Fortune cookies and smartphones: Weakly unrelayable channels to counter relay attacks[☆]

Mario Čagalj[a], Toni Perković[a,*], Marin Bugarić[a], Shujun Li[b]

[a] *FESB, University of Split, Croatia*
[b] *University of Surrey, UK*

Abstract

Smartphones are being increasingly used to perform financial transactions (through m-banking, virtual wallet or as a smartcard). The latter applications involve contactless technology (e.g., NFC) that is known to be vulnerable to *mafia fraud attacks*. In this work we show that a secret message inside an appropriately folded piece of paper (*fortune cookie*) can be used to effectively mitigate the mafia fraud attack. Fortune cookies implement a *weakly unrelayable* channel that, in combination with smartphones, provides a *provable* protection against those attacks. Our solution requires minimal or no hardware changes to the existing equipment (especially on the user's side) and is suitable for different communication technologies (e.g., intra-body communication, NFC, WiFi, Bluetooth, sound, infrared).

Keywords: Wireless security, mafia-fraud attack, weakly unrelayable channel, usable security, multichannel protocols

1. Introduction

Smartphones are constantly changing and influencing our lives thanks to their diverse capabilities (gaming, sensing, positioning or information availability). Recently, smartphones have become fully equipped payment devices that could easily replace credit cards, which is already accepted by many banks [1]. Google and some mobile operators (AT&T, Verizon and T-Mobile) have seen the potential of mobile payments and began pushing NFC chips on smartphone devices (such as Nexus S [2]). Alternative communication technologies have also emerged in the context of short-range mobile transactions. VeriFone's Zoosh uses ultrasound to exchange data between devices equipped with speakers and microphones [3]. On NRC's ATMs a user can withdraw money by simply scanning a QR code presented on the ATM's screen with a smartphone camera [4].

All the aforementioned communication technologies as well as the classical contact-based PIN-protected smart cards are vulnerable to a tricky form of a relay attack - *the mafia fraud attack* [5, 6, 7, 8]. A classic example of the relay attack was first presented in the work by Conway [9]. Some other similar attacks have been proposed, named wormhole attack [10], terrorist attack [7] and distance hijacking attack [11].

Let us briefly review the mafia fraud attack using the scenario shown in Fig. 1. An honest user equipped with a smartphone approaches a fake ATM that is under the control of the attacker. The user initiates the NFC-based transaction with the fake ATM, and at the same time, the attacker simply relays the communication between the user's smartphone (honest *prover*) and the honest terminal (*verifier*). After successfully authenticating the victim, the honest terminal outputs the requested money that the attacker simply picks up. Please note that the attacker does not have

[☆] This is the author's version of a work that was published in *Pervasive and Mobile Computing* in 2015. Changes resulting from the publishing process, such as peer review, editing, corrections, structural formatting, and other quality control mechanisms may not be reflected in this document. Changes may have been made to this work since it was submitted for publication. A definitive version has been published online by Elsevier in *Pervasive and Mobile Computing*, vol. 20, pp. 64-81, 2015, DOI: 10.1016/j.pmcj.2014.09.002.

[*] Corresponding author

Email addresses: mcagalj@fesb.hr (Mario Čagalj), toperkov@fesb.hr (Toni Perković), mbugaric@fesb.hr (Marin Bugarić), shujun.li@surrey.ac.uk (Shujun Li)

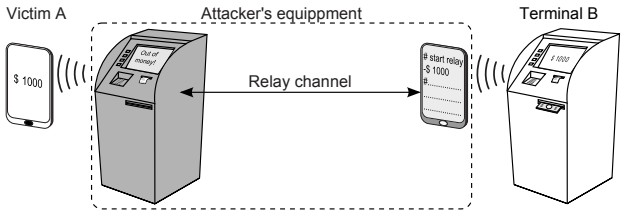


Fig. 1. The relay attack against the customer *A* who wants to withdraw some money using her NFC-enabled smartphone.

to know the transaction (authentication) details (i.e., the secret key shared between the honest parties) - the attacker simply relays all the messages in their original form [12]. This is what makes the mafia fraud attack so subtle.

Practical implementations of the mafia fraud attack have been proposed against contact and contactless based technologies [6, 12, 13, 14, 15]. Drimer and Murdoch [6] developed a relay attack against modern Chip-and-PIN system, while Francis et al. [12] demonstrated an NFC peer-to-peer relay attack on smartphones, showing the simplicity of such attacks using off-the-shelf equipment.

Most existing solutions against relay attacks are based on distance-bounding protocols that enable a verifier to establish an upper bound on the physical distance to a prover based on round-trip-time of cryptographic challenge-response pairs exchanged between two communicating parties [16, 17, 18, 19]. These solutions have been practically implemented in both contact and contactless (radio) systems [6, 18, 20, 21, 22]. Drimer and Murdoch [6] implemented a countermeasure against relay attacks on Chip-and-PIN systems using protocol by Hancke and Kuhn [17]. Rasmusen and Čapkun implemented radio distance-bounding protocol with fast processing equipment (Ultra-Wide-Band) in which a malicious prover can pretend to be at most 15 cm closer to the verifier [22]. In all these implementations, the protocol has to be run over special channels allowing for an accurate time/delay measurement - requiring nanosecond precision. Most existing secure distance-bounding protocols would require modified provers (smart tokens) and/or verifiers (terminals) to be effective against relay attacks. By the time such solutions are implemented, there will already be many distance-bounding non-compliant NFC systems and mobile devices around and retrofitting them may prove too costly if not impossible.

The ideal solution for mafia fraud attacks should cover a range of proximity technologies (e.g. NFC, Bluetooth, WiFi, IBC - Intrabody Communication). Moreover, the solution should not require any significant hardware changes to the existing equipment. In this paper, we design such a solution using the general paradigm of unrelayable channels that were introduced by Stajano et al. [23]. In [23] the authors suggest augmenting the channels normally used for the transaction with an additional, special channel that attackers will not be able to relay. However, the solutions proposed in [23] are highly impractical and serve only as illustrations of the *unrelayable channels* paradigm. Unrelayable channel is characterized by the following properties: *unclonability*, *untransportability* and *unsimulability*. In the present work, we introduce the notion of *weakly unrelayable* channel (see Definition 1) by relaxing the *unclonability* and *untransportability* properties. In other words, a weakly unrelayable channel is the one that can be relayed but only at a non-negligible time cost that is easily detectable by the common systems (smartphones and terminals).

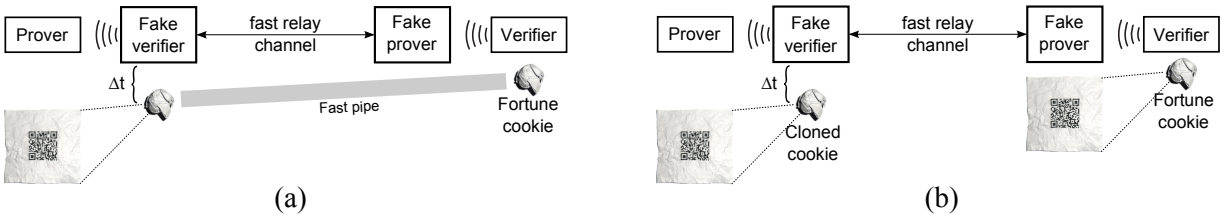


Fig. 2. Two possible relaying/attacking strategies: the attacker can either (a) physically transfer the original fortune cookie over to the honest prover or (b) somehow read (or guess) the secret nonce from the cookie and clone it on the side of the honest prover.

Definition 1. *A channel is said to be* ***weakly unrelayable*** *if it satisfies the following properties:*

- ***weak unclonability****: it must be prohibitively difficult to produce a copy of some physical aspect of the given channel within a predefined time period* Δt*;*
- ***weak untransportability****: it must be prohibitively difficult to manufacture a data pipe capable of transporting a detectable physical aspect of that channel from one location to another within a predefined time period* Δt*;*
- ***unsimulability****: it must be prohibitively difficult to simulate some physical aspect of that channel.*

In this paper we show how to implement such a channel using a piece of a regular paper to provide a strong and provable protection against mafia fraud attacks. An honest *verifier* (terminal B in Fig. 1) issues a secret challenge printed inside of an appropriately folded/crumpled paper (Fig. 2) to an honest *prover* (smartphone A in Fig. 1). The prover in turn has to sign the issued challenge and return the result back to the verifier. Without knowing the secret challenge, the prover cannot authenticate successfully with the verifier. In order to relay the challenge hidden within the folded paper, the adversary has only two possibilities: either (i) to physically transfer the original paper carrying the secret challenge between the two honest endpoints (Fig. 2(a)) or (ii) to somehow read (or guess) the challenge from the original paper and clone it on the side of the honest prover (Fig. 2(b)). Both strategies incur a non-negligible relay channel delay, because we use a special *paper-based* channel to transfer challenge values. By bounding this delay, we can effectively mitigate the relay attack.

Our contributions. We present a solution against relay attacks, which is suitable for scenarios that involve transaction (paper) receipts such as ATM money withdrawals and payment terminals. Our solution is based on a novel (multichannel) authentication protocol called Force (**For**tune **C**ooki**e**), designed to work with a regular radio channel and a special weakly unrelayable (paper-based) channel. We also provide a security proof of our protocol in a formal model (random oracle). Force is a reminiscent of regular distance-bounding techniques, but with an important difference that it does not require any significant hardware changes to the existing equipment (particularly on the user's side). Moreover, Force protocol works with diverse communication technologies (being contactless such as near-field communication - NFC, ultrasound, intra-body communication, WiFi, Bluetooth, or contact-based). We show that a paper-based channel is indeed difficult to relay within a time period of around 100 ms. At the same time, such relaying delays can be reliably measured by standard equipment.

We implemented two instantiations of our solution and performed an extensive user performance study with 54 participants. The study indicated that the proposed method has low execution times and minimal error rates.

2. Authentication protocol for weakly unrelayable channels

In this section we provide details of our Force protocol (**For**tune **C**ooki**e**), as shown in Fig. 3. A user, equipped with a smartphone (entity A), authenticates with the terminal (entity B), and wants to use a service offered by this terminal (e.g., withdraw some money). We assume that the smartphone A and the terminal B share a secret key K. As shown in Fig. 3, the Force protocol evolves in three phases.

Phase I: Initialization. The user enters the transaction details into her smartphone (i.e., PIN, desired amount of money) as well as the identity of the terminal B (this initial association can be done as with WiFi access points - by selection from the list). Having entered all the transaction details, the user uses her smartphone to initiate the Force protocol. At this stage, the smartphone A transmits its identity over a regular channel to the terminal B.

Phase II: Event synchronization/correlation. Having successfully verified user identity, the terminal B prepares the fortune cookie - i.e., generates a random challenge R, prints it on a paper, appropriately folds the paper (see Fig. 3(right)) and exposes it to the user via an appropriate dispenser. At this stage, the user pulls the fortune cookie out (marked with the dashed down-sloped arrow in Fig. 3) and at the same time, using her smartphone, records this event (see Fig. 3(right)). This is the crucial stage for our protocol, where the smartphone A and the terminal B separately detect the event of the fortune cookie leaving one *trusted zone* (the terminal) and entering another (the user's hand). These separate observations of the same event are thus correlated in time - the fact that we securely verify in this phase of the protocol. To accomplish this, the terminal B records the moments T_{B0} when the user begins to draw the fortune cookie and T_{B1} when the cookie leaves the terminal. At the same time the user, using her smartphone, also timestamps the very moment (T_A in Fig. 3) at which the cookie leaves the terminal. By comparing the recorded times T_A and T_{B1},

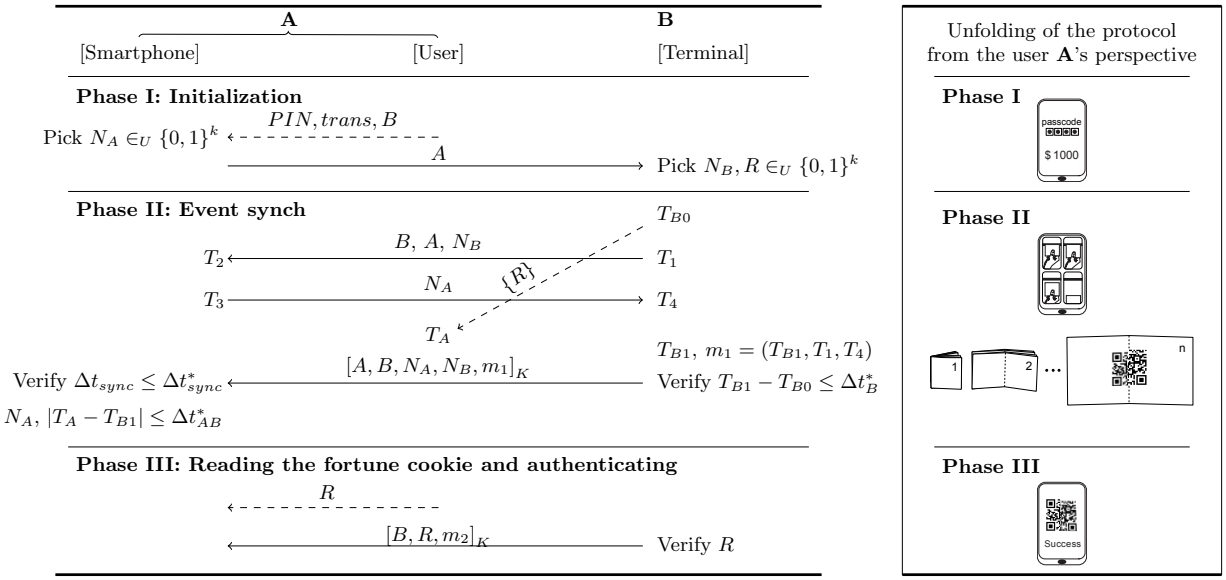


Fig. 3. Authentication protocol for weakly unrelayable channel - Force. Dashed arrows denote message transfer between the user and a device, while communication over a regular channel (e.g. WiFi, Bluetooth, NFC) is denoted with solid arrows.

A and B make sure that their observations are correlated in time. For this to work, A and B have to synchronize their clocks. For our purposes, a weak synchronization is sufficient (e.g., the corresponding clocks should be within 10 ms) and we use the synchronization protocol similar to [24].

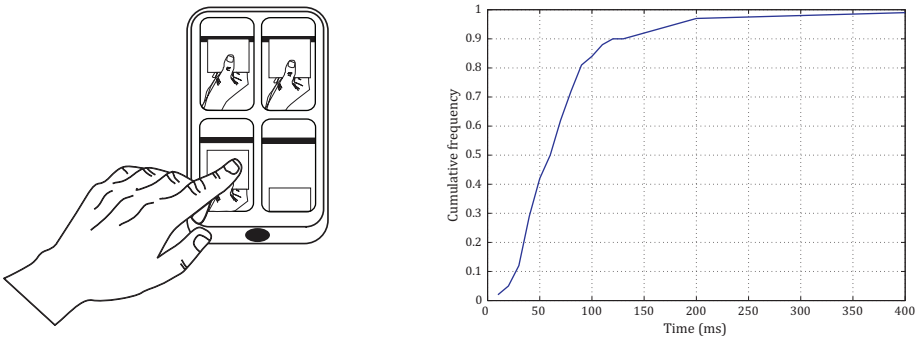
The coarse clock synchronization is initiated by the terminal B at the time instant T_1. As shown in Fig. 3, the terminal B transmits the random nonce N_B along the identities A and B; B records the moment T_1. The smartphone A receives this message at time T_2 and quickly responds with its own random nonce N_A at T_3; A records T_2 and T_3. The terminal B receives N_A at time T_4 and replies with the signed message $[A, B, N_A, N_B, m_1]_K$, where $m_1 = (T_{B1}, T_1, T_4)$ and the message is signed with the shared key K.

Upon receipt of the message $[A, B, N_A, N_B, m_1]_K$, the smartphone A verifies the message authenticity and that the end-to-end delay $\Delta t_{sync} \triangleq (T_2 - T_1 + T_4 - T_3)/2$ satisfies $\Delta t_{sync} \leq \Delta t^*_{sync}$. If all the verifications are successful, the smartphone calculates the clock offset $offset_A \triangleq T_1 + \Delta t_{sync} - T_2$, and adjusts accordingly its clock. After synchronizing its clock with the terminal B, the smartphone A verifies that the times T_A and T_{B1} are within the expected bound, i.e., $|T_A - T_{B1}| \leq \Delta t^*_{AB}$. For reasonably low values of Δt^*_{AB} (e.g. 100 ms). The last verification establishes that A and B are *event synchronized* - their observations of the fortune cookie-extraction event are correlated in time. For security reasons the terminal limits the cookie extraction time to Δt^*_B, i.e., $T_{B1} - T_{B0} \leq \Delta t^*_B$. After all the verifications, the protocol proceeds to Phase III.

Phase III: Reading the fortune cookie and authenticating. The user opens (unfolds) the fortune cookie and reads its content (the nonce R) using her smartphone (the dashed arrow in Fig. 3). The nonce R can be encoded with QR codes to make this process more user-friendly. Finally, the smartphone A sends to the terminal B the signed message $[B, R, m_2]_K$, where m_2 contains transaction details (i.e., the amount of money, the account, the user's ID, etc. - this message can also be encrypted to ensure privacy). Upon a successful verification of the last message, the terminal B provides the service (e.g. gives out the money).

Event synchronization with a smartphone

We describe a method used by the user A to precisely timestamp the moment (T_A) when the fortune cookie leaves the terminal. The proposed method utilizes a camera equipped smartphone. At the end of Phase I of the Force protocol (Fig. 3), the user (i) points the smartphone's camera towards the terminal and (ii) begins to pull the fortune cookie out of the terminal. At the same time the user has to start recording the cookie extraction process. With longer



(a) (b)

Fig. 4. (a) A sequence of recorded frames that capture the fortune cookie extraction as seen by the user on her smartphone. The user selects the first frame on which she sees that the fortune cookie left the terminal. (b) The cumulative frequency distribution of the cookie extraction time (for the 2 cm long QR encoded secret nonce R).

range communication technologies such as WiFi and Bluetooth, this can be made fully automated by the terminal B. Thus, the smartphone can begin recording upon the reception of the first clock synchronization message, i.e., (B, A, N_B) in Fig. 3. The recording ends with the last clock synchronization message (that is $[A, B, N_A, N_B, m_1]_K$). In the case of short range communication technologies such as NFC, the user would have to control the recording process manually. At this stage the user holds the fortune cookie in her hand (the user's trusted zone), while the terminal and the smartphone have synchronized clocks. To determine the moment T_A the user is presented with a sequence of recorded frames on the smartphone as shown in Fig. 4(a). The user has to simply select the first frame on which she sees the fortune cookie completely extracted from the terminal[1]. As each frame is timestamped relative to the beginning of Phase II, the smartphone can learn the time T_A at which the cookie left the terminal.

Possible alternative event synchronization methods include: the pushbutton-based event synchronization where the user pushes the button once she observes that the cookie left the terminal and the accelerometer-based event synchronization where instead of pushing the button, she briefly shakes her accelerometer equipped smartphone. Both of those methods are prone to errors that can make the whole protocol insecure (as shown in Section 6).

3. Attacker and protocol timing model

In this section, we introduce the attacker model, as well as the protocol timing model, which will be used later in the formal security assessment of the Force protocol. We consider a strong attacker model in which the attacker has a full control over both channels of Force protocol, thus he can eavesdrop, drop, delay, relay, replay and modify messages sent over both radio and *weakly unrelayable* (paper-based) channels. Although very powerful, we assume a realistic attacker in the sense that he has to perform some work (i.e., invest some time) to mount any of the above attacks.

Depending on the desired security level, the fortune cookie is folded once or multiple times. As can be seen from Fig. 5, the part of the fortune cookie holding the secret nonce (the inner dashed square) resides inside the terminal B (the shaded part denoted as a *trusted zone*). At this stage, the attacker does not have access to the secret nonce R. The part of the cookie outside of the terminal is within the untrusted zone that is under the attacker's control.

Fig. 5 shows the unfolding of the fortune cookie extraction over time in the presence of the relaying attacker. Recall, T_{B0} marks the beginning of the Phase II of the Force protocol (see Fig. 3); it is the moment (as recorded by the terminal) when the user/attacker begins to draw the legitimate fortune cookie out. Likewise, T_{B1} is the time at which the fortune cookie completely leaves the terminal's trusted zone (Fig. 5). We denote the *cookie extraction time*

[1] Already a large number of easy-to-use applications allow the user to grab frames from the recorded video [25, 26]. In these applications, the user simply slides through the frames and selects the most appropriate one.

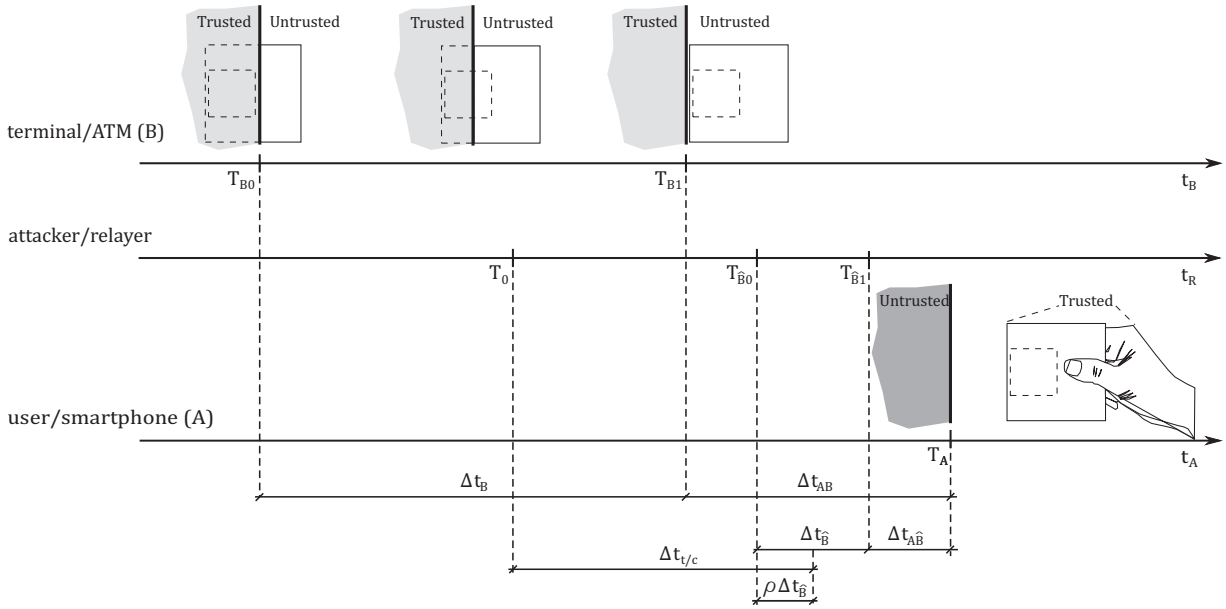


Fig. 5. Unfolding of the fortune cookie extraction over time in the presence of an active attacker.

with Δt_B, that is, $\Delta t_B \triangleq T_{B1} - T_{B0}$. For security reasons, we limit Δt_B to be smaller than some predefined value Δt_B^*, which we characterize later in the section. This security check is enforced by the terminal; the terminal will abort the protocol if $\Delta t_B > \Delta t_B^*$. Referring back to Fig. 5, T_A represents the very moment when the user's smartphone recorded the act of cookie leaving the terminal. From this moment on, the cookie (either legitimate or fake) certainly resides in the trusted zone (in the user's hand as shown in Fig. 5). In our model, from T_A on, the attacker can neither replace nor modify the cookie that the user holds. We denote the relative difference between T_A and T_{B1} with Δt_{AB}, that is, $\Delta t_{AB} \triangleq |T_A - T_{B1}|$. Thus, Δt_{AB} represents the delay with which the smartphone's camera detects the act of cookie leaving the terminal. Again, for security reasons (discussed in Sections 4.1 and 4.2), we limit Δt_{AB} to be smaller than some predefined value Δt_{AB}^*. The smartphone aborts the protocol if $\Delta t_{AB} > \Delta t_{AB}^*$.

The time periods Δt_B and Δt_{AB} together represent the time window during which the fortune cookie resides in the untrusted zone. Later in this section we provide a detailed characterization of Δt_B and Δt_{AB}.

3.1. The window of opportunity for the attacker

Without knowing the secret nonce R, the user cannot authenticate successfully with the terminal (as we prove in the Appendix). Therefore, to be successful, the relaying attacker has to somehow transfer R from the issuing terminal over to the honest user. The attacker has only two possibilities: (i) to physically transport the original fortune cookie, holding the secret nonce R, between the two honest parties (*the transport attack* - Fig. 2(a)), or (ii) to guess and/or somehow read a part or the whole nonce R from the original fortune cookie and clone it (produce a fake cookie holding the original nonce R) at the user's side (*the cloning attack* - Fig. 2(b)). Both attacking strategies are subject to the time constraints $\Delta t_B \leq \Delta t_B^*$ and $\Delta t_{AB} \leq \Delta t_{AB}^*$. We stress here that the attacker can potentially gain an extra time by mounting a successful *simulability attack* (Definition 1).

Fig. 5 shows the attacker's action timeline. We use T_0 to represent the moment when the attacker begins his relaying activity. In our model we assume $T_0 \geq T_{B0}$. As shown in Fig. 5, until the moment T_{B0}, the random nonce R resides in the terminal's trusted zone and hence is not available to the attacker. The honest user begins pulling the fortune cookie (original or fake, depending on the attacking strategy) from the fake terminal at the moment $T_{\widehat{B}0}$; we use "hat" to signify the presence of the attacker. Likewise, the time $T_{\widehat{B}1}$ is the moment at which the fortune cookie (original or fake) completely leaves the fake terminal. Using this timing model, the time available to the attacker for

the cloning and/or transport attack equals $(T_A - T_0)$. From Fig. 5, we can see that the following holds:

$$T_A - T_0 = \Delta t_{t/c} + (1-\rho)\cdot \Delta t_{\widehat{B}} + \Delta t_{A\widehat{B}}, \tag{1}$$

where $\Delta t_{t/c}$ is the time the attacker invests in relaying activities (e.g., unfolding the fortune cookie, physically transferring it, printing and creating a fake cookie, any other relaying action), $\Delta t_{\widehat{B}} \triangleq T_{\widehat{B}1} - T_{\widehat{B}0}$ is the cookie extraction time (by the honest user) at the fake terminal, and $\Delta t_{A\widehat{B}}$ is the delay introduced by the user's smartphone camera (not controlled by the attacker). The factor $0 \le \rho \le 1$ is used to mathematically represent the possibility that the attacker's relaying/cloning activities can take place in parallel/simultaneously with the fortune cookie extraction by the honest user. We call ρ the *simultaneity factor* and define it as follows:

$$\rho \triangleq \frac{\Delta t_{t/c} - \left(T_{\widehat{B}0} - T_0\right)}{\Delta t_{\widehat{B}}}. \tag{2}$$

For example, $\rho = 1$ implies simultaneous activities by the attacker and the honest user. It also implies longer $\Delta t_{t/c}$ (i.e., $\Delta t_{t/c} = T_{\widehat{B}1} - T_0$) for the given $(T_A - T_0)$, or in other words, more time available for the attacker. On the other hand $\rho = 0$ gives $\Delta t_{t/c} = T_{\widehat{B}0} - T_0$ and implies serial actions (first the attacker then the honest user). Using the definitions of Δt_B and Δt_{AB}, the bound $\Delta t_{AB} \le \Delta t^*_{AB}$ and the expression (1), we get the following bound on $\Delta t_{t/c}$ - the time window during which the attacker can mount the transport or cloning attack (conditional on no simulability attacks):

$$\Delta t_{t/c} \le \left(\Delta t^*_{AB} - \Delta t_{A\widehat{B}}\right) + \left[\Delta t_B - (1-\rho)\cdot \Delta t_{\widehat{B}}\right] - (T_0 - T_{B0}). \tag{3}$$

We will use the expression (3) to reason about different attacking strategies and their implications in the context of the transport and cloning attacks. Detailed analysis is deferred until Section 4. Before moving on with this analysis, we first characterize the time periods Δt_B, $\Delta t_{\widehat{B}}$, Δt_{AB} and $\Delta t_{A\widehat{B}}$ that appear in the Eq. (3).

3.2. Distribution of the cookie extraction time (Δt_B)

The fortune cookie extraction time Δt_B reflects the speed with which users can pull the fortune cookie out of a terminal. Thus, Δt_B is a (user-dependent) random variable. In order to assess the distribution of Δt_B, as well as other aspects of the Force protocol, we conducted an extensive user performance study (with 54 participants). In this section, we only present the results related to Δt_B, while in Section 6 we give other results obtained from the study. Fig. 4(b) shows the cumulative frequency distribution of Δt_B; the results obtained from the study were scaled to 2 cm long physical representation of the nonce R, i.e., 2 cm long QR code. As can be seen in Fig. 4(b), in 90% of cases, the tested users extracted the fortune cookie within 120 ms. For this reason, we upper bound Δt_B by $\Delta t^*_B = 120$ ms. If the terminal measures the cookie extraction time to be greater than 120 ms it will abort the protocol. Recall, this bound is obtained for 2 cm long physical representation of the nonce R. With smaller physical representations of the secret nonce we expect shorter extraction times and the smaller bound Δt^*_B.

3.3. Characterization of the delay Δt_{AB} caused by a camera

As discussed above, Δt_{AB} is the time difference between the moment T_A when the user's smartphone recorded the act of the cookie leaving the terminal and T_{B1} (the actual time at which the cookie left the honest terminal, Fig. 5). Given that the smartphone and the honest terminal are synchronized (as we prove in the Appendix), the delay Δt_{AB} is influenced primarily by the limited frame rate of the smartphone's camera and the related motion blur effects (due to the moving cookie). As discussed in Section 2, the smartphone learns/extracts T_A from the frame selected by the user during the event synchronization phase. In the example shown in Fig. 6(a), the user would normally select the 4th frame, where she can clearly see the cookie completely outside of the terminal. This is in spite of the fact that the cookie already left the terminal at the time corresponding to an earlier frame (Fig. 6(a)). It is the motion blur that introduces this uncertainty in the user's decision and thus in the delay Δt_{AB}. In what follows, we will show that for a standard 25-30 fps camera, Δt_{AB} is smaller than 70 ms (or about 1.75 video frames) with a high probability.

Motion blur occurs when a recorded object (the cookie in our case) moves during the camera exposure time or the camera moves itself (we do not consider this case in our model). We use the following motion blur model [27]. The opaque object (the fortune cookie) moves (from left to right) during the process of image capture. In the captured image, the cookie's boundary on the left and right partially occludes the background, and is shown as *semitransparent*

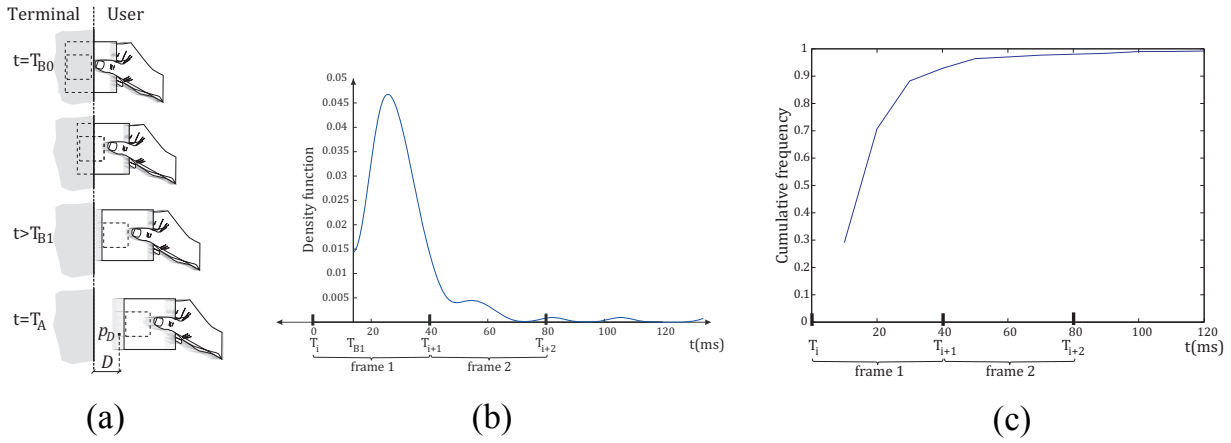


Fig. 6. Motion blur effects on the delay Δt_{AB} (and $\Delta t_{A\widetilde{B}}$): (a) A sequence of motion blurred frames as seen by the user on her smartphone, (b) estimate of the distribution of the time at which the cookie reaches the distance $D = 0.5$ cm from the terminal, and (c) the cumulative frequency distribution of $\Delta t(D)$, with $D = 0.5$ cm.

(Fig. 6(a)). The transparency for each image pixel is determined by the proportion of the time that the background is exposed to the camera. Now, the resulting captured image I is a linear combination of the foreground image F (the moving cookie) and the background image B (e.g., the floor or the terminal) through the *alpha channel*, that is, for the given image pixel p we have: $I(p) = \alpha(p)F(p) + (1 - \alpha(p))B(p),\ \alpha(p) \in \{0, 1\}$. Here, $\alpha(p)$ corresponds to the fraction of the exposure time/frame during which the moving object projects to (is captured by) the pixel p. Thus, $\alpha(p)$ close to 1 implies that the pixel p captures only the cookie (p is not mixed with the background), whereas lower positive $\alpha(p)$ values result in the pixel p blended to the background.

We use the following *user model* to describe the user's behavior. We assume that the user will see (on her smartphone) the cookie completely detached from the terminal when the alpha value $\alpha(p_D)$ of the image pixel p_D that captures the physical scene at the distance D from the terminal, is smaller than some predefined value α_D, i.e., $\alpha(p_D) < \alpha_D$ (see Fig. 6(a), the last frame). Let us denote the frame selected by the user (who follows the above frame-selection convention) as frame j. The frame j begins at T_j and lasts Δt_f; we assume all frames to be of the same duration. It follows from our user and motion blur models that the cookie projected to the pixel p_D during at most the initial α_D fraction of the frame j. Then we know that the cookie must have already left the terminal at the time $(T_j + \alpha_D \Delta t_f)$. We conveniently take this time as T_A, i.e., $T_A = T_j + \alpha_D \Delta t_f$.

To be more specific, let us set D to 0.5 cm in our user model. In Fig. 6(b) we overlay the probability density function of the time $\Delta t(D)$ it takes the cookie (pulled out by the user) to travel the distance $D = 0.5$ cm, on top of two video frames (i and $(i + 1)$) of duration $\Delta t_f = 40$ ms. The distribution of $\Delta t(D)$ is obtained by scaling the distribution of the cookie extraction time Δt_B to 0.5 cm. As shown in Fig. 6, the terminal recorded that the cookie left it at T_{B1}; T_{B1} happened to fall in the exposure interval of the frame i (i.e., $T_i < T_{B1} < T_{i+1}$). The pulled cookie will reach the distance $D = 0.5$ cm at the moment equal to T_{B1} plus the time $\Delta t(D)$ sampled from the overlayed distribution. Hence, the random time $(T_{B1} + \Delta t(D))$ will fall into either the frame i or some later frame, i.e., a frame $(i + k)$, with $k \in \{0, 1, \ldots\}$. Then, according to our user and motion blur models, the user will select the frame $(i + k)$ for which $(T_{B1} + \Delta t(D) - T_{(i+k)})\,/\Delta t_f < \alpha_D$ as the one that shows the cookie detached from the terminal. In turn, the smartphone will set $T_A = T_{(i+k)} + \alpha_D \Delta t_f = T_i + (k + \alpha_D)\Delta t_f$. In general, T_A can be expressed as follows:

$$T_A = T_i + (k + \alpha_D)\,\Delta t_f, \text{ with } k = \left\lceil \frac{T_{B1} - T_i + \Delta t(D)}{\Delta t_f} - \alpha_D \right\rceil . \tag{4}$$

Using the Eq. (4) and inequality ($\lceil x \rceil \leq x + 1$) , we can finally derive the bound on Δt_{AB} as follows.

$$\Delta t_{AB} \triangleq T_A - T_{B1} \leq \left(\frac{T_{B1} - T_i + \Delta t(D)}{\Delta t_f} - \alpha_D + 1 + \alpha_D \right) \Delta t_f + T_i - T_{B1} = \Delta t(D) + \Delta t_f \ .$$

Let us plug in some numbers into this bound to put it into perspective. The cumulative frequency distribution of $\Delta t(D)$ (for $D = 0.5$ cm) is shown in Fig. 6(c), from which we can read $P(\Delta t(D) \leq 30 \text{ ms}) > 0.88$. In addition, for 25 fps cameras we have $\Delta t_f \leq 40$ ms. Therefore, for the selected parameters, $\Delta t_{AB} \leq 70$ ms or about 1.75 video frames, with probability at least 0.88. Based on this argument we set Δt^*_{AB} in the expression (3) to 70 ms.

4. Security properties of a paper-based weakly unrelayable channel

In this section we show that the paper-based channel, in tandem with a smartphone, indeed satisfies the properties of weak untransportability, weak unclonability and unsimulability.

4.1. Weak untransportability

The weak untransportability property is required to mitigate the transport attack, in which the attacker physically relays a legitimate fortune cookie from the honest verifier to the fake one (Fig. 2(a)). The distance over which the legitimate cookie is relayed/transported is measured from the outermost point of the legitimate terminal; more precisely, we assume $T_0 = T_{B1}$ (Fig. 5). The immediate implication is that the bound (3) on $\Delta t_{t/c}$, the time window within which the attacker has to relay the original cookie, does not depend on Δt_B (the cookie extraction time at the legitimate terminal).

Consider now the moment $T_{\widehat{B}0}$ when the user begins pulling the cookie out of the fake terminal. We have two possibilities: (i) the original cookie must have already reached the fake terminal (the attacker's targeted location) at $T_{\widehat{B}0}$, or (ii) the user is initially pulling a fake cookie, to which the original cookie will be attached on-the-fly. We assume that, in both cases, the original cookie will leave the fake terminal at a speed proportional to the pulling force applied by the user (i.e., the attacker does not shoot the cookie at the user). Therefore, in the first case, the attacker will not transport the cookie any farther after $T_{\widehat{B}0}$, meaning that the simultaneity factor ρ in (3) must satisfy $\rho = 0$. In the second case, the attacker may still be relaying the original cookie after $T_{\widehat{B}0}$, but the honest user will take it over at certain moment during the cookie extraction time $\Delta t_{\widehat{B}}$, that is, we will have $\rho < 1$. Therefore, in the transport attack scenario, the inequality (3) simplifies to:

$$\Delta t_t \leq \Delta t^*_{AB} - \Delta t_{A\widehat{B}} - (1-\rho)\Delta t_{\widehat{B}} \text{ with } \rho < 1. \tag{5}$$

We drop the subscript c from $\Delta t_{t/c}$ in (3), as we consider only the transport attack in this subsection.

To put the bound (5) into perspective, let us plug-in some numbers. Let us take, conservatively, $\Delta t_{A\widehat{B}} = 0$, $\Delta t^*_{AB} = 70$ ms (derived in the previous section), $P(\Delta t_{\widehat{B}} > 25 \text{ ms}) > 0.9$ (see Fig. 4(b)) and $\rho = 0.5$ (i.e., the attacker relays the cookie during the first half of the cookie extraction period $\Delta t_{\widehat{B}}$). Then, $P(\Delta t_t < 57.5 \text{ ms}) > 0.9$. We argue that 57.5 ms is a way too short period for the attacker to physically move a piece of paper over a large distance and to load it into a fake terminal. In [15] the authors estimate the delay times induced by relaying NFC communication over various channels. For example, it is shown that the on-device access by an application to a secure element on a NFC-enabled phone takes between 50-80 ms. We therefore claim that a successful transport attack within 60 ms is unrealistic, even for very short distances.

Next we give a more precise characterization of distances over which the attacker can potentially transport the cookie within Δt_t. The attacker is clearly limited by certain physical laws and constraints such as the maximum velocity v_m of the cookie, its initial and terminal velocities (assumed zero in our case), as well as the maximum acceleration/deceleration a. It is well known that so called *bang-bang* strategy maximizes the distance that a cookie can travel within Δt_t, subject to the above constraints [28]. The bang-bang strategy yields the following solution for the maximum distance $d(\Delta t_t)$:

$$d(\Delta t_t) = \begin{cases} v_m(\Delta t_t - v_m/a) & \text{for } v_m \leq a\Delta t_t/2; \\ a\Delta t_t^2/4 & \text{otherwise .} \end{cases} \tag{6}$$

For example, for $v_m = 100$ m/s (360 km/h) and $a = 1000$ m/s^2, we can see that v_m is not attainable within $\Delta t_t = 57.5$ ms, so we use the second expression in (6) to obtain $d(\Delta t_t) \approx 83$ cm. It follows readily from the above expression that $d(\Delta t_t - 1 \text{ ms}) \approx 0.97 d(\Delta t_t)$. From this we can obtain that by decreasing Δt_t in our example by only 15 ms (e.g., through the system parameter Δt^*_{AB}), the relaying distance reduces from 83 cm to about 50 cm. We therefore claim

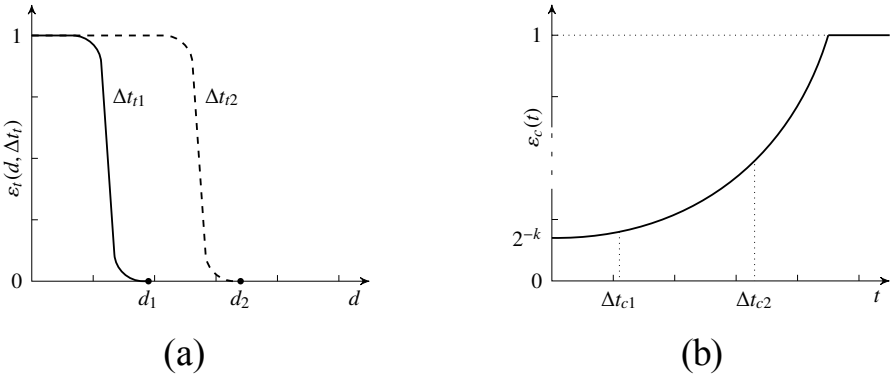


Fig. 7. Qualitative advantage of the attacker in: (a) the transport attack, and (b) the cloning attack.

that even with very advantageous assumptions for the attacker (i.e., a large acceleration and conservative bounds on Δt_t) the maximum relaying distance is rather limited.

Based on the presented model, we can define the advantage of the attacker in the transport attack as follows: *the advantage $\varepsilon_t(d, \Delta t_t)$ of the attacker is defined as the probability that he successfully relays/transports a fortune cookie over the fixed distance d within the time Δt_t.* In Fig. 7(a) we show $\varepsilon_t(d, \Delta t_t)$ qualitatively for two values of Δt_t, with the corresponding maximum relaying distances $d_1 = d(\Delta t_{t1})$ and $d_2 = d(\Delta t_{t2})$. Clearly, here we have $\Delta t_{t1} < \Delta t_{t2}$. Please note that $\varepsilon_t(d, \Delta t_t)$ satisfies another reasonable property, namely, $d^{'} > d$ implies $\varepsilon_t(d^{'}, \Delta t_t) \leq \varepsilon_t(d, \Delta t_t)$ for the fixed Δt_t. Please note that the weak untransportability property is directly related to the advantage $\varepsilon_t(d, \Delta t_t)$: *we say that a given paper-based channel satisfies the weak untransportability property if $\varepsilon_t(d, \Delta t_t)$ is satisfactorily small for the targeted distance d.*

4.2. Weak uncloneability

The cloning attack is a more general attacking strategy than the transport attack, in which the attacker guesses or reads the secret nonce R, relays R (using some fast channel) close to the victim (the honest verifier) and prints the nonce R on a fake paper (cookie). As before, the attacker has to execute the attack within the limited time given by the expression (3).

Using the expression (3), we can see that the optimal attacking strategy (timewise) is characterized by the following constraints: (i) $T_0 = T_{B0}$, that is, the cloning attack starts at the early stage of the fortune cookie extraction, (ii) $\rho = 1$, and (iii) $\Delta t_B = \Delta t_B^*$ (meaning that the attacker uses all the available time to extract the legitimate fortune cookie - fast extraction only reduces the available time). Inserting these constraints into the expression (3), it simplifies to:

$$\Delta t_c \leq \left(\Delta t_{AB}^* - \Delta t_{A\widehat{B}}\right) + \Delta t_B^* \,. \tag{7}$$

We drop the subscript t from the expression (3), since we consider only the cloning attack. *By comparing Δt_t (given in (5)) with Δt_c, we can see that timewise the cloning attack is preferable.*

We next discuss a potential work required on the side of the attacker to successfully mount the cloning attack within the time given by the expression (7):

• **Naive guessing attacker:** The simplest attack would involve guessing the secret nonce R and preparing in advance the fake fortune cookie. "All" the attacker has to do during the attack is to relay the legitimate messages transmitted over a regular channel (while making sure that the extraction of the legitimate fortune cookie and of the fake one is synchronized). While the work by the attacker is relatively low, the probability that he will succeed is only 2^{-k} (in a single attempt), where k is the size of the secret nonce R.

• **Very strong X-Ray attacker:** We next consider a very strong and sophisticated attacker, equipped with a X-Ray scanner, who tries to learn the content of the fortune cookie without actually opening it, for example using X-Ray

tomography [29]. We know, from the expression (7), that X-Ray recording and processing, relaying the learned nonce, printing it on the fake cookie and any other required action have to be accomplished within the time Δt_c. For $\Delta t^*_{AB} = 70$ ms (as derived in Section 3), $\Delta t_{A\widehat{B}} = 0$ and $\Delta t^*_B = 120$ ms (Section 3), the expression (7) evaluates to $\Delta t_c \leq 190$ ms. We claim that this is very challenging, since the WiFi relay path estimated in [15] introduces the delay in the range of 100 to 210 ms.

Even if we assume that we will face such sophisticated attackers in reality, we could fold the fortune cookie multiple times, or even better make it into a crumpled paper ball (as the one shown in Fig. 2) to significantly reduce the chance for the attacker to probe the interior of the fortune cookie [29]. Since Δt^*_{AB} and Δt^*_B are system parameters, we can always reduce them, at the cost of the increased rate of false positives.

• **Realistic cloning attacker:** A realistic attacker is equipped with a regular camera and controls a fake terminal. This attacker extracts the legitimate cookie from the legitimate terminal and manually unfolds it. The attacker then uses his camera to record a part or the whole secret nonce R, relays it to the fake terminal where R is printed on the fake fortune cookie. We conservatively assume that the attacker does not fold the cookie at the fake terminal; moreover, we assume that the honest user will not notice this (a human error).

For this attacker model, we are interested in the speed with which the attacker can manually unfold a fortune cookie. We motivated the students to give their best by rewarding the top three *fastest hands*; the first place reward was 40 USD, the second place 20 USD and the third place 10 USD. Overall 20 students were challenged to unfold different types of fortune cookies (one-, two- and three-times folded paper), and the results of this competition are presented in Fig. 8(a). The best result is 281 ms (for the one-time folded paper), which exceeds by far the conservative limit $\Delta t_c \leq 190$ ms. In Fig. 8(a) we plot the cumulative frequency distributions of fastest opening times and the CDF of the cookie extraction time Δt^*_B. It is easily seen that the gap between the CDF curve of Δt^*_B and the onset of the CDF curve for the 1-time folded paper is over 300 ms. Based on these experiments we conclude that this attacking strategy involving (manually) opening the cookie will fail with a high probability.

• **Powerful realistic cloning attacker:** The difference between this attacker and the previous one is that here we assume that the attacker holds more expensive cameras (e.g. high speed, pipe snake-like cameras, cameras that use very bright lights). With such equipment the attacker can increase the probability of a successful attack by trying to probe inside a closed cookie. However, as discussed above, we can reduce this probability by adjusting the way and the number of times the paper is folded. Likewise, we can reduce Δt^*_B (the maximum allowable extraction time) by reducing the size of the fortune cookie, or requiring a faster user's camera (thus decreasing Δt^*_{AB} and Δt_c).

Based on the previous analysis of the whole spectrum of possible attackers (ranging from naive to very strong ones) we can qualitatively represent the advantage of the cloning attacker as shown in Fig. 7(b). We can denote the advantage of the cloning attacker as: *the probability that an attacker successfully mounts the cloning attack within the time* Δt_c. It is interesting to observe the important difference between $\varepsilon_c(\Delta t_c)$ and the attacker's advantage $\varepsilon_t(d, \Delta t_t)$ in the case of the transport attack. Thus, $\varepsilon_c(\Delta t_c)$ does not depend on the relaying distance d, only on the available time Δt_c; this is because the cloning attacker does not relay any information over the special paper channel. Since the attacker can always try to guess the random nonce R (with probability 2^{-k}) the following must hold: $\varepsilon_c(\Delta t_c) \geq 2^{-k}$ for any $\Delta t_c > 0$. Moreover, following our previous discussion, it is reasonably to assume that $\varepsilon_c(\Delta t_{c1}) \leq \varepsilon_c(\Delta t_{c2})$ for $\Delta t_{c1} < \Delta t_{c2}$.

We say that a given special channel satisfies the weak unclonability property if $\varepsilon_c(\Delta t_c)$ *is satisfactorily small.*

4.3. Unsimulability

Unsimulability property refers to hardness of simulating some physical aspect of a special weakly unrelayable channel. In the context of our Force protocol, we realize such a channel by appropriately orchestrating the three elements: a piece of paper (fortune cookie), a smartphone and the end user. The user records the cookie extraction using her smartphone and in turn selects the frame (from the recorded video) that corresponds to the moment (T_A) at which the paper/cookie left the terminal. In this way, the user's smartphone learns time T_A which it uses (along with T_{B1}) to decide if the protocol has to be terminated or not. Consequently, the security of our protocol to some extent relies on the alertness of a user taking part in the protocol. We stress here that this is the only step in Force protocol where we rely on user's alertness.

The attacker can try to exploit this fact by tricking the honest user into selecting a wrong frame. For example, the attacker can color the outer half of the (fake) cookie using some dark (background) color. The honest user might

select this frame as the synchronizing frame, although the fake cookie has not yet left the fake terminal. In this way, the attacker gains the additional time to relay the content (the nonce R) of the legitimate cookie, while all the timing constraints will be met.

In the context of the Force *protocol the unsimulability property refers to the difficulty of mounting such attacks.* The advantage of the camera-based event-synchronization is that the user can review the extraction process (multiple times if needed) on her smartphone (e.g., by simply sliding through the recorded frames). The user will drop the ongoing session if she observes something unusual.

It is also important to emphasize that in our attacker model we assume that *the honest verifier always reads the same nonce as is printed on the extracted (fake) paper*. For example, the adversary cannot present an empty fake cookie initially (within the time frame given by the expression (3)) and later project (using a video projector) the original nonce R to it. We assume that such attempts will be noticed by the user.

We use ε_s to denote the probability that the user will be successfully deceived by the attacker who launches some form of the simulability attack. As we prove in the Appendix, the simulability attack increases the overall attacker's advantage against our protocol by the factor $\varepsilon_s \cdot (\varepsilon_t(d, \Delta t_{out}) + \varepsilon_c(\Delta t_{out}))$, where Δt_{out} is the allowable session duration. Please note that $\Delta t_t, \Delta t_c < \Delta t_{out}$; in other words, a successful simulability attack buys the attacker an extra time to perform either the transport or cloning attack. In the following section we state our main security result.

5. The main security result

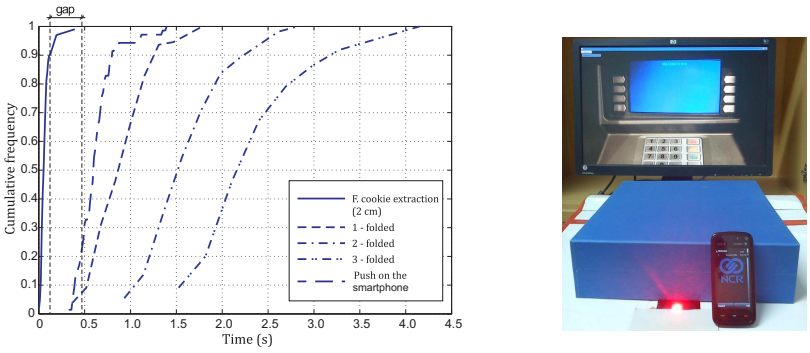
In this section, we first formally define what we mean by a secure (*mafia-fraud resistant*) protocol. Then we state our main security result in Theorem 1. We use two parameters $d \geq 0$ and $\varepsilon \in [0, 1]$ to characterize the resistance of the given protocol against the mafia fraud attack. The interpretation of these parameters at a high level is as follows: d represents the maximum distance over which a mafia fraud adversary is *successful* against the given protocol, with the probability at most ε. Definition 2 clarifies the meaning of *attacker being successful*.

Definition 2. *We say that a given protocol, executed between two honest parties A and B, is* (d, ε)-mafia fraud resistant*, if the following holds except for the probability* ε*: an honest party B* accepts *to provide the service (e.g., the terminal accepts a request for money as legitimate) at time instant* $(t + \Delta t_s)$ ***only if*** *(1) an honest party A (e.g., the user) has requested that service from B during period* $[t, t + \Delta t_s)$*, and (2) A and B have been within the distance d of each other during that period; here* Δt_s *represents the allowable session duration.*

The first condition implies that B has successfully authenticated the user A, while the second condition asserts that the user A was located in the close vicinity of the terminal B at the time she issued the request to B. Please note that the second condition in the definition does not imply that A and B have to be located within distance d of each other during the *whole* session period $[t, t + \Delta t_s)$. This last condition has to hold for only an arbitrary short, but non-null time period within $[t, t + \Delta t_s)$.

Note that we are not concerned with protecting the honest party A's privacy; we focus only on the authenticity of A's transaction details. In real practical scenarios, Force protocol can be implemented using regular *message authentication codes (MACs)*. If in addition the transaction privacy is required, we can replace the MAC with an appropriate *non-malleable authenticated encryption* (AE) scheme; *non-malleability* property appears to be essential for the security of our and similar distance bounding schemes.

Definition 2 is quite general as it characterizes an arbitrary secure authentication protocol (not necessarily designed with the mafia fraud attack in mind) as a mafia fraud resistant, but with potentially unfavorable parameters (d, ε). Using Definition 2 we could, for example, classify dedicated distance-bounding protocols utilizing nano-precision clocks (such as [22]) as $(10cm, \varepsilon)$-mafia fraud resistant, with $\varepsilon \ll 10^{-4}$. Please note that depending on spatial resolution of such protocols, the attacker could still mount the attack successfully if he stays within the bounds of the error margin (e.g., within 5-10 cm due to the clock drift). In the same vein, we argue that no existing solution can prevent attacks where an adversary tampers with a legitimate terminal and tricks the user into accepting it as being a genuine one. The adversary can for example cover the door at the terminal where the money goes out and wait until the honest party successfully authenticates with the terminal. If necessary, the adversary can also install a fake display over the authentic one. *In other words, we claim that* $(0, \varepsilon)$*-mafia fraud resistant protocols, with* $\varepsilon < 1$*, are hardly attainable, at best; even by employing strategies suggested in [22].*



(a) (b)

Fig. 8. (a) Manual cookie opening experiment representing cumulative frequency distribution of fortune cookie extraction, push on the smartphone after the extraction and unfolding a 1-folded, 2-folded and 3-folded fortune cookie. (b) The experimental setup used in our user performance study.

Table 1.
Summary of the users' demographics.

	Age				Use QR codes		Use m-banking		Use ATM	
	18-25	26-29	30-34	>35	Y	N	Y	N	Y	N
Pushbutton solution	9	3	2	0	7	7	2	12	11	3
Camera solution	10	6	2	2	16	4	4	16	16	4
Simulability attack	19	1	0	0	13	7	2	18	15	5

The best we can hope for in terms of protection against mafia-fraud attacks is to force the adversary to be very close to both legitimate parties. We cannot rely on end users to verify the authenticity of the terminal. Therefore in the present paper we consider the situations where the adversary does not try to modify in any way the legitimate terminal. However the adversary may install a new fake ATM/terminal in our attacker model.

Theorem 1. *The* Force *protocol is* (d, ε)-mafia fraud resistant, *with* $\varepsilon \leq q \cdot \left[(q+1) \cdot 2^{-k} + \varepsilon_t(d, \Delta t_t^*) + \varepsilon_c(\Delta t_c^*) + \varepsilon_s \cdot (\varepsilon_t(d, \Delta t_{out}) + \varepsilon_c(\Delta t_{out}))\right]$,

where $\varepsilon_t(\cdot)$, $\varepsilon_c(\cdot)$ and ε_s are the probability of the successful transport, cloning and simulability attack, respectively, d is the relaying distance, Δt_t^* and Δt_c^* are equal to the bound in the expression (5) and (7), respectively (Section 4), Δt_{out} the maximum session duration, while q is the number of oracle calls made by the attacker. The proof of Theorem 1 is available in the Appendix.

6. User performance study

In this section our goal was to estimate the cost of each protocol phase in terms of time (and effort) required to execute them by the end user. We implemented the Force protocol (Fig. 8(b)) with two alternative event-synchronization methods: (i) the *pushbutton* method (Section 2) and (ii) the *camera-based* method (Section 2). Our simulator (Fig. 8(b)) consists of a part representing an ATM terminal and a smartphone device (Nokia 5800). The terminal was emulated using a 22" monitor and an optical mouse both attached to HP 6730b laptop. The optical mouse was used for the purpose of precise measurement of the duration of the paper extraction process (i.e., the time instants T_{B0} and T_{B1}), as seen in Fig. 8(b). In most of our tests, the testers extracted the same (2 times folded) fortune cookie of the fixed size - 10 cm by 5 cm. The fortune cookie contained 2 cm by 2 cm QR code. During the tests the fortune cookie was placed inside the improvised cookie dispenser, 8 cm inside the outer edge of the dispenser and 2 cm outside the edge. We used Bluetooth wireless technology as a primary communication channel between the terminal and the smartphone. The software services used in our simulator were implemented in Java.

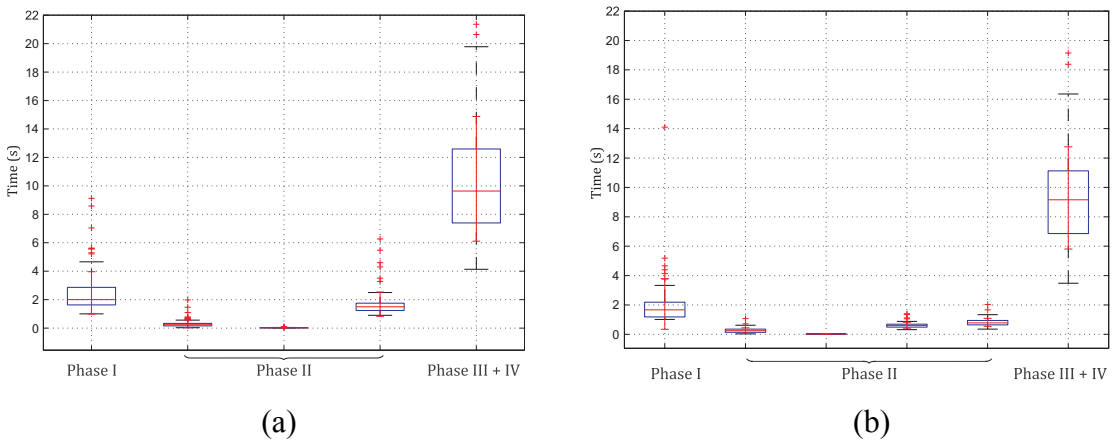


Fig. 9. Box plots representing the average protocol execution time for each phase: (a) the *pushbutton*-based event-synchronization (Phase I - Initialization, Phase II - Cookie extraction, clock synchronization, push on the button after cookie extraction, pushing the button twice (for error prevention), Phase III - Manual opening of the cookie and QR code detection, and Phase IV - transmission of the last protocol message (b) the *camera-based* event-synchronization (differs only in Phase II - Cookie extraction, clock synchronization, selection of the event-synchronization frame).

Table 1 summarizes the users' age, their experience in the use of QR codes, m-banking and ATM terminals. The tests were voluntary and the testers were recruited among the students and staff of our university. Before proceeding to each test, the users were briefly introduced to the concept of mafia-fraud attacks and explained how our protocol is secure against them. During the evaluation of both the *pushbutton* and *camera*-based event-synch methods, the users were asked to successfully run the Force protocol for five times. There were no economical incentives provided to the testers in order to achieve smaller error rates and faster protocol execution times.

6.1. Results of the pushbutton study

In the pushbutton event-synchronization method, the smartphone finds the moment T_A by having the user push the button once she observes that the fortune cookie completely left the terminal dispenser. For error prevention, we asked our testers to make 3 consecutive button pushes. Fig. 9(a) shows the average execution times taken by each phase of the Force protocol. The average overall protocol execution time is 13.08 sec (with std. = 4.09 sec). Here we do not consider the time it takes for the user to log into the smartphone and prepare the desired transaction. As shown in Fig. 9(a), the major part of the protocol execution time falls on the Phase III: the paper unfolding, QR code detection and secret nonce verification (average time 9.28 sec with std. = 3.47 sec). The average times taken to complete the initialization, pulling out the paper, clock synchronization and pushing of the button were 2.06 sec (std. = 1.72 sec), 0.271 sec (std. = 0.17 sec), 22.65 ms (std. = 10.66 ms) and 0.62 sec (std. = 0.201 sec), respectively. The slowest part of the protocol is the QR code detection and reading, however, modern smartphones have better QR code processing capabilities, so this delay could be significantly reduced.

Error rates. During our tests, two testers pressed the synch button while the cookie was still (partly) inside the terminal. We ran another series of tests to understand to what extent we can rely on users' alertness. We tried to trick the testers into pushing the button while the cookie was still inside the terminal, thus giving the attacker more time to mount the relay attack. To accomplish this we asked each tester to run the protocol 10 times in a row under normal conditions (i.e., with regular-sized cookie (10 cm) and no attacks). In the final run we replaced the original cookie with a longer one (around 30 cm). As a result we were able to trick 5 out of 20 testers (25%), thus proving that the pushbutton event-synch method is not robust enough.

6.2. Test results of the camera study

In the camera-based event synchronization, the user selects the first frame on which she sees that the fortune cookie left the terminal, and thus defines the moment T_A. In our implementation of this method, the testers were presented

with a randomly selected sequence of frames showing the recorded cookie extraction process (these were previously recorded and inserted into the smartphone). Fig. 9(b) shows the average execution times taken by each phase of the Force protocol. The overall time to complete the protocol was 13.28 sec on average (std. = 4.73 sec). The major part of the induced delay in this method is also due to the paper unfolding and QR code detection and decoding (the average time was 8.81 sec with std. = 4.41 sec). The average time taken to complete the initialization, pulling out the fortune cookie, clock synchronization and the frame selection were 2.47 sec (std. = 1.48 sec), 0.293 sec (std. = 0.27 sec), 24.48 ms (std. = 14.18 ms) and 1.68 sec (std. = 0.85 sec), respectively.

Error rates. The most important result obtained from this study is that all the testers correctly selected the first frame and therefore executed the protocol without any errors. This is important as it shows that the only part of the protocol that relies on the user's alertness can be made robust against human errors at a reasonably low additional cost (the users have to record the extraction process).

7. Conclusions

In this paper we have proposed a multichannel authentication protocol Force, a practical solution against *mafia fraud attacks* suitable for financial transaction that involve (paper) receipts. Force is inspired by distance-bounding protocols and the work of Stajano et. al. [23] on unrelayable channels, but designed to work with a regular radio and a special weakly unrelayable (paper-based) channel.

Our solution requires minimal or no hardware changes to the existing equipment and is independent of the underlying contactless (e.g., being NFC, WiFi, Bluetooth, Infrared) or contact-based technology (e.g., intra-body communication - IBC). The proposed solution is suitable for authorization of a high-risk financial transactions in the unknown environment (e.g., withdrawal of large sums of money from a terminal in a foreign country). We have shown that in some realistic attacking scenarios, even an unfolded paper (fortune cookie) may provide a good protection against relay attacks, which would make our solution highly practical and user-friendly.

The results of user performance study indicate that the proposed solution has reasonably low execution time (around 13 seconds on average), minimal error rate and is easy to understand. In our future work we also plan to investigate the usability of an accelerometer based solution for the paper detection with the smartphone.

References

[1] Spanish Bank Installs 'First' Contactless ATMs, http://www.nfctimes.com/news/spanish-bank-installs-first-contactless-atms, [Online; last access 5-November-2013].

[2] Nexus S Android Smartphone, http://www.samsung.com/us/mobile/cell-phones/GT-I9020FSTTMB/, [Online; last access 5-November-2013].

[3] VeriFone, Zoosh, http://www.verifone.com/industries/taxi/way2ride/, [Online; last access 5-November-2013].

[4] NCR makes wireless withdrawals in under 10 seconds at the ATM, http://www.ncr.com/newsroom/resources/mobile-cash-withdrawal-news/, [Online; last access 5-November-2013].

[5] Y. Desmedt, C. Goutier, S. Bengio, Special Uses and Abuses of the Fiat-Shamir Passport Protocol, in: CRYPTO '87: A Conference on the Theory and Applications of Cryptographic Techniques on Advances in Cryptology, Springer-Verlag, 1988, pp. 21–39.

[6] S. Drimer, S. J. Murdoch, Keep Your Enemies Close: Distance Bounding Against Smartcard Relay Attacks, in: Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium, USENIX Association, 2007, pp. 7:1–7:16.

[7] Y. Desmedt, Major Security Problems with the 'Unforgeable' (Feige)-Fiat-Shamir Proofs of Identity and How to Overcome Them, in: Proceedings of the 6th Worldwide Congress on Conference on Computer and Communications Security and Protection (SecuriCom), 1998.

[8] R. Anderson, M. Bond, The Man-in-the-Middle Defence, in: Cambridge Security Protocols Workshop, Springer-Verlag, 2009, pp. 153–156.

[9] J. H. Conway, On Numbers and Games, 2nd Edition, AK Peters, Ltd., 2000.

[10] Y.-C. Hu, A. Perrig, D. B. Johnson, Wormhole Attacks in Wireless Networks, IEEE Journal on Selected Areas in Communications 24 (2) (2006) 370–380.

[11] C. Cremers, K. B. Rasmussen, B. Schmidt, S. Čapkun, Distance Hijacking Attacks on Distance Bounding Protocols, in: 33rd IEEE Symposium on Security and Privacy, S&P 2012, IEEE Computer Society, 2012, pp. 113–127.

[12] L. Francis, G. Hancke, K. Mayes, K. Markantonakis, 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, RFIDSec'10, Springer-Verlag, 2010, pp. 35–49.

[13] K. M. L. Francis, G. Hancke, K. Markantonakis, Practical Relay Attack on Contactless Transactions by Using NFC Mobile Phones, Cryptology ePrint Archive, Report 2011/618 (2011).

[14] G. Hancke, A Practical Relay Attack on ISO 14443 Proximity Cards, Tech. rep., Technical report (2005).

[15] M. Roland, Applying Recent Secure Element Relay Attack Scenarios to the Real World: Google Wallet Relay Attack, Cryptology ePrint Archive, Report 2011/618, http://eprint.iacr.org/2011/618 (2011).

[16] S. Gezici, Z. Tian, G. B. Biannakis, H. Kobayashi, A. F. Molisch, H. V. Poor, Z. Sahinoglu, Localization via Ultra-wideband Radios: A Look at Positioning Aspects for Future Sensor Networks, IEEE Signal Processing Magazine 22 (2005) 70–84.
[17] G. P. Hancke, M. G. Kuhn, An RFID Distance Bounding Protocol, in: Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks, SECURECOMM '05, IEEE Computer Society, 2005, pp. 67–73.
[18] N. O. Tippenhauer, S. Čapkun, ID-based Secure Distance Bounding and Localization, in: Proceedings of the 14th European conference on Research in computer security, ESORICS'09, Springer-Verlag, 2009, pp. 621–636.
[19] S. Capkun, J.-P. Hubaux, Secure Positioning of Wireless Devices with Application to Sensor Networks, in: IEEE INFOCOM, 2005.
[20] G. P. Hancke, Design of a Secure Distance-Bounding Channel for RFID., J. Network and Computer Applications 34 (3) (2011) 877–887.
[21] A. Francillon, B. Danev, S. Čapkun, Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars, in: Network and Distributed System Security Symposium (NDSS), 2011.
[22] K. B. Rasmussen, S. Čapkun, Realization of RF Distance Bounding, in: Proceedings of the 19th USENIX conference on Security, USENIX Security'10, USENIX Association, 2010, pp. 389–402.
[23] F. Stajano, F.-L. Wong, B. Christianson, Multichannel Protocols to Prevent Relay Attacks, in: Proceedings of the 14th international conference on Financial Cryptography and Data Security, FC'10, 2010, pp. 4–19.
[24] S. Ganeriwal, C. Pöpper, S. Čapkun, M. B. Srivastava, Secure Time Synchronization in Sensor Networks, ACM Transactions on Information and System Security (TISSEC) 11 (4) (2008) 23:1–23:35.
[25] Sony Digital Network Applications. Inc., Frame Grabber, http://www.sonydna.com/sdna/e/products/framegrabber/index.html/, [Online; last access 5-November-2013].
[26] BlackBerry 10, Time Shift, http://us.blackberry.com/campaigns/blackberry-10.html/, [Online; last access 5-November-2013].
[27] A. Giusti, V. Caglioti, Isolating Motion and Color in a Motion Blurred Image, in: Proceedings of the British Machine Vision Conference 2007, British Machine Vision Association, 2007.
[28] E. H. Yilmaz, W. H. Warren, Visual Control of Braking: A Test of the tau Hypothesis, Journal of Experimental Psychology: Human Perception and Performance 21 (5) (1995) 996–1014.
[29] A. D. Cambou, N. Menon, Three-dimensional Structure of a Sheet Crumpled Into a Ball, Proceedings of the National Academy of Sciences of the United States of America 108 (36) (2011) 14741–14745.

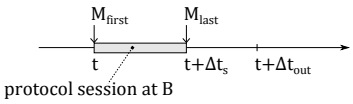


Fig. 10. Definition of a protocol session.

Appendix. Proof of Theorem 1

The timing model. We assume that genuine party B (as well as genuine party A) can participate in a single session at a time. From the party B's perspective, the protocol session begins at the time instant t at which it receives the first/initial protocol message (denoted M_{first} in Fig. 10). We say that B accepts a service request from the honest party A at the time instant $(t + \Delta t_s)$ at which B receives the last message in the protocol (M_{last} in Fig. 10) - Δt_s thus represents the session duration. Please note that Δt_s does not account for the time it takes B to actually deliver the requested and accepted service (e.g., the time it takes to B to prepare and deliver the requested amount of money). In case of any inconsistencies or errors in the ongoing session (authentication failures, missing/incomplete messages, etc.), B does not accept and finishes the current session. The maximum session duration is bounded by Δt_{out}, i.e., $\Delta t_s \leq \Delta t_{out}$. This is the latest time at which B can accept a service request in the ongoing session.

Advantage of a mafia fraud attacker. Using Definition 2, we can characterize a successful mafia fraud attacker as follows. First, we observe that the honest party B provides the service only if the requestor's identity in the initial message that B receives belongs to a legitimate user. In practice, the requestor ID is used by B to lookup the corresponding authentication credentials. Thus it is pointless for the attacker to make requests with non-existing IDs[2].

For a given security protocol SP and a fixed relaying distance d, the advantage $\mathbf{Adv}^{mafia}_{\mathsf{SP}}(d)$ of a mafia fraud attacker against SP is the probability that any of the following holds: *the honest party B accepts a service request with a legitimate requester identifier ID_A at some time instant $(t + \Delta t_s)$ while (1) the honest party A with identifier ID_A has not output the service request during period $[t, t + \Delta t_s)$, or (2) the requesting honest party A (with identifier ID_A) was located outside the distance d from B throughout the session period $[t, t + \Delta t_s)$.*

Therefore, for a given (d, ε)-mafia fraud resistant protocol we have $\mathbf{Adv}^{mafia}_{\mathsf{SP}}(d) = \varepsilon$.

Bounding the advantage $\mathbf{Adv}^{mafia}_{\mathsf{Forces}}(d)$. We use *the random oracle model* for this purpose. In our protocol, $[m]_f$ denotes a pair $(m, f(m))$ where $m \in \{0,1\}^*$ and f is a random function $f : \{0,1\}^* \rightarrow \{0,1\}^k$ (k being the size of the key and random nonces); f on each different input returns a bit string uniformly distributed on $\{0,1\}^k$ (on the same input f returns the same output). In our model, genuine parties B_j (e.g., different ATMs) are all assumed to be under the control of a single trusted entity (e.g., a bank) with which each honest party A_i shares a common random function f (a unique secret K_{A_iB} in the real world); hence, each party B_j also has an access to the f oracle. Clearly, the adversary is not given access to the f oracle directly, but only through interaction with honest parties A_i and B_j. All legitimate parties are assumed to have unique identities.

To bound $\mathbf{Adv}^{mafia}_{\mathsf{Forces}}(d)$, we fix a session (i.e., fix A, B and a time instant t) and first bound the attacker's advantage $\mathbf{Adv}^{mafia(A,B,t)}_{\mathsf{Forces}}(d)$ in this fixed session. By assumption, the attacker can participate in at most q sessions, from which we have: $\mathbf{Adv}^{mafia}_{\mathsf{Forces}}(d) \leq q \cdot \mathbf{Adv}^{mafia(A,B,t)}_{\mathsf{Forces}}(d)$. Following the Definition 2, we use B to denote the event that the honest party B accepts, and we use C_1 and C_2 to denote that the first and the second condition, respectively, from the definition hold true, all this in the fixed session (A, B, t). Then we can express $\mathbf{Adv}^{mafia(A,B,t)}_{\mathsf{Forces}}(d)$ as follows:

$$\begin{aligned}\mathbf{Adv}^{mafia(A,B,t)}_{\mathsf{Forces}}(d) \triangleq \mathbf{P}\left(\overline{C}_1 \vee \overline{C}_2 | B\right) &= 1 - \mathbf{P}\left(C_1 \wedge C_2 | B\right) \\ &= 1 - \left(1 - \mathbf{P}\left(\overline{C}_2 | C_1 B\right)\right) \cdot \left(1 - \mathbf{P}\left(\overline{C}_1 | B\right)\right) \\ &\leq \mathbf{P}\left(\overline{C}_2 | C_1 B\right) + \mathbf{P}\left(\overline{C}_1 | B\right) . \end{aligned} \tag{8}$$

The proof proceeds in three steps. We first establish that if the honest party B accepts, the honest parties A and B must have mutually authenticated each other (as well as all the messages they have exchanged) - except with a negligible

[2] We are not concerned with possible DoS attacks here.

probability. In other words, we show $\mathbf{P}\left(\overline{C}_1|B\right)$ in the bound (8) to be negligible. In the second step we establish that honest parties A and B are mutually synchronized at the end of the session (except with a negligible probability). Finally, in the third step, building upon the facts that A and B are mutually authenticated and synchronized, we prove that A and B must have been located within the relaying distance d at some time during the given session (except with a satisfactorily small probability $\mathbf{P}\left(\overline{C}_2|C_1B\right)$).

Step 1: If B accepts, then A and B have mutually authenticated each other, as well as messages m_1 and m_2. Fix A, B and a time instant t. At t, B receives the first protocol message comprising an identifier ID_A of the honest party A. The identifier must belong to the honest party, otherwise B will simply abort the protocol[3]. By the protocol, B accepts if it receives (and successfully verifies) $[B, R, m_2]_f$ at time instant $(t + \Delta t_s)$, with $\Delta t_s \leq \Delta t_{out}$. If no honest party previously output $[B, R, m_2]_f$, no party previously submitted $[B, R, m_2]$ to the f random oracle. Therefore, the probability that the attacker can compute correctly $[B, R, m_2]_f$ in this session is bounded by 2^{-k}. Consider now the case where some honest party did output $[B, R, m_2]_f$. The form of the message $[B, R, m_2]_f$ implies that it must be honest party $A^{'}$ (not a terminal) that received R in the corresponding fortune cookie. The probability that this happened before the moment T_{B0} ($T_{B0} \in (t, t + \Delta t_s)$) at which B output (in the present session) the random nonce R (in the fortune cookie) is at most $q_1 \cdot 2^{-k}$, where q_1 is bounded by the number of calls made to the f random oracle by that time.

Next, we establish that $A^{'} = A$ by means of contradiction. Let us assume that $[B, R, m_2]_f$ was output by $A^{'} \neq A$ at some time $(t, t + \Delta t_s)$. By the protocol, $A^{'}$ must have received $[A^{'}, B, N_A^{'}, N_B^{'}, m_1^{'}]_f$ at some earlier time. If no honest party previously output $[A^{'}, B, N_A^{'}, N_B^{'}, m_1^{'}]_f$, no party previously submitted $[A^{'}, B, N_A^{'}, N_B^{'}, m_1^{'}]$ to the f random oracle. Therefore, the probability that the attacker can compute correctly $[A^{'}, B, N_A^{'}, N_B^{'}, m_1^{'}]_f$ is bounded by 2^{-k}. Consider now the case where some honest party did output it. The form of the message $[A^{'}, B, N_A^{'}, N_B^{'}, m_1^{'}]_f$ implies that it must be honest party B (the terminal). The probability that this happened before the moment $t(N_A^{'})$ at which the honest party $A^{'}$ output the random nonce $N_A^{'}$ is at most $q_2 \cdot 2^{-k}$, where q_2 is bounded by the number of calls made to the f random oracle by that time.

Claim 1. *$t(N_A^{'}) \geq t(R)$ where $t(R) \triangleq T_{B0}$ and $t(R) \in (t, t + \Delta t_s)$.*

Proof. Let us assume by contradiction that $t(N_A^{'}) < T_{B0}$, that is $t(N_A^{'}) < t(R)$. As $A^{'}$ is the honest party, the following holds by the protocol: $t(N_B^{'}) < t(N_A^{'}) < t(R)$, where $t(N_B^{'})$ is the moment at which honest B output $N_B^{'}$. Again, by the protocol, B must have output $R^{'}$ at some earlier time $t(R^{'})$. Therefore, $t(R^{'}) < t(N_B^{'}) < t(R)$. We already established above that $A^{'}$ output $[B, R, m_2]_f$ after $t(R)$ (i.e.T_{B0}), except with $(q_1 + 1) \cdot 2^{-k}$. But the condition $t(R^{'}) < t(R)$ implies that $A^{'}$ could have output $[B, R, m_2]_f$ with only a negligible probability $q_1 \cdot 2^{-k}$, leading to the contradiction. We conclude that $t(N_A^{'}) \geq T_{B0}$, that is, $t(N_A^{'}) \geq t(R)$. □

We established above that the message $[A^{'}, B, N_A^{'}, N_B^{'}, m_1^{'}]_f$ was output by honest B and after the time $t(N_A^{'})$, except with $(q_2 + 1) \cdot 2^{-k}$. Combining this result and Claim 1, we have $t([A^{'}, B, N_A^{'}, N_B^{'}, m_1^{'}]_f) \geq t(N_A^{'}) \geq t(R)$. But honest B output only one message $[A, B, N_A, N_B, m_1]_f$ during $(t(R), t + \Delta t_s)$; honest B can participate in at most one session at a time, by design. Therefore, $A^{'} = A$, $N_A^{'} = N_A$ (since $[A, B, N_A, N_B, m_1]_f$ accepted by $A^{'}$), $N_B^{'} = N_B$ and $m_1^{'} = m_1$ (since $[A, B, N_A, N_B, m_1]_f$ output by B).

Summing up, the probability $\mathbf{P}\left(\overline{C}_1|B\right)$ that honest B accepts in a given/fixed session while no honest A requested the service in the same session is bounded by $(q_1 + q_2 + 2) \cdot 2^{-k}$. More precisely, $\mathbf{P}\left(\overline{C}_1|B\right) \leq q \cdot 2^{-k}$, where q (the number of oracle calls available to the attacker) satisfies $q = (q_1 + q_2 + 2)$.

Step 2: At the end of a successful session A and B are mutually synchronized within the system parameter Δt_{sync}^{*}. In the proof, we assume that the local clocks of A and B are of a reasonable quality - meaning that they drift apart by only a negligible time (e.g., up to a few microseconds) during some relative short time period (in the order of tens of milliseconds). We are interested only in an instantaneous (i.e., short term) synchronization and do not care about possible long term effects of clock drifts. This assumption greatly simplifies our proof, yet it does not severely affect the security of our protocol in the real world. These small errors (measured in μs) simply add to the overall synchronization error margin (measured in ms).

[3]The identifier is used by B to lookup the corresponding authentication credentials.

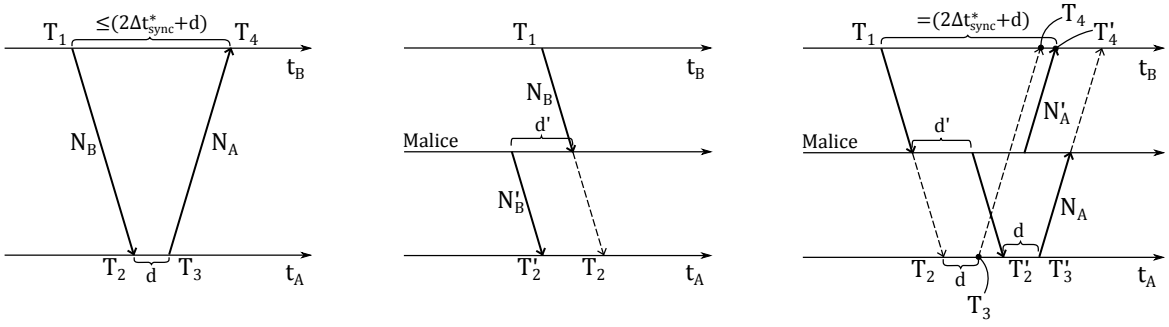


Fig. 11. (a) The end-to-end delay Δt_{sync} is a bound on the time of flight of the nonces N_A and N_B. The attacker (Malice) has only two possibilities when attacking the synchronization protocol: he can either (b) advance or (c) delay the delivery of the random nonces N_B and N_A.

In this step of the proof, we show that the probability $\mathbf{P}\left(\overline{S}|C_1 B\right)$ is negligible, where S denotes the event that the honest A and B are synchronized in the fixed session (A, B, t). Recall, C_1 and B imply the messages m_1 and m_2 to be authentic. In particular, given C_1 and B, the nonces N_A and N_B as well as the timestamps T_1 and T_4 are all successfully authenticated by the honest party A in the corresponding session.

Recall from Section 2, the honest party A, having received the authentic message $[A, B, N_A, N_B, m_1]_f$, with $m_1 = (T_{B1}, T_1, T_4)$, verifies that the *end-to-end delay* Δt_{sync} is within the error margin Δt^*_{sync}, which is a system parameter. More precisely, A has to verify that the following holds: $\Delta t_{sync} = [(T_4 - T_1) - (T_3 - T_2)]/2 \leq \Delta t^*_{sync}$. Referring to Fig. 11(a), we can see that Δt_{sync} is essentially the bound on the time of flight of the nonces N_A and N_B transmitted over a radio channel. Please note that we do not assume either A or B to be equipped with nano-precision clocks. The value of Δt_{sync} (and hence of Δt^*_{sync}) is essentially determined by the precision with which A and B (their operating systems) can timestamp a message transmission and reception. Please note that in line with the assumption of negligible clock drifts during the short synchronization phase, it is reasonably to assume that $\Delta t_{sync} \geq 0$. We do acknowledge that in the real world it may happen, due to anomalous drifts in the A's and B's local clocks, the end-to-end delay to be negative. In our proof, we can handle this by requiring that the honest party A runs the following verification before proceeding with the protocol: $|\Delta t_{sync}| \leq \Delta t^*_{sync}$.

Using the authentic timestamps T_1, T_2, T_3 and T_4, the party A can calculate the offset between its clock and the B's local clock as follows[4]:

$$offset_A = T_1 + \Delta t_{sync} - T_2 \quad . \tag{9}$$

In turn, A can adjust its local clock t_A as follows: $t_A = t_A + offset_A$. Using the fact that $0 \leq \Delta t_{sync} \leq \Delta t^*_{sync}$ we can bound $offset_A$ as follows:

$$T_1 - T_2 \leq offset_A \leq T_1 - T_2 + \Delta t^*_{sync} \; . \tag{10}$$

We are now ready to prove that an attacker cannot manipulate $offset_A$ by a larger amount than Δt^*_{sync}. Observe first that the attacker has no control over the timestamp T_1 and Δt^*_{sync}; the timestamp T_1 is controlled and singed by the honest party B (see Step 1) and Δt^*_{sync} is simply a fixed system parameter. Therefore, the attacker can only affect the timestamp T_2 in the bounds given by (10). The attacker has only two possibilities as shown in Figs. 11(b) and 11(c): either cause (i) a decrease or (ii) an increase of T_2. The attacker (Malice in Figs. 11(b) and 11(c)) can accomplish this by either advancing or delaying the delivery of the random nonces N_B and N_A. Thus, to cause the party A to timestamp an arrival of N_B with $T_2' < T_2$, the attacker has to transmit N_B before the party B has output it, see Fig. 11(b). As the

[4]Here we neglect possible clock drift that potentially accumulates during Δt_{sync}; Δt_{sync} is reasonably short and in addition our protocol can tolerate synchronization errors in the order of couple of milliseconds.

random nonces are authenticated by A, this means that the attacker has to essentially guess it, which can happen with the probability 2^{-k}.

Likewise, to cause the party A to timestamp an arrival of N_B with $T_2^{'} > T_2$, the attacker has to delay the transmission of N_B to A (e.g., the attacker jams A, receives N_B and re-transmits it with some delay $d^{'}$ as shown in Fig. 11(c)). It follows readily from the graphical argument shown in Fig. 11(c) that the attacker can delay the transmission of N_B by at most $d^{'} = \Delta t^*_{sync} - \Delta t_{sync}$, otherwise, the end-to-end delay Δt_{sync} will exceed the error margin Δt^*_{sync} and the party A will abort the protocol. The only way for the delay $d^{'}$ to go beyond $\Delta t^*_{sync} - \Delta t_{sync}$ without causing the protocol to fail, is to have the attacker transmit N_A before A has output it (see Fig. 11(c)). As the random nonces are authenticated by A, this means that the attacker has to essentially guess N_A, which can happen with the probability 2^{-k}. Overall, the probability that the attacker succeeds (in the fixed session) satisfy $\mathbf{P}\left(\overline{S}|C_1B\right) = 2^{-k}$, because a failure to guess correctly either N_B or N_A will result in the aborted protocol.

We conclude this part of the proof by observing that the parties A and B are synchronized within $|offset_A^{\text{actual}} - offset_A| \leq |\Delta t_{e2e} - \Delta t^*_{sync}| \approx \Delta t^*_{sync}$, where Δt_{e2e} is a true end-to-end delay, i.e., the time of flight of random nonces over a radio channel and $\Delta t_{e2e} \ll \Delta t^*_{sync}$.

Step 3: The requesting honest party A was located within the relaying distance d from the honest party B at some time during the session period $[t, t + \Delta t_s)$ (except with the probability $\mathbf{P}\left(\overline{C}_2|C_1B\right)$). In the final step, we bound the probability $\mathbf{P}\left(\overline{C}_2|C_1B\right)$. We observe the following:

$$\begin{aligned}\mathbf{P}\left(\overline{C}_2|C_1B\right) &= \mathbf{P}\left(\overline{C}_2S|C_1B\right) + \mathbf{P}\left(\overline{C}_2\overline{S}|C_1B\right) \\ &\leq \mathbf{P}\left(\overline{C}_2|SC_1B\right) + \underbrace{\mathbf{P}\left(\overline{S}|C_1B\right)}_{2^{-k}\text{ (Step 2)}} . \end{aligned} \tag{11}$$

Therefore, we wish to bound the probability $\mathbf{P}\left(\overline{C}_2|SC_1B\right)$. Recall, before signing and sending the random challenge R, A verifies that $|T_A - T_{B1}| \leq \Delta t^*_{AB}$ holds. As explained in Section 2, T_A is extracted from the video frame that the user A has selected in the Phase II of the Forces protocol. Let us denote with T_A^{act} the *actual time* at which the cookie left the terminal at the user A's side. We distinguish the following two exhaustive and mutually exclusive cases: (i) $|T_A^{act} - T_{B1}| \leq \Delta t^*_{AB}$ and (ii) $|T_A^{act} - T_{B1}| > \Delta t^*_{AB}$, which we denote as event T and $\overline{T}$, respectively. Since the clocks of A and B are synchronized and $|T_A - T_{B1}| \leq \Delta t^*_{AB}$ holds (conditional on the joint event (S, C_1, B)), the event $\overline{T}$ can happen only as a consequence of a user's mistake (selection of a wrong frame) or a successful *simulability* attack. We assume conservatively that the event $\overline{T}$ is always the consequence of a successful simulability attack against the human user - more formally $\mathbf{P}\left(\overline{T}|SC_1B\right) = \varepsilon_s$. Therefore, $\mathbf{P}\left(\overline{C}_2|SC_1B\right)$ in (11) can be bounded as follows:

$$\mathbf{P}\left(\overline{C}_2|SC_1B\right) \leq \mathbf{P}\left(\overline{C}_2|TSC_1B\right) + \mathbf{P}\left(\overline{C}_2|\overline{T}SC_1B\right) \cdot \underbrace{\mathbf{P}\left(\overline{T}|SC_1B\right)}_{\varepsilon_s} .$$

Therefore, we wish to bound the probabilities $\mathbf{P}\left(\overline{C}_2|TSC_1B\right)$ and $\mathbf{P}\left(\overline{C}_2|\overline{T}SC_1B\right)$ that the honest parties A and B were at least the distance d apart throughout the given session (A, B, t), during which (i) A and B mutually authenticated each other and the corresponding messages, (ii) A and B had their clocks synchronized, and (iii) the user A extracted the cookie at the time $T_A^{act} \in (t, t + \Delta t_s)$, with $|T_A^{act} - T_{B1}| \leq \Delta t^*_{AB}$ and $|T_A^{act} - T_{B1}| > \Delta t^*_{AB}$, respectively. Observe that the condition (i) implies that the smartphone A holds the same challenge R as the one generated by B in the session (A, B, t). Moreover, in our attacker model, the smartphone must have read R off the cookie (paper) that was extracted by the user A, all during the same session (A, B, t). Since the cookie extracted by A can be either exactly the one output by B or a different one, the following must hold: $\mathbf{P}\left(\overline{C}_2|TSC_1B\right) \leq \varepsilon_t(d, \Delta t^*_t) + \varepsilon_c(\Delta t^*_c)$, where $\varepsilon_t(\cdot)$ and $\varepsilon_c(\cdot)$ are the probability of a successful transport and cloning attack, and Δt^*_t and Δt^*_c are equal to the bound in the expression (5) and (7), respectively (Section 4). Likewise, $\mathbf{P}\left(\overline{C}_2|\overline{T}SC_1B\right) \leq \varepsilon_t(d, \Delta t_{out}) + \varepsilon_c(\Delta t_{out})$.

The theorem follows by summing up the probabilities obtained in each step and plugging the result back into $\mathbf{Adv}^{mafia}_{\text{Forces}}(d) \leq q \cdot \mathbf{Adv}^{mafia(A,B,t)}_{\text{Forces}}(d)$.