a)), \\ y &= y_2 - l_1(sin(a)) - l_2(cos(a)) \end{aligned}\right\} \tag{7}$$

This calculation can either be performed by the mobile phone, the mobile network operator or a third party location services provider. Unfortunately, the determination of location from the received power of cell broadcasts is known to lack precision and consistency due to the spatial and temporal variations of the radio environment. As a measure for determining relative separation between devices it should work over long distances (with respect to the cell radii), although its effectiveness over shorter ranges in uncontrolled physical environments cannot be relied upon. Therefore it may provide a strong separation indicator for long range relaying via GPRS and a weaker indicator for a shorter range relay bearer such as Bluetooth. Similarly, a cell diameter can be quite large so if using only Cell-ID a relay attack mounted from within the same cell would be difficult to detect, and an efficient countermeasure would ideally need to obtain parameters from multiple neighbouring cells to improve the location resolution. When both parties are connected to different mobile network operators the Cell-IDs and LACs could also vary. Hence, care needs to be taken in design of the application that generates the location information in this way and its verification. There are also viable security threats to mobile location information integrity, such as false base station attacks.

b) GPS Based Location Sensing: The Global Positioning System (GPS) is a navigational system based on earth-orbiting

satellites and provides location information around the globe. GPS finds applications in many fields such as transportation, aviation and shipping. The GPS system is based on 24 satellites in six different orbital-paths. The satellites and the receivers are synchronised with high precision clocks which is used to estimate the distance between them and the receiver. A GPS receiver requires an unobstructed line-of-sight to at least four or more satellites in order to calculate its three-dimensional position (latitude, longitude, altitude). However, with three satellites in view the receiver is able to compute its two-dimensional location (latitude, longitude) [75]. Figure 5 illustrates the GPS based location sensing and triangulation method. In the figure, d_i represents the distance of i^{th} satellite from Earth. c is the speed of light (299,792,458 m/s). ΔT is the time difference of signal sent from the satellite and received on the Earth.

$$\left.\begin{aligned} {d_i}^2 &= (x^i - x)^2 + (y^i - y)^2 + (z^i - z)^2, \\ {d_j}^2 &= (x^j - x)^2 + (y^j - y)^2 + (z^j - z)^2, \\ {d_k}^2 &= (x^k - x)^2 + (y^k - y)^2 + (z^k - z)^2 \end{aligned}\right\} \quad (8)$$

By solving (8), and after error corrections we get, $[X, Y, Z]$ where X = longitude, Y = latitude, and Z = altitude.

An increasing number of mobile phones contain Global Position System (GPS) receivers. GPS is a reliable system for determining the location of the phone. Most mobile phone platforms allow access to GPS location information through public APIs. The GPS receivers can be categorised broadly as follows:

- Integrated/Autonomous GPS: Here the GPS receiver is embedded within the mobile device. The most accurate location sensing, is achieved when the receiver can receive the satellite transmissions clearly without any obstruction.
- Assisted GPS: In Assisted GPS (A-GPS), direct satellite observation and a network "server" is used to generate accurate position information. A-GPS that is network assisted could faster compared to integrated GPS, and perform better in poor signal conditions. A-GPS devices cannot work outside the mobile network coverage region as it needs to be connected to the servers.
- External GPS: An external GPS is a physically separate device that can be linked to a mobile device over interfaces such as Bluetooth or USB.

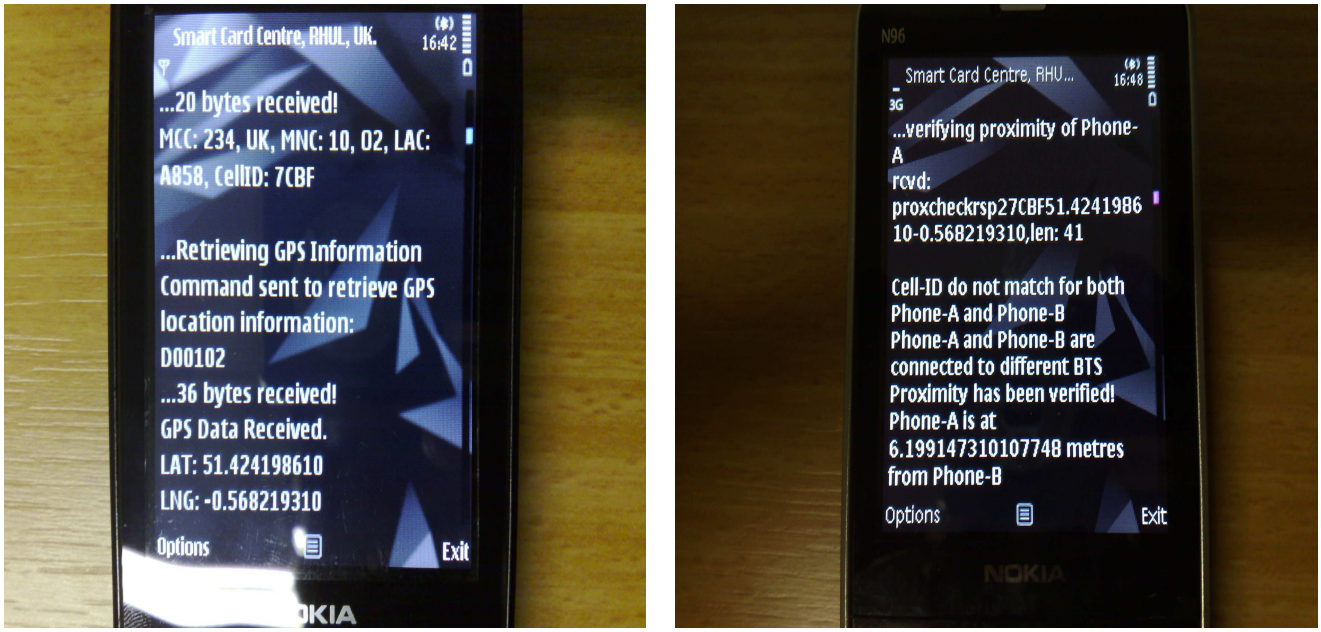
Deriving location information from GPS also has some disadvantages. A mobile phone would need to be equipped with a GPS receiver and the accuracy of integrated receivers is greatly diminished when operating indoors, where you would expect most transactions to take place.

c) Other Location Mechanisms: There are a number of other method for determining device location, even FM radio technology has been proposed as a localization technology [76], although these are not as directly linked to mobile phones as Cell-ID and GPS. All that is required is that two devices can with some certainty verify that they are in close proximity to each other. This only requires the devices to be aware of their relative location, i.e. where they are with respect to each other, so absolute location information is not needed. More peripherals for wireless sensing and communication are being integrated in mobile phones and it is possible that these could eventually be used to construct proximity proofs. There are several proposals for how two devices can verify that they are in the same location. For example, in multi-channel protocols [77] the device associates additional media that is difficult to relay with the transaction, e.g. both devices can hear the same audio or are observing a picture known to be in the area (one of the device could generate the audio or picture). An area could also be associated with a location 'dongle' or beacon [78], and if both devices can observe this dongle during the transaction they are likely to be in a specific location. Although these proposals are interesting we are of the opinion that a countermeasure should ideally use the location information already available on the devices in question. We therefore implemented a proof-of-concept proximity location application using GPS and mobile network Cell-ID information.

d) Practical Proof-of-Concept Implementation: When a transaction uses location information as an additional security metric, it could potentially detect relay attacks. For example, a device would simply incorporate a location signature into the transaction data, which could be checked by the recipient and compared to its own location in order to verify device proximity. The location information may be generated by using any of the methods discussed previously. Based on this information a location signature record could be constructed as follows:

```
<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>30056</lac><cellid>4404719</cellid>
</location information>
</signature>D09A3B57D49CA179</signature>
</location proof>
```

A simple proof-of-concept countermeasure application was implemented in order to demonstrate the feasibility of retrieving location and verifying proximity between two transacting parties. The mobile applications were developed and installed on two Nokia N96 mobile phones ("Prover" and "Verifier") that are based on Symbian S60 3^{rd} Edition FP1 platform [79]. For each mobile phone a native Symbian C++ application was developed and installed that had access to restricted low-level APIs such as network, location, communication, and security APIs. The application was code signed according to [80] in order to allow access to the restricted APIs. A J2ME/MIDP 2.0 application implemented the Bluetooth API (JSR 82), proximity verification, and graphical user interface.



(a) Prover - Phone A

(b) Verifier - Phone B

Fig. 6. "Prover" and "Verifier" mobile phones computing proximity based on location information.

The proximity verification was performed based on the location information retrieved. Cell broadcast information was used to check whether the Prover and Verifier are connected to the same mobile cell. GPS co-ordinates were also retrieved and by using the Haversine method [81] the distance between Prover and Verifier was computed. Accurate GPS (by using integrated GPS) based location information was derived outdoors whereas location information using A-GPS was derived indoors. For both methods of location sensing, the J2ME application relied upon native Symbian application. Both mobile phones interacted with each other over Bluetooth communication. For instance, the Verifier would send a request to the Prover to reply with its location information, which is compared by the Verifier to its own location. An example message exchange between the Prover and Verifier is shown in Figure 6. Here the Cell-IDs of the two phones do not match, but the GPS information is sufficient to determine that the phones are actually in close proximity (approximately 6 m). If the phones were far apart, and the communication was relayed, the verifier would observe, from the location information integrated into the transaction data, that the legitimate prover is not within proximity, and the attack would be detected.

The disadvantage to using such location-based security in contactless systems, apart from the potential inconsistency of the radio environment and lack of precision, is that both parties need to be location aware. NFC-enabled handsets would be able to derive and verify location information but conventional contactless tokens would not benefit from this countermeasure. Fixed-location POS devices could potentially be programmed with its known location during installation.

2) Relay resistance at the communication layer: There are some aspects of the underlying communication processes that could be used to detect a relay attack. The NFC controller is responsible for all physical communication operations, such as anti-collision, token selection, communication parameter setup and data formatting for transmission. During anti-collision and token selection the hardware UID of the token is normally used. The legitimate device and the proxy-token should in theory therefore have different UIDs. If the transaction data is linked to a UID, the verifying recipient should observe that the UID in the data does not correspond to the UID of the device it is communicating with, thus detecting the relay. In some systems this countermeasure is possible but increasingly contactless tokens are transmitting random identifiers during anti-collision as a privacy preserving/untraceability measure [65]. Furthermore, the Blackberry emulation API allows the UID to be set by the application, and although our attempt to get this to work within the token emulation profile we used was not successful, we assume that this functionality will be available. Binding the transaction data to a UID would therefore be of little use, in cases of random identifiers, or provide no advantage in terms of security if the attacker can set the UID of the proxy-token.

The emulation application passes any data to be transmitted to the controller, which is responsible for framing and error correction as needed. For example, in ISO 14443 each data byte is transmitted along with an odd parity bit and a 16-bit CRC is appended to the message. When using ISO 14443-4 formatted APDUs these parity bits and CRC bits are sent in plaintext and removed from the message by the recipient. Some contactless tokens only partially adhere to standards.

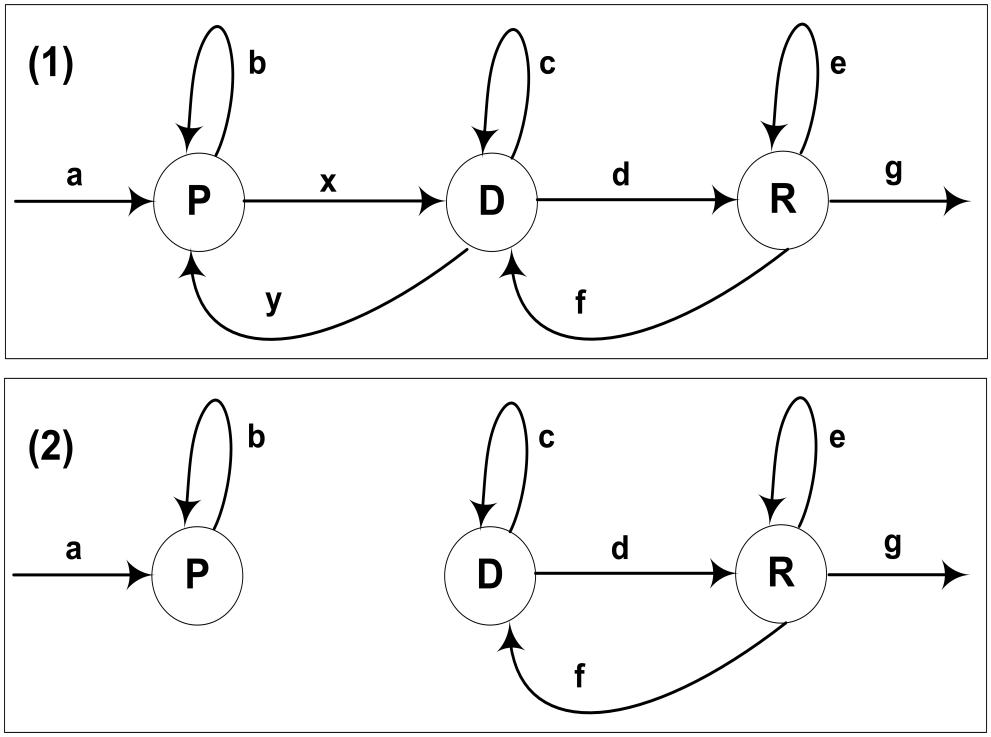


Fig. 7. State diagrams: (1) emulation API routine showing relay, and (2) relay protection.

One such proprietary product, Mifare Classic [82] also encrypts the parity and CRC bits. It is not thought possible to relay this communication with our relay implementation as it is not possible to retrieve the encrypted parity and CRC bits from the controller (it is mostly likely that the controller will discard the message since what it considers to be plaintext parity and CRC bits will not match to the rest of the data). As a result, the message from the legitimate token cannot be captured or transmitted to the reader in its true form and the relay will fail. The Blackberry emulation API does allow for emulating and reading Mifare Classic tokens but in this case the attacker would need the right key to interact with the legitimate card and reader. However, proprietary tokens, especially Mifare Classic, have often been shown to contain significant security vulnerabilities [83], [84] so using such a product purely as a relay attack countermeasure is not at all recommended.

3) Application Restrictions: One immediate solution to prevent the discussed attacks is to remove the "soft-SE" and the associated emulation API altogether. However, this may not be acceptable due to the benefits associated with a open development philosophy. The contactless applications based on "soft-SE" can utilise fast processing and large memory capacity of the mobile phone. The "soft-SE" approach allows more flexibility and control for the end-user to manage emulated contactless applications, and is independent to the mobile network, or specific Trusted Service Manager (TSM) controls. An intermediate solution is to strengthen the control that the run-time environment has over applications implementing the emulation API. The state diagram of soft-SE emulation API routine is shown in Figure 7.(1). It illustrates the Process (P), Delay (D), Relay (R) and some possible state transitions. The core of the emulation API is processCommand() function (as shown below), and is responsible for handling the command messages from the contactless reader.

```
net.rim.device.api.io.nfc.emulation
VirtualISO14443Part4TargetCallback
byte[] processCommand(byte[] command)
{
//handle APDU commands
}
```

The parameter 'command' contains the ISO 14443-4 command sent by the external contactless reader. The function returns a byte array containing the response to be sent to the external reader. In order to implement the relay attack, the command was initially captured and sent over the relay bearer. The application then enters a delay state until the response is available to be returned to the reader. As shown in Figure 7.(2) any application that has entered state P (received command from a reader) should not be allowed to execute arbitrary delays (state D), or in fact be allowed to invoke other communication calls to transmit the command or facilitate the reception of the relay response (state R). Alternatively, there could also be additional restrictions on the use of the API, such as not allowing the application identifier (AID) to be set to a value reserved for security sensitive applications, unless additional developer verification has taken place. This could potentially

be incorporated into existing application signing processes. We have considered the possibility that such a system could be implemented using application permissions. Applications executing on mobile platforms need to be granted, normally by the user, permissions for performing certain functions or for having access to certain data. Permissions in their current form, unfortunately, does not seem to be a suitable vehicle for this countermeasure. None of the permissions on NFC platforms we worked with, Android 2.3 and Blackberry 7, contain the type of restriction we need to implement this scheme. Also, as permissions are largely controlled by the user an attacker could simply grant his attack application, running on his mobile device, the required permission.

V. Conclusion

In this paper we described the first generic practical implementation of a contactless relay attack using only NFC-enabled mobile phones and software applications. We were able to build a passive proxy-token, a proxy-reader and a suitable communication channel between the proxies by using only publicly available platform APIs. Our relay attack demonstrates a reduced complexity of attack as it did not require special hardware. The attack implementation required no unlocking of devices or secure elements, no hardware or software modification to the phone platform, and minimal knowledge of the data that was to be relayed. Neither was there any need to access the mobile network or any related services, and we utilised devices of a form factor accepted by merchants. The attack implementation was application independent so would work against a number of conventional contactless systems. For example, we experimentally verified that the implementation work against both test payment and e-passport systems. The attack therefore holds implications for all contactless systems and can be implemented against any system using NFC or compatible technology, with a few exceptions as discussed in Section IV-B2. Research work on relay attacks, preceding this paper, have often been dismissed by system implementers as a complicated attack that is unlikely to be used in the real world. The 'software-only' nature of this relay attack implementation increases the likelihood of it being used in practice (e.g. an attacker simply downloads the applications), and so represents a potential threat to real-world systems. This paper effectively disproves the opinion that relay attacks are complex attacks that do not translate to an effective real-world threat as argued in [31], [32].

The effectiveness and ease of the attack means that ticketing, payment (credit card and mobile wallets) and access control application need to be hardened against relay attacks. Currently, virtually no deployed products implement relay resistant mechanisms, with the exception of NXP's new Mifare Plus smart card and that has up to now only seen limited deployment and it is unknown how many systems that do use Mifare Plus actually take advantage of this security service. There are a number of countermeasures in literature that are considered effective against relay attacks, and mobile platforms have much possibilities when compared to conventional smart cards. We discussed several of these potential countermeasures capable of mitigating such a relay attack in a mobile environment. The early results of this work and suggested countermeasures were shared with relevant industry parties so that appropriate remedial measures could be considered such as changes to standardisation and implementation choices. The use of SEs that may be misused as development attack platforms also raises interesting questions regarding SE architecture and application management. Our future work will investigate whether a security framework for "soft-SEs" could be implemented that promotes the open development platform philosophy while at the same time protecting against 'malicious' applications misusing the platform.

Acknowledgements

The authors would like to acknowledge the many helpful suggestions of anonymous reviewers and the participants of RFIDsec Asia 2012 Workshop on earlier versions of this paper. We would like to thank Giesecke & Devrient (GmbH) and Comprion (GmbH) for providing equipment support. We would also like to thank secunet Security Networks AG for providing eMRTD reader software. We also thank the editors of this Journal.

References

[1] EMV. EMV Contactless Specifications for Payment Systems, EMV Communication Protocol Specification, Version 2.0, August, 2007, http://www.emvco.com/.
[2] Visa® payWave™, http://www.visaeurope.com/en/.
[3] MasterCard® PayPass™, http://www.paypass.com/.
[4] International Civil Aviation Organization (ICAO), Document 9303 Machine Readable Travel Documents (MRTD), Part 1: Machine Readable Passports, 2005.
[5] ISO/IEC 7501, Identification Cards - Machine Readable Travel Documents, October, 2005.
[6] Transport for London Oyster Card, TFL, UK, https://oyster.tfl.gov.uk/.
[7] Octopus, Octopus Cards Limited, Hong Kong. http://www.octopus.com.hk/.
[8] ISO/IEC 15693, Identification cards – Contactless integrated circuit cards – Vicinity cards, http://www.iso.org/.
[9] HID Global, Access Control Solutions, http://www.hidglobal.com/main/id-cards/iclass-standard-credentials/.
[10] G. P. Hancke, Practical eavesdropping and skimming attacks on high-frequency RFID tokens, Journal of Computer Security - 2010 Workshop on RFID Security (RFIDSec'10 Asia), Volume 19, Issue 2, pp 259–288, April, 2011.
[11] I. Kirschenbaum, and A. Wool, How to Build a Low-Cost, Extended-Range RFID Skimmer, In Proceedings of 15th USENIX Security Symposium, pp 43-57, August, 2006.
[12] D. Oswald, T. Kasper, and C. Paar, Side-Channel Analysis of Cryptographic RFIDs with Analog Demodulation, Springer LNCS, In Proceedings of RFIDSec 2011, Northampton, USA, 2011.
[13] S. C. Bono, M. Green, A. Stubblefield, A. Juels, A. D. Rubin, and M. Szydlo, Security Analysis of a Cryptographically-Enabled RFID Device, In Proceedings of Usenix Security Symposium, 2005.
[14] A. Juels, D. Molnar, and D. Wagner, Security and Privacy Issues in E-passports, In Proceedings of First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM '05), IEEE Computer Society, Washington, DC, USA, pp 74–88, 2005.
[15] T. S. Heydt-Benjamin, D. V. Bailey, K. Fu, A. Juels, and T. OHare, Vulnerabilities in first-generation RFID-enabled credit cards, In Proceedings of Financial Cryptography and Data Security, pp 1–22, February, 2007.
[16] G. P. Hancke, K. E. Mayes, and K. Markantonakis, Confidence in Smart Token Proximity: Relay Attacks Revisited, Elsevier Computers & Security, Vol. 28, Issue 7, pp 615–627, October, 2009.

[17] Near Field Communication (NFC) Forum, http://www.nfc-forum.org/.

[18] Nokia C7, Nokia, http://www.nokia.co.uk/gb-en/products/phone/c7-00/.

[19] BlackBerry Bold 9900/9930, RIM, http://uk.blackberry.com/.

[20] Samsung GT-I9020 (Google Nexus S), Samsung, http://www.samsung.com/.

[21] Google Wallet, Google Inc, http://www.google.com/wallet/.

[22] Orange Quick Tap, Orange, France Telecom, http://shop.orange.co.uk/mobile-phones/contactless/.

[23] G. Madlmayr, J. Langer, C. Schaffer, and J. Scharinger, NFC Devices: Security and Privacy, In Proceedings of 3rd International Conference on Availability, Reliability and Security, Barcelona, 2008.

[24] L. Francis, G. P. Hancke, K. E. Mayes, and K. Markantonakis, Potential Misuse of NFC Enabled Mobile Handsets with Embedded Security Elements as Contactless Attack Platforms, In Proceedings of 1st Workshop on RFID Security and Cryptography (RISC'09), in conjunction with ICITST 2009, pp 1–8, November, 2009.

[25] R. Anderson, RFID and the Middleman, Conference on Financial Cryptography and Data Security, pp 46–49, December, 2007.

[26] Z. Kfir, and A. Wool, Picking Virtual Pockets using Relay Attacks on Contactless Smartcard Systems, In Proceedings of IEEE/CreateNet SecureComm, pp 47–58, 2005.

[27] G. P. Hancke, Practical Attacks on Proximity Identification Systems (short paper), In Proceedings of IEEE Symposium on Security and Privacy, pp 328–333, May, 2006.

[28] Libnfc, Public Platform Independent Near Field Communication (NFC) Library, http://www.libnfc.org/documentation/examples/nfc-relay/.

[29] M. Weiss, Performing Relay Attacks on ISO 14443 Contactless Smart Cards using NFC Mobile Equipment, Master Thesis, Technischen Universitat Munchen, Munich, Germany, 2010.

[30] L. Francis, G. P. Hancke, K. E. Mayes, and K. Markantonakis, Practical NFC Peer-to-Peer Relay Attack using Mobile Phones, In Proceedings of 6th Workshop on RFID Security (RFIDSec'10), LNCS, Springer-Verlag, June, 2010.

[31] W. Knight, The price of love, Elsevier Infosecurity, pp 30-33, Vol. 5, Iss. 1, January, 2008.

[32] M. Roberti, Are RFID-Enabled Credit Cards Safer Than Magstripe Cards? RFID Journal, September, 2010, http://www.rfidjournal.com/blog/entry/7870/.

[33] 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/.

[34] ISO/IEC 21481 (ECMA-352), Information technology – Telecommunications and information exchange between systems – Near Field Communication Interface and Protocol-2 (NFCIP-2), 2005, http://www.iso.org/.

[35] ISO/IEC 14443, Identification cards - Contactless integrated circuit cards - Proximity cards, http://www.iso.org/.

[36] FeliCa, http://www.sony.net/Products/felica/.

[37] Bluetooth Core Specification Version 2.1.+ EDR, Volume 2, July, 2007.

[38] Essentials for Successful NFC Mobile Ecosystems, NFC Forum White Paper, October, 2008.

[39] Sun Microsystems, Java Card Platform Specification v2.2.1, http://java.sun.com/products/javacard/specs.html.

[40] NXP, Java Card Open Platform, http://www.nxp.com/.

[41] Global Platform, Card Specification v2.1.1, http://www.globalplatform.org/.

[42] NXP Semiconductor, Mifare Standard Specification, http://www.nxp.com/.

[43] 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/.

[44] SD Card Association, http://www.sdcard.org/.

[45] Candidate Technical Specification: Signature Record Type Definition. NFC Forum. October 2009.

[46] NFC Data Exchange Format (NDEF), NFC Forum Technical Specification, Rev. 1.0, Jul. 2006.

[47] M. Roland, J. Langer, and J. Scharinger, Security Vulnerabilities of the NDEF Signature Record Type, In Proceedings of Third International Workshop on Near Field Communication, pp. 65-70, 2011.

[48] C. Mulliner, Vulnerability Analysis and Attacks on NFC-Enabled Mobile Phones, Proceedings of International Conference on Availability, Reliability and Security, ARES 2009, pp 695–700, March, 2009.

[49] R. Verdult, and F. Kooman, Practical Attacks on NFC Enabled Cell Phones, In Proceedings of Third International Workshop on Near Field Communication, pp. 77-82, 2011.

[50] J. H. Conway, On Numbers and Games, Academic Press, 1976.

[51] Y. Desmedt, C. Goutier, and S. Bengio, Special Uses and Abuses of the Fiat-Shamir Passport Protocol, Advances in Cryptology (CRYPTO), Springer-Verlag LNCS 293, pp 21, 1987.

[52] Y. C. Hu, A. Perrig, and D. B. Johnson, Wormhole Attacks in Wireless Networks, IEEE Journal on Selected Areas in Communications (JSAC), pp 370–380, 2006.

[53] G. P. Hancke, and M. G. Kuhn, An RFID Distance Bounding Protocol, In Proceedings of IEEE CreateNet SecureComm, pp 67–73, September, 2005.

[54] A. Francillon, B. Danev, and S. Capkun, Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars, In Proceedings of Network and Distributed System Security Symposium (NDSS), 2011.

[55] RFID IO Tools, http://www.rfidiot.org/.

[56] S. Drimer, and S. J. Murdoch, Keep your enemies close: distance bounding against smartcard relay attacks, In Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium (SS'07), USENIX Association, Berkeley, CA, USA, 2007.

[57] Oracle/Sun Microsystems, JSR-000082 Java API for Bluetooth 2.1, http://jcp.org/aboutJava/communityprocess/final/jsr082/index.html.

[58] Oracle/Sun Microsystems, JSR-000118 Mobile Information Device Profile 2.0, http://jcp.org/aboutJava/communityprocess/final/jsr118/index.html.

[59] Oracle/Sun Microsystems, JSR-000257 Contactless Communication API 1.0, http://jcp.org/aboutJava/communityprocess/final/jsr257/index.html.

[60] Nokia 6131 NFC SDK, http://www.forum.nokia.com/.

[61] Oracle/Sun Microsystems, Java Code Signing for J2ME, http://www.oracle.com/technetwork/java/javase/tech/getcodesigningcertificate-361306.html.

[62] Uncovered: The hidden NFC potential of the Google Nexus S and the Nokia C7, Near Field Communications World, 13 February 2011, http://www.nfcworld.com/2011/02/13/35913/.

[63] Research In Motion Limited, BlackBerry v7.1 NFC API, http://www.blackberry.com/developers/docs/7.1.0api/.

[64] Research In Motion Limited, Java Code Signing Keys, http://us.blackberry.com/developers/javaappdev/codekeys.jsp.

[65] MIFARE DESFire EV1 contactless multi-application IC, Product short data sheet, Rev. 02, 6 March, 2009, http://www.nxp.com/acrobat_download/datasheets/MF3ICD21_41_81_SDS_2.pdf.

[66] S. Brands, and D. Chaum, Distance Bounding Protocols, In Proceedings of Advances in Cryptology, EUROCYPT '93, Springer-Verlag LNCS 765, pp 344–359, May, 1993.

[67] J. Clulow, G. P. Hancke, M. G. Kuhn, and T. Moore, So Near and Yet So Far: Distance-Bounding Attacks in Wireless Networks, In Proceedings of European Workshop on Security and Privacy in Ad-Hoc and Sensor Networks (ESAS), Springer-Verlag LNCS 4357, pp 83–97, September 2006.

[68] G. P. Hancke, and M. G. Kuhn, Attacks on Time-of-Flight Distance Bounding Channels, In Proceedings of First ACM Conference on Wireless Network Security (WISEC'08), pp 194–202, March, 2008.

[69] G. P. Hancke, Design of a Secure Distance-Bounding Channel for RFID, Elsevier Journal of Network and Computer Applications, Volume 34, Issue 3, pp 877–887, May, 2011.

[70] Virginia Tech Cybersecurity Breakthrough Keeps Sensitive Data Confined in Physical Space, October 17, 2011, http://www.vtnews.vt.edu/articles/2011/10/101711-outreach-cybersecurephones.html.

[71] L. Francis, K. E. Mayes, G. P. Hancke, and K. Markantonakis, A location based security framework for authenticating mobile phones, In Proceedings of 2nd International Workshop on Middleware for Pervasive Mobile and Embedded Computing (M-MPAC'10). ACM, 2010.

[72] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, and D. Boneh, Location Privacy via Private Proximity Testing. Proceedings of Network and Distributed System Security Symposium (NDSS), 2011.

[73] F. Park, C. Gangakhedkar, and P. Traynor, Leveraging Cellular Infrastructure to Improve Fraud Prevention. Proceedings of Annual Computer Security Applications Conference (ACSAC), 2009.

[74] S. Y. Willassen, Positioning a Mobile Station, 1998, http://www.willassen.no/msl/node6.html.

[75] B. W. Parkinson, and J. S. James, Global Positioning System: Theory and Practice. Volumes I and II., Washington DC, American Institute of Aeronautics and Astronautics Inc, 1996.

[76] A. Papliatseyeu, N. Kotilainen, O. Mayora, and V. Osmani, FINDR: Low-Cost Indoor Positioning Using FM Radio, In Proceedings of Mobileware 2009, pp 15–26, 2009, http://dx.doi.org/10.1007/978-3-642-01802-2_2.
[77] F. Stajano, F. L, Wong, and B. Christianson, Multichannel protocols to prevent relay attacks, Proceedings of Financial Cryptography, 2010.
[78] A. Studer, and A. Perrig, Mobile user location-specific encryption (MULE): using your office as your password, In Proceedings of ACM Conference on Wireless Network Security (WiSec), 2010.
[79] S60 3rd Edition Feature Pack 1, S60 Platform SDKs for Symbian OS, for C++, Nokia, 2006, http://forum.nokia.com/.
[80] Symbian Foundation Ltd. Symbian Signed, https://www.symbiansigned.com/.
[81] R. W. Sinnott, Virtues of the Haversine, Sky and Telescope, vol. 68, no. 2, 1984.
[82] K. Nohl, D. Evans, Starbug, and Plötz, H., Reverse-engineering a cryptographic RFID tag. In Proceedings of USENIX Security 2008, pp 185–193, 2008.
[83] F. D. Garcia, G. Gans, R. Muijrers, P. Rossum, R. Verdult, R. W. Schreur, and B. Jacobs, Dismantling MIFARE Classic, In Proceedings of European Symposium on Research in Computer Security (ESORICS'08), volume 5283, LNCS, pp 97–114, Springer, 2008.
[84] F. D. Garcia, P. Rossum, R. Verdult, R. W. Schreur, and B. Jacobs, Wirelessly Pickpocketing a Mifare Classic Card, In Proceedings of IEEE Symposium on Security and Privacy, pp 3–15, 2009.

