N-Guard: a Solution to Secure Access to NFC tags

Roberto Di Pietro*, Gabriele Oligeri, Xavier Salleras‡ and Matteo Signorini§

*College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar
‡Pompeu Fabra University, Barcelona, Spain
§Nokia Bell Labs, Paris, France

***Abstract*—In this paper we propose *N-Guard*: a portable, effective, and efficient solution to thwart contactless skimming of NFC cards. Our solution enables an NFC-compliant smartphone to protect the user's cards, preventing the adversary from harvesting the cards' data. Moreover, we also introduce a fine grained access control mechanism, allowing the user to discriminate between NFC cards that can be opportunistically queried and sensitive ones that can be read only under the strict permission of the owner.**

We implemented a proof-of-concept of *N-Guard* for Android OS and tested it under several digital skimming scenarios showing its effectiveness in thwarting unauthorized access attempts. Moreover, we also measured the consumption of *N-Guard* and proved that its energy consumption is negligible. Further, it is worth noting that *N-Guard* requires neither any specific modification to the NFC standard, nor any change on users behavior. Finally, through some empirical evidence, we show *N-Guard* to be effective even when the interaction between the NFC tags and the reader is driven by proprietary protocols (e.g. Mastercard). All the reported results, having being developed over an NFC-enabled credit-card use case, are general and applicable to all NFC tags.

I. Introduction

Since the introduction of Radio-Frequency Identification (RFID) [1], this technology has led into the invention of new standards and derived technologies. One of them is Near-field communication (NFC) that, according to NFC Forum , enables simple and safe two-way interactions between electronic devices, allowing consumers to perform contactless transactions, access digital content, and connect electronic devices with a single touch.

One of the most important and common uses of NFC are contactless payments. According to MasterCard , *Charg-It* was the first bank card introduced in 1946 by John Biggins. Since then, payment cards have significantly evolved from embedding magnetic stripes, chips, and finally RFID transducers with the to-days contact-less cards. Over time, several security measures have been designed to protect online transactions, such as the card verification value (also known as CVV) or the Meg-Stripe standard [2]. An interesting observation arises from the fact that, while in the past frauds involving credit and debit cards were complex to achieve since magnetic stripe and Chip & PIN cards required fraudsters to have direct physical access to them, nowadays attacks can be perpetrated at a devastating scale, since there is no need to have physical access to the card. Indeed, the current trend of including NFC technology into bank cards (contact-less cards) has simplified attacks such as *skimming* and *relaying* [3], [4]. Therefore, sensitive information such as credit card's number, expiration date and card-holder's name can be easily stolen (and subsequently exploited for a number of malicious activities, such as micro payments [5], or for supporting identity theft).

An early attack to EMV *Chip & PIN* cards was performed by Murdoch et al. [6], who discovered a weakness that allows to bypass the secret code and PIN verification for off-line transactions on certain cards. Subsequently, authors in [7] pointed out a weakness on random numbers generation by EMV terminals showing how transaction authorizations could be pre-computed by having access to a real credit card and then re-played with a cloned one. The attack revealed by [7] has been further refined and enhanced by [8] who discovered how to unleash the pre-play attack not just for specific terminals but, more generally, against contact-less cards. Unlike the previous attacks, authors in [8] showed how to clone credit cards and to compute transactions' pre-authorization codes without knowing in advance both the amount of money to be spent and the initialization state of the terminal's random number generator. Even though this attack is still limited by the maximum amount that can be authorized with a PIN-less contact-less transaction, it showed that credit card transaction security is based on the payment protocol which is usually proprietary and (not having been subject to public scrutiny) subject to vulnerabilities.

More in general, contact-less cards technology, while being more and more intuitive and versatile, is natively affected by eavesdropping and radio skimming attacks [9]. Indeed, NFC systems use near-field magnetic coupling to implement communication channels—the advertised operational range of these channels being in the order of a few centimeters. Indeed, the large majority of the implemented systems assume that the communication channel is location limited and therefore relatively secure. This is clearly not true for a few scenarios at least, e.g., crowded environments, where an adversary can massively collect information by walking around and radio-skimming all the bank-cards from the people in the neighborhood. This is becoming easier and easier, in particular today, since the large majority of smartphones feature NFC readers and there are plenty of applications enabling them to systematically read data from NFC-enabled cards. While being very efficient, unfortunately the above attack has so far no effective solution. Indeed, only a few researchers focused

Personal copy of the authors - paper accepted version. Cite as: *R. Di Pietro, G. Oligeri, X. Salleras, M. Signorini, N-Guard: a Solution to Secure Access to NFC tags, in Proc. of IEEE Conference on Communications and Network Security (CNS), Bejing, Jun., 2018.*

on improving the security of contact-less cards and they only proposed a theoretical framework for mitigating attacks on RFID-based systems [10]; to the best of our knowledge, no implementations supported by real measurements of an effective solution has been proposed so far.

Contributions. This paper proposes a novel technique to prevent unauthorized card skimming by exploiting NFC-enabled smartphones. Driven by the the fact that most people bring NFC cards and smartphones altogether, we propose an NFC firmware implementation to be deployed on the user's smartphone that detects unauthorized readings of some pre-defined cards (e.g. bank cards), and it reacts by blocking the communication between the card and the attacker. This way, by assuming the user keeps his sensitive cards together with the smartphone (or close to it in his own pocket), we enable the smartphone to continuously monitor the NFC channel, and reacting to unauthorized readings by preventing the cards to broadcast their sensitive information. Moreover, our solution features several properties:

- *Software based*: *N-Guard* only requires minor modifications to the NFC firmware of the smartphone to enable it to monitor and block all the unauthorized transactions involving the NFC cards;
- *Universal*: *N-Guard* is compliant with all the NFC cards independently of the standard implementation. In particular, *N-Guard* can protect the most popular standards for NFC communications such as the ISO 14443-3—involving MIFARE cards, bio-metric passports, German identity cards, and the vast majority of bank payment cards;
- *Configurable*: *N-Guard* prevents the unauthorized readings of only a pre-defined set of NFC cards while enabling full access to not sensitive ones, e.g., metro or bus cards;
- *Efficient*: *N-Guard* requires only one NFC communication. Indeed, it exploits one single command to stop the communication between the victim and the attacker.

Paper organization. This paper is organized as follows. Section II reviews the current state of the art as for the NFC attacks and recent proposed solutions, while Section III presents the building blocks of the NFC technology that enable our solution. Section IV introduces the threat model, the definitions, and the entities involved in our use case scenario. Section V depicts the implementation details of our solution, while Section VI provides the details of our experimental tests. Finally, some concluding remarks are reported in Section VII.

II. Related work

NFC Cards authentication and security. One of the first and most effective attack on NFC payment cards appeared in [4]. Authors proposed a relay attack by reading the content of a credit card with one smartphone and forwarding the stolen data via a proxy communication channel (Bluetooth) to another smartphone. Eventually, the second smartphone that is acting as a proxy token, mimics the original credit card on a contactless reader (PoS). Authors provide a concrete proof of the attack feasibility using a Blackberry and a Nexus S mobile phone. They also suggested a few countermeasures such as timing, distance bounding and location estimation by leveraging the GPS functionality of the smartphone

In [11], authors discussed the feasibility of both passive and active relay attacks on RFID based smart tokens and they subsequently evaluate several techniques that might make the technology robust to relay attacks such as introducing additional verification techniques.

Another relay attack has been proposed in [12]. Authors investigated the effectiveness of combining the "leech", i.e., the device that steals the information from the target, and the "ghost", i.e., the device that is far away and leverages the stolen information by the leech to get the authorization to a specific service. Authors highlighted how relay-based attacks can be effective in several context such as payment cards, electronic passports, authentication badges, etc. Mostly important, authors pointed out as the attack is also effective even in the case of strong authentication and encryption algorithms: modern communication channels can make the communication delay between the leech and the ghost as negligible.

A prototype antenna for long distance queries has been proposed in [13]. Authors confirmed experimentally what it has been previously predicted by [12]: RFID tags can be queried by a distance of about half a meter as also acknowledged by the theoretical analysis of [14] showing that for a desired range r, the optimal antenna diameter d is $d \approx r$. This can be also considered an upper bound limit on the portability of a device by an adversary: bigger antennas might be difficult to hidden and the adversary might be easily identified.

NFC architecture security. Authors in [15] proposed a physical layer based solution to secure NFC communications from eavesdropping. The proposed solution affects the signals on the initiators to hide the communications, but does not require any changes to the targets. In particular, authors addressed a practical problem of synchronization offset between two NFC terminals, which can actually be exploited by an eavesdropper to compromise the transmitted bits.

Authors in [16] proposed a key-agreement protocol exploiting NFC full-duplex capability. The proposed solution exploits two devices to send random bits to each other simultaneously without strict synchronization or perfect match of amplitude and phase. The method randomly introduces synchronization offset and mismatch of amplitude and phase for each transmitted bit in order to prevent a passive attacker from determining the generated key. Finally, a shared bit can be established when the two devices send different bits.

Authors in [17] analyze and conducted security testing on NFC-enabled mobile phones based on reader/writer operating mode in a peer-to-peer fashion manner to find vulnerabilities. The author use Denial-of-Service attack methods for attacking NFC-enabled mobile phones by using two semantic levels of distributed denial-of-service attack techniques.

Authors in [18] presents a prototype of a hardware module called Pay-Cloak, which could be used as a back cover for

an NFC-enabled smartphone. The module communicates via Bluetooth with an Android application installed in the smartphone after verifying the user's identity through a capacitive fingerprint sensor. The application processes both merchant point-of-sale (PoS) payments using quick response (QR) codes and peer-to-peer (P2P) payment using NFC. As Pay-Cloak, other solutions such as EnGarde [19] and nShield [20] present innovative hardware modules but their solution focus more on the efficiency of their NFC jamming approach.

An overview about security requirements and constraints for NFC devices is proposed by [21]. Authors discussed how to storing payment keys and executing payment applications on mobile phones via near-field communication at the point of sale (PoS) and other criteria such as hardware requirements, availability, management complexity, and performance.

Some prudent engineering practices and recommendations to follow, together with typical mistakes to avoid, when designing new ultra-lightweight authentication protocols are proposed by [22]. Their work can help, as a sanity check, designers of RFID, NFC, and sensor networks based security solutions to improve the security, reliability, and longevity of ultra-lightweight authentication protocol designs.

III. NFC Technology: Building blocks

In this section we introduce the NFC technology and we discuss the details of the standard ISO 14443 that enables our solution. The Near Field Communication (NFC) represents a set of standards for near-field radio communications including ISO 14443[1] and FeliCa[2]. The main purpose of these standards is to enable and regulate the interactions among mobile devices when they are brought within a short distance. The NFC protocol stack contains different layers which can be grouped into the following macro categories: physical characteristics (A1), radio frequencies (A-2), initialization (A3) and finally data exchange (A4).

Our solution exploits only the initialization and activation section of the protocol stack, i.e. ISO 14443 A-3.

A. ISO 14443 A-3 details

ISO 14443 A-3 describes how devices interact among each others at the protocol level by regulating the interactions among readers and tags. Tags supporting the ISO 14443 A-3 can belong to two types: *A* or *B*. Both of them communicate by leveraging the 13.56 MHz frequency (RFID HF). However, although they use the same transmission protocol, they differ in the modulation methods, coding schemes and protocol initialization procedures. It is important to highlight that our solution has been tested for both tags (A and B) and, more in general, it can also be applied to any protocol (even proprietary ones) that leverages on UIDs (Unique Permanent Identifiers) for NFC tag identification.

The ISO 14443 A-3 (A/B) is mainly characterized by the following messages:

[1]https://en.wikipedia.org/wiki/ISO/IEC_14443

[2]https://en.wikipedia.org/wiki/FeliCa

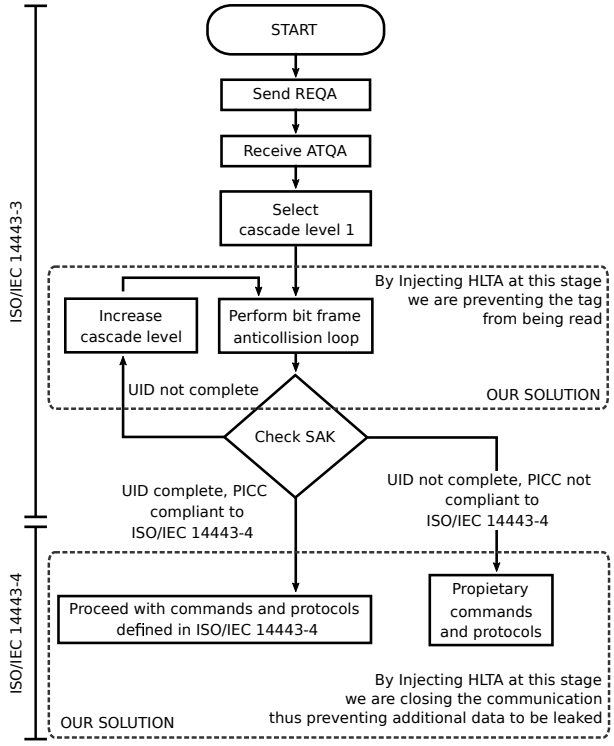


Fig. 1. PCD Initialization and Anti-collision

- *REQA and ATQA*: these messages are exchanged during the initialization phase, i.e. when the NFC tag enters the electromagnetic field emitted by the reader. During this phase the reader broadcasts REQAs messages and collects ATQAs replies from the tags;
- *SELECT and SAK*: the SELECT message is sent by the reader to ask for UIDs from those tags in close proximity. If only one reply is received, the transport layer protocol (ISO 14443-4) is started. Otherwise, the anti-collision protocol is initiated until a SAK command (also known as *select acknowledge*) is sent from the selected tag.

Figure 1 resumes the ISO 14443 (A/B) protocol as for the layers A-3 (top box) and A-4 (bottom box) respectively. The protocol is initiated by the reader via broadcasting REQA messages (*SEND REQA* in Fig. 1). Tags in close proximity can then reply with ATQA messages and finally the reader starts the anti-collision protocol that eventually will end up with the selection of an RFID UID. Such a selected tag can then proceed with the commands and protocols defined in the ISO 14443 A-4 (A/B).

Table I shows an example of the above ISO 14443 A-3 (A/B) messages with data extracted from a real communication among a reader and an NFC tag. The reading procedure is initiated by the reader with a REQA message (line 1) and a reply from the tag with an ATQA message (line 2). Then the reader starts the anti-collision procedure in order to obtain the tag's UID. During the first round of the anti-collision protocol, the reader requests the first 4 bytes of the tag's UID (line 3) and the tag replies with such an information (*04 46 70 ba*

TABLE I. A SAMPLE RUN OF THE NFC PROTOCOL

	Sender	*Message Code*	*Info*
1	R	26	REQA
2	V	44 00	ATQA
3	R	93 20	Anti-collision (1/2)
4	V	88 *04 46 70 ba*	V's reply
5	R	93 70 88 *04 46 70 ba* bf 41	SELECT (1/2)
6	V	04 da 17	V's confirmation
7	R	95 20	Anti-collision (2/2)
8	V	fa *d9 49 81 eb*	V's reply
9	R	95 70 fa *d9 49 81 eb* e5 05	SELECT (2/2)
10	V	00 fe 51 cd	V's confirmation

in line 4). At this point the reader selects the tag (line 5) which, in turn, replies and confirms its availability (line 6). The reader then requests the last 4 bytes of the tag's UID (line 7) and acquires them (line 9). In the rest of the protocol (from line 10 on) the reader gets access to the tag's memory which is accomplished via ISO 14443 A-3, thus not showed in Table I as it does not contain any useful information for our approach. Last but not least, it is worth noting that, in the above toy example, the anti-collision procedure is carried out in only two steps (lines 3 and 7) since only one tag is in the reader's close proximity. In general, more rounds might be necessary given the presence of multiple tags.

Based on the content of Table I we can observe that a device eavesdropping NFC communications can detect if unexpected tag reads have been performed by readers (potentially malicious) on tags (seen as the victim) and it can also identify the latter by their UIDs. As such, we have designed our solution to eavesdrop tags' UIDs from within the user's smartphone and to check if they belong to sensitive tags (the communication should be terminated) or not (the communication can continue). In case of a sensitive tag, our solution reacts by sending a command to the tag that resets its status thus preventing it to proceed with the data exchange protocol (ISO 14443 A-4).

IV. DEFINITIONS AND THREAT MODEL

Contactless technology comes with vulnerabilities that are usually exploited to perform eavesdropping, skimming, and relay attacks which are defined as it follows [8]:

- *Eavesdropping*: a malicious user is able to collect all the communications exchanged between the reader and the card over the wireless link;
- *Skimming*: a malicious user is able to capture credit card data (more in general: tag data) to reuse them later on;
- *Relay*: a malicious user is able to forward the communication between a dummy credit card (i.e. a *proxy*) that is used to perform some transactions at a point of sale (i.e. a *mole*) which has physical access to the real credit card.

The *eavesdropping* attack usually takes place when the card's owner is actively making a transaction and the malicious user is listening over the air. As such, eavesdropping has not been taken into account in this work since our solution focuses on those attacks in which the read of the victim's card goes unnoticed while the malicious user has full access to it (an example could be digital pick-pocketing).

To design our solution as a countermeasure for *skimming* and *relay* attacks, we followed the threat model defined by Roland and Langer in 2013 [8]. Indeed, they have implemented a *downgrade* and *pre-play* attack targeting EMV contactless credit and debit cards and leveraging several weaknesses in the kernel 2's Mag-Stripe authentication protocol. Thanks to these vulnerabilities, Roland and Langer were able to implement a real attack in which the attacker pre-computes a number of dynamic card verification codes (CVC3) from a genuine card and stores them on a cloned one. Furthermore, as their attack was protocol-based, Roland and Langer also implemented a *downgrade attack* in which other cards were forced to use the kernel 2's Mag-Stripe authentication protocol, thus being subject to the same vulnerabilities.

In our work, we assume the attacker to be able to accomplish not only the *downgrade* and *pre-play* attacks designed by Roland and Langer but also other attacks based on other protocol vulnerabilities targeting ISO 14443 contactless cards. Indeed, our solution enforces security during the *discovery* process (ISO 14443 A-3) when the attacker tries to connect to the victim's NFC cards (see Section V). During this process no data exchange protocols are executed (neither priprietary nor open-sourced ones) thus making us able to protect all those tags that abide to ISO 14443.

Definitions. We refer to the entities taking part in our envisaged scenario as follows (see Fig. 2):

- *Victim* (V): a tag to be protected, e.g. a credit card;
- *Tag* (T): any other generic tag;
- *Reader* (R): an NFC equipped reader trying to steal data from V;
- *Smartphone* (S): An NFC enabled smartphone running *N-Guard* and protecting V from unauthorized reads by R.

V. OUR SOLUTION: *N-Guard*

We introduce *N-Guard*: an NFC firmware implementation that enables every smartphone to prevent unauthorized readings from NFC skimmers. *N-Guard* changes the behavior of the smartphone from a standard NFC reader to a "smart firewall" that blocks readings on a pre-defined set of NFC enabled cards. In our solution, the user divides his own cards into two sets: sensitive (i.e *black-listed*) and not sensitive (i.e. *white-listed*). Black-listed cards cannot be read when they are safely stored close to the smartphone, while white-listed cards can be read at any time. Finally, the cards can be moved from one list to the other one as the user wish.

According to the previous scenario, the user safely keeps all of his NFC enabled cards close to the smartphone; for instance, by using a flip cover with card slots. *N-Guard* enables the smartphone to monitor the readings performed over the cards and to block those querying the cards that

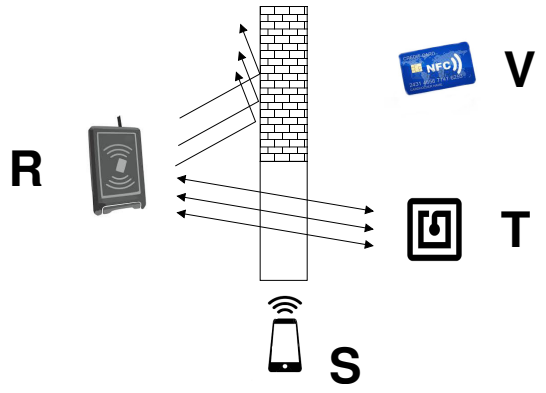


Fig. 2. Reference scenario with a reader (R), a sensitive NFC card (V) and a smartphone (S). The smartphone runs *N-Guard* thus protecting V from being read while allowing the access to non sensitive tags (i.e. T).

are black-listed. Conversely, white-listed cards are allowed to reply to all the requests.

We focus on the following two (mainstream) use cases such as *payments* and *tap-to-go* cards. On the one hand, in order to perform a payment, using his NFC enabled cards, the user has to take them outside of the flip cover, since the cards need to be placed at a given minimum distance from the cover to evade the smartphone fire-walling capability (see Section VI for more details). Hence, the user will be in full control of the payment procedure, without requiring any change to his usual behaviour. On the other hand, tap-to-go cards are usually adopted to grant the user the access to mobility services such as metro or bus. These cards do not contain any sensitive information and they are typically used several times a day. The user will tap the flip cover (containing the smartphone and all the cards) on the card reader, and eventually, the reader will be able to retrieve the information to authenticate the user but it will be prevented to access the black-listed cards.

A. Information flow

N-Guard is structured over two main phases: *selection* and *protection*. During selection, the user chooses the tags to be protected by *N-Guard*. The protection phase is the one actually preventing the chosen tags from being read, while leaving the other open to any kind of interaction. In the remaining of this section, we will first introduce the overall information flow, i.e. a high level description of all the functions that belong to both the *selection* and the *protection* phases (see Fig. 3). Then, we will describe in details how they have been implemented.

Selection Phase:

1) *TAG_SCANNING*: initiated by the application layer of S, it scans all the tags within close proximity;
2) *POLL_FOR_UIDs*: executed by the NFC chip, it polls for available tags UIDs within close proximity and reports them back to the application layer;
3) *ADD_TO_LIST*: executed by the application layer, it checks if the sensed UID has already being added to the list of those that has to be protected (i.e. it has been blacklisted).

Protection Phase:

1) *START_PROTECTING*: executed at the application layer of S, it loads all the blacklisted UIDs and triggers the *snoop* function;
2) *SNOOP*: executed at the firmware layer, it listens to the NFC channel for any communication;
3) *READ/SNIFF_TAG*: executed by R, the *read* function is used to maliciously collect data from the victim tag V. Since V is in close proximity of S, the NFC chip of S is also able to sniff the communication via the *sniff* function;
4) *CHECK_UID*: executed by the NFC chip of S, it analyzes the UID sent back to R by V in reply to *READ_TAG* function and if it is not blacklisted, it starts again snooping for other UIDs. Otherwise, the protection phase continues;
5) *INJECT_HALT*: executed by the NFC chip of S, it injects a *HLTA* command to V thus forcing the tag to move into the HALT state (i.e. preventing the communication with R to be completed);
6) *GOING_TO_HALT*: executed by V, it executes the *HLTA* command.

Depending on the current state of the victim tag V while being queried by the malicious reader R, the injection of the *HLTA* command can cause two different behaviors, i.e., change of states, as described in the following:

- READY to IDLE: V has not been selected yet by R. This means that V just replied to R with its UID thus remaining in the READY state. However, S has sniffed the V's UID and now it can check if there is a match of the UID in the list of sensitive tags. If so, S injects the *HLTA* command thus forcing V to go back to *IDLE*. The above behaviour is also consistent if V is in the *READY** state (see Fig. 3);
- ACTIVE to HALT: V has already been selected by R and it is waiting for the *READ* command to initiate the data exchange protocol. By injecting the *HLTA* command, S ensures that the exchange data protocol will be never initiated and no data will be read by R. This behaviour is also consistent if V is either in *DATA EXCHANGE* or *ACTIVE** state (see Fig. 3).

B. Implementation

In this work, we provide a real implementation of our solution by considering the Android OS platform and Google Nexus 5X phones, which are characterized by a large open source community with a long tradition of development and deployment of NFC services and applications. The android OS platform adopts two libraries to access the NFC chip: *libnfc-nxp*[3] (for NXP's PN54x) and *libnfc-nci*[4] (for Broadcom's BCM2079x and possibly other NCI-compliant chips). However, while the above libraries give access to a wide range of NFC functionalities, to the best of our knowledge, they lack the so-called *monitor mode*, or *RFMON*, i.e. Radio Frequency MONitor mode. Indeed, RFMON only applies to wireless networks and it allows to capture all the packets being

[3]https://android.googlesource.com/platform/external/libnfc-nxp/
[4]https://android.googlesource.com/platform/external/libnfc-nci/

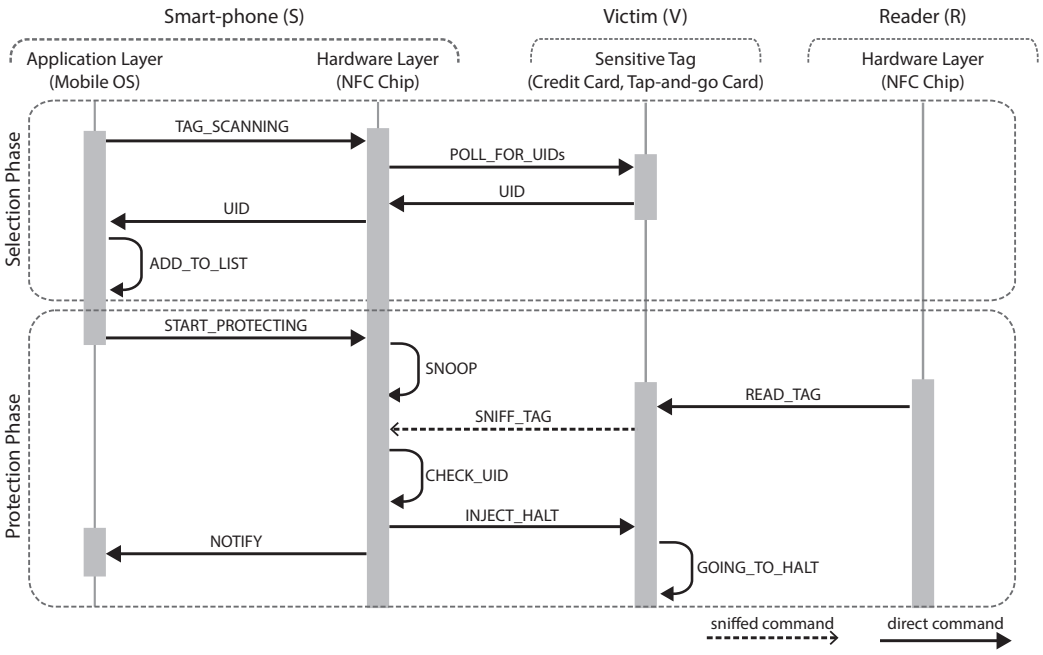


Fig. 3. Information flow implementing *N-Guard* detection and reaction logic.

transmitted in the radio spectrum without having to associate to either an access point or an ad-hoc network.

Since current NFC controllers embedded in the smartphones are not open, in order to prove the viability of our solution, we provide a real implementation of *N-Guard* by using a third party NFC chip to be attached to the smartphone. The adopted device is a Proxmark3[5], a Software Defined Radio (SDR) compliant with both NFC and RFID standards. The Proxmark3 is capable of transmitting and receiving at different protocol-specific timing requirements and it also provides full control over the radio layer in addition to software support for several higher-level protocols.

In the following, we provide the implementation details of *N-Guard* as for the previously introduced phases: *Selection* and *Protection*. For the former we have implemented an Android-based application that can be used as a managing interface for the *N-Guard*'s selection phase by using three main functions: i) *card loading*, ii) *card selection* and iii) *card protection*. In the former, the application waits for the user to tap a new card. Then, once a new card has been read by S, all its information are shown to the user by the *selection* function and last but not least, in the *protection* function the user selects the cards that have to be protected.

To implement the above functions within the Android OS client, we have modified the Proxmark3 client by creating a new function named *CmdHF14AProtection* within the *cmdhf14a.c* source file. This function takes as input a list of sensitive UIDs and it passes them to the NFC chip controller. As regards the protection phase, this phase has been realized by making S able to continuously eavesdrop the NFC channel and to prevent any communications from UIDs that do generate a hit in the blacklist. To do so, we have identified the function *UsbPacketReceived* defined in the file *appmain.c* that is triggered by the Android client executed by the NFC chip controller. We have then modified this function to receive the UIDs list and to pass it to the *CHECK_UID* and *INJECT_HALT* functionalities inside the Proxmark3 (see Fig. 3). These two functionalities have been implemented inside the *ReadDetection14443a* and *ReadReactionIso14443a* functions as follows:

- *ReadDetection14443a*: this function snoops V's UIDs sent to the malicious reader R and, if they are blacklisted, interrupts the snoop process thus going back to the *UsbPacketReceived* function. Otherwise, it continues with the next function;
- *ReadReactionIso14443a*: this function sends an HLTA command to V thus forcing it to move the HALT state which causes the interruption of the communication with the malicious reader R.

VI. Experimental Tests

During our experimental analysis we have used two Proxmark3 devices as well as a smartphone and a MiFARE tag. One proxmark acted as the malicious reader (R) and tried to continuously read the victim tag (V). The other one acted as the smartphone S and leveraged *N-Guard* to prevent unauthorized readings of V, implemented with the MiFARE Ultralight tag.

As already introduced in Section V-B, our proof-of-concept involves the above proxmark external devices to behave as NFC radio eavesdroppers and transmitters due to the impossibility to directly re-programming nowadays smartphone NFC controllers. However, our solution is completely portable and deployable on any NFC enabled device that

[5]https://evola.fr/en/rfid/847-proxmark3-v2-kit.html

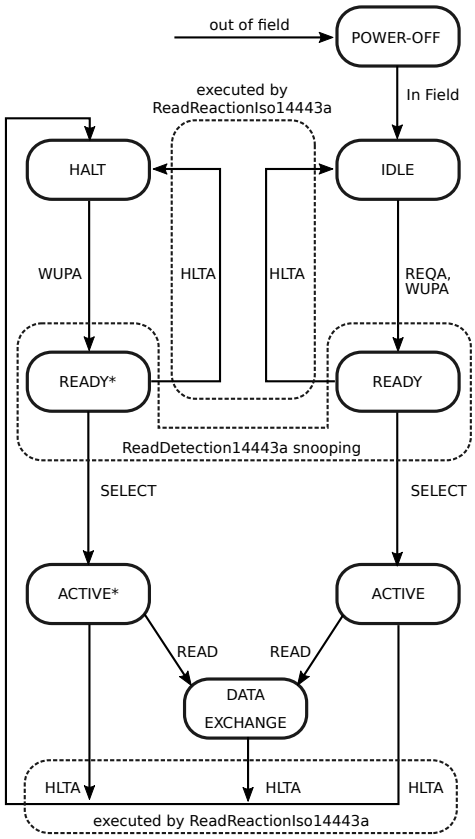


Fig. 4. ISO 14443-A3 execution diagram. Dotted boxes represent the processes enforced by our solution.

abides to ISO 14443. Indeed, as detailed in Section V-B, we have adopted and modified commands, functions, and files that are fully compliant to the ISO 14443-3 standard, and all of them can be promptly embedded on NFC controllers by manufacturers or developers in case of open hardware.

A. Blocking unauthorized access: experimental results

In this section we analyze *N-Guard* performances as concerns its capacity of blocking unauthorized access to the NFC tags. We will consider two different usage scenarios: when the Standard Protocol ISO 14443A is employed; and, when a proprietary protocol is used. For this latter scenario we will utilize the protocol implemented by a popular credit card brand, i.e., MasterCard.

- **Standard Protocol**: we have used a re-writable MiFARE ISO 14443A tag and tried to read its content from an NFC-info android application. In this approach, both the reader (the android application) and the tag followed the ISO 14443-4 protocol as reported in Table II and Table III;
- **Proprietary Protocol**: we have used a real MasterCard and the EMVemulator[6] Android OS application. Both the reader and the card adopt a proprietary protocol but *N-Guard* is still able to detect the attack and to mitigate it.

Standard Protocol. We started by cloning the information from a VISA credit card to an ISO compliant open tag. Then, we tried to retrieve such information from the tag while protecting the communication with *N-Guard*. In Table II is possible to see the tag's state transition from READY to IDLE. Such a transition is triggered by *N-Guard*, hence preventing the adversary to read the tag's sensitive information. Indeed, after the REQA and ATQA messages, we can see at line 4 that the tag V is replying with the first part of its UID (04 46 70). At this point, *N-Guard* is able to detect the reply and to recognize that the UID is one of those that have been blacklisted. Therefore, S reacts by injecting the HLTA command (line 6) which eventually prevents the malicious reader R to read the V's content (lines 7 and 8 have no replies). However, as shown in lines 9 and 10, R can resend REQA to initialize again the communication with V. Indeed, *N-Guard*'s protection is not a one time process but it is executed every time a SELECT is made (more details on the experimental tests are reported on Section VI-B).

Table III shows the list of commands involved in the ACTIVE to HALT behavior. In this case, we are assuming that R is able to select and to initialize a communication with the victim tag V. Indeed, we can see that at line 6 and 10, V confirms its UID. We are also assuming that *N-Guard*'s reaction is slower than V reaction, indeed no HLTA command is injected at this time. Therefore, the malicious reader R is able to send a *BLOCK READ* command and to receive a reply from the victim tag V (line 12). Eventually, the HLTA command sent by *N-Guard* is received by V (line 13) and the next BLOCK READ request does not receive any reply (line 14). As in the previous case, *N-Guard* should be repeated over the time as soon as a new interaction between the victim tag V and the malicious reader R is detected.

Proprietary protocol. We tested *N-Guard* against a proprietary protocol, e.g., the malicious reader R communicates with the victim tag V with a protocol unknown to S. We installed the *EMVemulator* application in our malicious reader R and used it to steal information from our victim tag V (using MasterCard PayPass tag[7]). EMVemulator is based on the combined pre-play and downgrade attack described by Michael Roland and Josef Langer as previously introduced in Section IV. Again, as for the standard protocol scenario, *N-Guard* behaves differently as a function of the interleaving of the messages exchanged by the various entities.

- *Successful EMVemulator*: R is able to read V's card number and expiration date as well as to compute the payment pre-authorization keys;
- *Dropped EMVemulator*: R is able to read V's card number and expiration date. However, while computing the payment pre-authorization keys, *N-Guard* prevents the communication between R and V since V is forced to the HALT state. The result is a failed attack;

[6]https://github.com/MatusKysel/EMVemulator

[7]http://www.mastercard.com/sea/consumer/paypass.html

TABLE II. *N-Guard* READY TO IDLE REACTION

	Sender	*Message Code*	*Info*
1	R	26	REQA
2	V	44 00	ATQA
3	R	93 20	Anti-collision (1/2)
4	V	88 *04 46 70 ba*	V's reply
5	R	93 70 88 *04 46 70 ba* bf 41	SELECT (1/2)
6	S	*50 00* 57 cd	HLTA command
7	R	*30 00* 02 a8	READ BLOCK 00
8	R	*30 04* 02 a8	READ BLOCK 04
9	R	26	REQA
10	V	44 00	ATQA

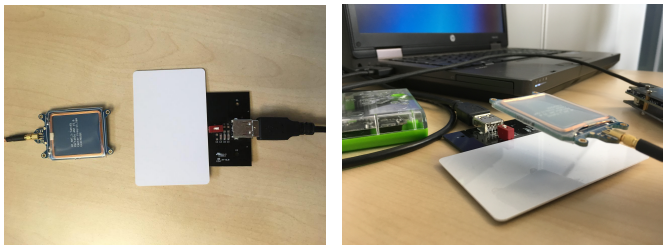
TABLE III. *N-Guard* ACTIVE TO HALT REACTION

	Sender	*Message Code*	*Info*
1	R	26	REQA
2	V	44 00	ATQA
3	R	93 20	Anti-collision (1/2)
4	V	88 *04 46 70 ba*	V's reply
5	R	93 70 88 *04 46 70 ba* bf 41	SELECT (1/2)
6	V	04 da 17	V's confirmation
7	R	95 20	Anti-collision (2/2)
8	V	fa *d9 49 81 eb*	V's reply
9	R	95 70 fa *d9 49 81 eb* e5 05	SELECT (2/2)
10	V	00 fe 51 cd	V's confirmation
11	R	*30 00* 02 a8	READ BLOCK 00
12	V	04 46 70 ba fa d9 49 81 eb 48 00 00 e1 10 12 00 ad c5	BLOCK 00 content
13	S	*50 00* 57 cd	HLTA command
14	R	*30 04* 02 a8	READ BLOCK 04
15	R	26	REQA

- *Denied EMVemulator*: *N-Guard* is already running when R tries to execute the EMVemulator attack towards V. The result is a complete attack failure. Indeed, EMVemulator is still waiting for a tag to be in proximity while the communication from the victim tag V, i.e., MasterCard, stopped—V is forced to a HALT state by *N-Guard*.

Analyzing the log of the messages being exchanged during the attack, we observe that Proxmark3 is not able to interpret them as belonging to a proprietary protocol, i.e. the one implemented by MasterCard.

However, regardless of the protocol being used, the tag has to claim its identity to be recognized by the reader. Indeed, the victim tag V has to transmit its UID anyway. Since *N-Guard* identifies the NFC tags by eavesdropping the transmitted UIDs, our solution is still able to prevent the attack even in the case the tag is queried by a proprietary protocol. In fact, running the same attack scenario but considering a smartphone S running *N-Guard* we notice that every time the victim tag V sends its UID, *N-Guard* replies with the *HALT* command, practically stopping the victim to further proceed in the communication exchange with R—thus forcing the malicious reader R to start the protocol again, if it wants to be successful. The result turns out to be an infinite loop in which no pre-authentication tokens are collected by R proving the effectiveness of *N-Guard* even in those cases where the protocol used by the (legitimate) reader is proprietary.



(a) Shifting the tag V from S. (b) Lifting the tag V from S.

Fig. 5. Testing *N-Guard* from different positions.

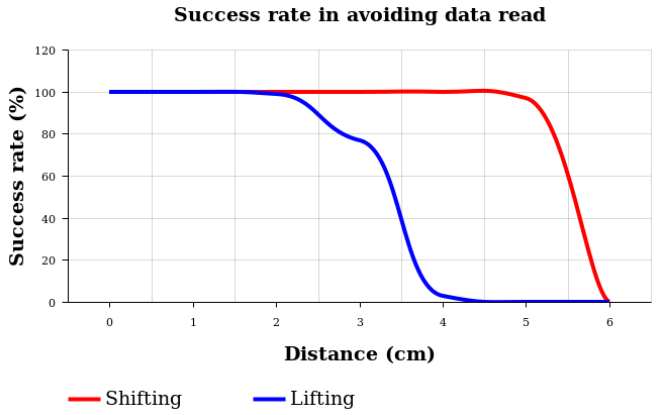


Fig. 6. Success rate when preventing data exfiltration. This test has been performed by using the experimental limit of 3500 blacklisted UIDs.

B. Performance Analysis

We used two methodologies: *shifting* and *lifting* the smartphone S from the victim tag V while keeping the malicious reader R in touch with V (see Fig. 5a and Fig. 5b).

For each test, we considered a total of 3500 different UIDs for V, which is the experimental limit before overflowing Proxmark3's buffer. With both methodologies, we performed a set of measures at different distances each one consisting of an attack on V (i.e. R trying to read data from V). We registered a *success* when S (running *N-Guard*) was able to prevent the attack or a *failure* when R was succeeding, respectively. The tests were performed as follows: starting from a complete overlap of the devices, we recorded our first 100 measures. Then we shifted/lifted S from V increasing their distance 1cm every time, collecting other 100 measures. We iterated the previous process until we reached the *critical distance* when all the attacks succeeded, i.e. *N-Guard* success rate equal to 0% in Fig. 6 at about 4cm for the lifting, and about 6cm for the shifting. We observe that, for short distances between S and V, *N-Guard* always achieves a success rate close to 100% with both methodologies.

C. Power Consumption

This section discusses the results of the the power consumption measurements we performed over an (emulated) smartphone running *N-Guard*. Indeed, we performed the power consumption measurements on a Proxmark3 running

N-Guard emulating a real smartphone behaviour. We consider three different configurations such as i) the smartphone S is kept in the idle state, ii) *N-Guard* is active within the smartphone S but does not react to malicious activities and iii) *N-Guard* is active and also reacts. The measurements have been performed by using a USB dongle connected to the Proxmark3 and logging the consumption values. We did not observe any noticeable increase in the power consumption. Moreover, we highlight that the NFC eavesdropping functionality is already working independently from *N-Guard* thus not affecting the battery drain of the smartphone. Furthermore, once an attack is detected, *N-Guard* requires the transmission of one only message to switch the victim tag to the *HALT* status. As such, we observed *N-Guard* to be also extremely efficient as regards the energy consumption.

VII. Conclusions

In this paper we have introduced *N-Guard*: a solution to prevent fraudulent extraction of sensitive data from NFC enabled devices that relies on an NFC capable smart-phone. We have detailed the rationals supporting *N-Guard*, and we have experimentally shown its effectiveness. In particular, *N-Guard* is able to protect standard-abiding NFC communications, as well as proprietary protocols (we have shown it using the Mastercard's NFC proprietary protocol). Moreover, our solution is general (it applies to all NFC tags), completely transparent, and fine grained. For instance, adopting *N-Guard* does not require any change to the customer's habits, to the reader or even to the tag. Finally, we verified that *N-Guard* has a negligible power consumption overhead.

References

[1] H. Stockman, "Communication by means of reflected power," in *Proceedings of the I.R.E.* USA: IEEE, October 1948, pp. 1196–1204.

[2] MasterCard, "PayPass - M/Chip requirements," pp. 1–84, 2014. [Online]. Available: https://www.paypass.com

[3] R. J. Anderson, "Position statement in RFID s&p panel: RFID and the middleman," in *Financial Cryptography and Data Security, 11th International Conference, FC 2007*. Heidelberger Platz 3, 14197 Berlin: Springer, Berlin, Heidelberg, 2007, pp. 46–49.

[4] F. Lishoy, H. Gerhard, M. Keith, and M. Konstantinos, "Practical nfc peer-to-peer relay attack using mobile phones," in *Proceedings of the 6th International Conference on Radio Frequency Identification: Security and Privacy Issues*, ser. RFIDSec'10. Heidelberger Platz 3, 14197 Berlin - Germany: Springer, Berlin, Heidelberg, 2010, pp. 35–49.

[5] T. S. Heydt-Benjamin, D. V. Bailey, K. Fu, A. Juels, and T. O'Hare, "Vulnerabilities in first-generation rfid-enabled credit cards," in *Financial Cryptography and Data Security, 11th International Conference, FC 2007*. Heidelberger Platz 3, 14197 Berlin - Germany: Springer, Berlin, Heidelberg, 2007, pp. 2–14.

[6] S. J. Murdoch, S. Drimer, R. Anderson, and M. Bond, "Chip and pin is broken," in *2010 IEEE Symposium on Security and Privacy*. USA: IEEE, May 2010, pp. 433–446.

[7] M. Bond, O. Choudary, S. J. Murdoch, S. P. Skorobogatov, and R. J. Anderson, "Chip and skim: cloning emv cards with the pre-play attack," *CoRR*, vol. abs/1209.2531, p. 21, 2012.

[8] M. Roland and J. Langer, "Cloning credit cards: A combined pre-play and downgrade attack on emv contactless," in *Presented as part of the 7th USENIX Workshop on Offensive Technologies*. Washington, D.C.: USENIX, 2013, pp. 1–22.

[9] G. P. Hancke, "Practical eavesdropping and skimming attacks on high-frequency rfid tokens," *J. Comput. Secur.*, vol. 19, no. 2, pp. 259–288, Apr. 2011.

[10] M. R. Rieback, B. Crispo, and A. S. Tanenbaum, *RFID Guardian: A Battery-Powered Mobile Device for RFID Privacy Management*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 184–194.

[11] G. P. Hancke, K. E. Mayes, and K. Markantonakis, "Confidence in smart token proximity: Relay attacks revisited," *Comput. Secur.*, vol. 28, no. 7, pp. 615–627, Oct. 2009.

[12] Z. Kfir and A. Wool, "Picking virtual pockets using relay attacks on contactless smartcard," in *Security and Privacy for Emerging Areas in Communications Networks, 2005. SecureComm 2005. First International Conference on*. USA: IEEE, Sept 2005, pp. 47–58.

[13] I. Kirschenbaum and A. Wool, "How to build a low-cost, extended-range rfid skimmer," in *Proceedings of the 15th Conference on USENIX Security Symposium - Volume 15*, ser. USENIX-SS'06. Berkeley, CA, USA: USENIX Association, 2006, pp. 1–22.

[14] Y. Lee, "Antenna circuit design for RFID application." in *Microchip Technology, Application Note AN710, DS00710C*. Chandler, Arizona, USA: Microchip Technology Inc, 2003, pp. 1–50.

[15] R. Jin and K. Zeng, "Secnfc: Securing inductively-coupled near field communication at physical layer," in *Communications and Network Security (CNS), 2015 IEEE Conference on*. USA: IEEE, Sept 2015, pp. 149–157.

[16] R. Jin, X. Du, Z. Deng, K. Zeng, and J. Xu, "Practical secret key agreement for full-duplex near field communications," *IEEE Transactions on Mobile Computing*, vol. 15, no. 4, pp. 938–951, April 2016.

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

[18] A. Majumder, J. Goswami, S. Ghosh, R. Shrivastawa, S. P. Mohanty, and B. K. Bhattacharyya, "Pay-cloak: A biometric back cover for smartphones: Facilitating secure contactless payments and identity virtualization at low cost to end users." *IEEE Consumer Electronics Magazine*, vol. 6, no. 2, pp. 78–88, April 2017.

[19] J. J. Gummeson, B. Priyantha, D. Ganesan, D. Thrasher, and P. Zhang, "Engarde: Protecting the mobile phone from malicious nfc interactions," in *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '13. New York, NY, USA: ACM, 2013, pp. 445–458.

[20] R. Zhou and G. Xing, "nshield: A noninvasive nfc security system for mobiledevices," in *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '14. New York, NY, USA: ACM, 2014, pp. 95–108.

[21] D. A. Ortiz-Yepes, "A review of technical approaches to realizing near-field communication mobile payments," *IEEE Security Privacy*, vol. 14, no. 4, pp. 54–62, July 2016.

[22] G. Avoine, X. Carpent, and J. Hernandez-Castro, "Pitfalls in ultra-lightweight authentication protocol designs," *IEEE Transactions on Mobile Computing*, vol. 15, no. 9, pp. 2317–2332, Sept 2016.