In-region Authentication

Mamunur Rashid Akand and Reihaneh Safavi-Naini

University of Calgary, Calgary, AB, Canada
{mdmamunurrashid.akan,rei}@ucalgary.ca

Abstract. Location information has wide applications in customization and personalization of services, as well as secure authentication and access control. We introduce *in-Region Authentication (inRA)*, a novel type of authentication, that allows a prover to prove to a set of cooperating verifiers that they are in possession of the correct secret key, and are inside a specified (policy) region of arbitrary shape. These requirements naturally arise when a privileged service is offered to registered users within an area. Locating a prover without assuming GPS (Global Positioning System) signal however, incurs error. We discuss the challenge of designing secure protocols that have quantifiable error in this setting, define and formalize correctness and security properties of the protocols, and propose a systematic approach to designing a family of protocols with provable security where error can be flexibly defined and efficiently minimized. We give a concrete instance of this family that starts with two verifiers, prove its security and evaluate its application to four different policy regions. Our results show that in all cases false acceptance and false rejection of below 6% can be achieved. We compare our results with related works, and propose directions for future research.

Keywords: In-region · Distance Bounding · Authentication.

1 Introduction

Location-based services (LBS) have provided exciting opportunities to use position related information such as location, proximity or distance in improving system security, control access, and personalized service delivery [13, 15, 28]. One of the earliest applications of user location is for securing authentication systems [8] against man-in-the-middle (MiM). Secure authentication protocols are challenge-response protocols between a *prover* and a *verifier*. In an MiM attack against these protocols, an attacker runs two simultaneous sessions of the protocol, one with an honest prover and one with the (honest) verifier, and by passing the responses of the prover to the verifier, succeeds in the protocol. Desmedt *et al.* [8] showed that this attack that does not have any cryptographic solution, can be prevented if the verifier uses an estimate of the location of the prover (e.g. distance to the verifier) as a second factor in authentication. Distance (Upper) Bounding (DUB) protocols [4] are challenge-response authentication protocols that provide cryptographic authentication security with the extra guarantee that the user is close to the verifier. These protocols have been widely studied, their

security has been formalized, and protocols with provable properties have been proposed [4, 3, 21, 9]. Successful authentication allows the user to perform a privileged action, e.g., open the car door [11], or access a special system resource.

In this paper we consider access control with respect to a region $\mathcal{R}$, that is called *policy region*. The user has to "prove" to the verifier(s) that, (i) they have the secret key k_u, and (ii) they are within the region $\mathcal{R}$. This setting naturally arises when a privileged service is offered in a region $\mathcal{R}$. For example a project team in a software development company can access proprietary project information when they are at their workspace. In this setting authentication protocol must prove the conjunction,

$$User \text{ has the shared secret } k_u \ \wedge \ (User \text{ is in } \mathcal{R}). \tag{1}$$

We propose a new authentication system that is called *in-Region Authentication (inRA)*, that provides proof that the above conjunction holds.

A simplistic solution to prove the conjunction (1) is to use a secure cryptographic authentication to allow the user to prove that they know the secret k_u, and then use a secure location verification protocol to prove their location. This solution however will be insecure because, firstly, proving the two clauses separately allows new attacks, for example the prover changing the location in between the two steps, and secondly, secure location verification protocols [18, 7] start with the prover claiming a location, which needs them to access GPS signal. This not only limits application of the protocol to locations where GPS signal is available, but also opens the possibility of GPS spoofing attacks [23].

One can combine the two steps when $\mathcal{R}$ is a circular region by employing a secure DUB protocol: the verifier of the DUB protocol will be placed at the center of the region and the distance bound will be chosen as the radius of $\mathcal{R}$ (Fig. 1a). The approach works perfectly because $\mathcal{R}$ is perfectly covered with the circle associated with the boundary of the DUB protocol. For arbitrary $\mathcal{R}$, one can use an *approximate cover* by using one or more verifiers (See Fig. 1b and 1c): the prover must prove its distance to the corresponding verifier of each part of the region. This is the approach in [16] to solve the closely related problem of *in-region location verification* where the goal is to verify that the prover is within the region, *without requiring authentication of the user* or quantifying error.

Using multiple verifiers to cover $\mathcal{R}$ requires one to determine the *verifier configuration*, which is specified by (i) the number of verifiers, (ii) their locations, and (iii) their associated distance bounds. Note that the error associated to a configuration does not have an algebraic form and one cannot use traditional optimization methods to find the optimal configuration, and this is true even if the number and location of verifiers are known.

Our work.

Model. Our goal is to design provably secure authentication protocols that allow the prover to prove conjunction (1), while minimizing protocol error. In Section 3 we formally define an inRA system for, a set of registered (provers) and set of unregistered users, a set of collaborating verifiers, and an inRA protocol whose correctness is defined using FA (False acceptance) and FR (False rejection) with

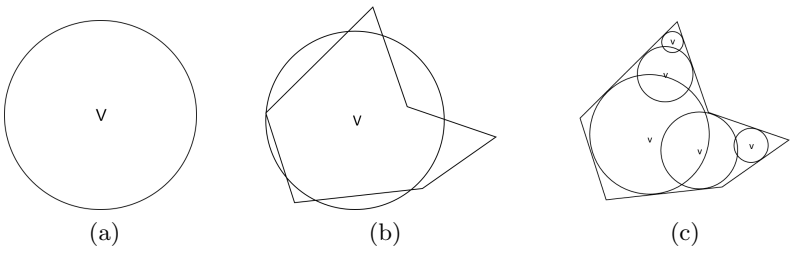


Fig. 1. Location verification for (a) circular region $\mathcal{R}$that is perfectly covered by a single verifier placed at the center of $\mathcal{R}$, (b) arbitrary shaped region $\mathcal{R}$, a single verifier does not give perfect coverage, and (c) arbitrary shaped region $\mathcal{R}$, multiple verifiers are placed inside $\mathcal{R}$ ([16]'s approach), also does not give perfect coverage.

respect to the policy region $\mathcal{R}$. Our security definition formalizes attacks that involve a malicious prover outside $\mathcal{R}$, an unregistered user inside $\mathcal{R}$, and a collusion of a malicious prover and a helper who is inside $\mathcal{R}$. A significant challenge in modelling and achieving security is the possibility of the prover moving between their interactions with different verifiers. Our security model uses ITMs (Interactive Turing Machines) to model the prover and verifiers and does not formalize time (movement of the prover). We assume prover movement will be detected through other mechanisms, and our protocol introduces a mechanism that does that, allowing us to use our security model.

Construction. Armed with this model and definition, we propose a systematic approach to designing inRA protocols for the proof of the conjunction (1) and with *quantifiable correctness error*, and give an efficient algorithm to minimizing the error (see below). The approach in its basic form uses two verifiers V_0 and V_1, and covers the region $\mathcal{R}$ with a *pseudo-rectangle (P-rect)* $R'(V_0, V_1)$ that is formed by two rings centered at the two verifiers (See Figure 2 and Section 3). A ring is formed by a verifier running a DUB protocol followed by a *Distance Lower Bounding (DLB)* protocol [27] (that guarantees a lower bound on the distance of the prover to the verifier- See Section 2), with the prover. The two verifiers work in tandem, with the second immediately following the first. Verifiers use omnidirectional antennas during the protocol initialization, and use *directional antennas* for the challenge-response phase.

This basic inRA protocol approximates $\mathcal{R}$ with a P-rect and results in FA and FR. We define the total error as the sum of FA and FR errors and aim at minimizing it. Our approach however can be easily extended to the case that the two types of errors have different significance – See Sec 6.

Minimizing error. For fixed locations of V_0 and V_1, the total error is a function of the distance bounds of the two verifiers. To minimize error one can use brute force method and for every possible values of distance bounds, find the error and select the minimal value. This is an infeasible task in practice. We give an innovative approach that uses *maximum subarray problem* [12] algorithm to solve the optimization problem of finding a P-rect that is proved to minimize the total

error in approximate coverage of $\mathcal{R}$ with a P-rect. The algorithm has complexity $O(n^3)$ where n is the size of $\mathcal{R}$ represented as a point set. This basic algorithm can be employed multiple times using more verifiers, to increase accuracy. In Section 6 we show that using two P-rects to cover the region reduces the total error by up to 15%. We leave the problem of optimizing the number and the locations of the verifiers as an interesting direction for future work.

Security proof. In our basic protocol (Section 4) we will use a novel approach to detecting movement of the prover during protocol execution, by using each verifier to play the role of an observer for the other verifier's interaction with the prover. We will then use our security model to prove security against attacks. We discuss how protection against a new attack called *key splitting attack* that is the result of using a pair of DUB and DLB protocols with two verifiers, can be avoided by using keys shared with V_0 and V_1 both, to generate the fast phase responses to each verifier.

Implementation and experimental results. We implemented the optimization algorithm for two verifiers and applied it to four policy regions corresponding to buildings in our University (Section 6). We started with a 640×640 Google Map image of the policy region, and converted it into a binary image for point-set representation of the policy regions. To achieve higher accuracy, we used two P-rects to cover the policy region. Table 1 summarizes our results. The highest accuracy is obtained for the most regularly shaped rectangular region. In all cases FA and FR range between 0.81% to 5.16%, and 3.89% to 5.58%, respectively.

We compared our approach with the scheme in Sastry et al. [16]. This is the only system with comparable security goals and assumptions. Comparison (Sec. 6) clearly shows superior performance of our approach: [16] uses 5 verifiers to achieve 93% accuracy, and uses informal security analysis, while we use 2 verifiers, achieve 96.4% accuracy, and provide formal security proof.

Extensions. One can define weights for each type of FA and FR error depending on the application, and use optimization approach on the weighted error function. The approach raises numerous interesting open questions such as optimizing the total error when there are more than two verifiers, and one needs to select their locations and distance bounds. We leave these for future work.

Organization. Section 2 is preliminaries. Section 3 describes our inRA model. Section 4 details the inRA protocol Π_{rect}, and the security analysis of Π_{rect}. Section 5 provides our approach to minimize error. Section 6 includes our experimental results. Section 7 presents related works and Section 8 concludes the paper. Appendix includes security models of DUB and DLB protocols, and explains *erasure sequence* that is used in DLB, and our proofs. Appendix includes security models of DUB and DLB protocols, and explains *erasure sequence* that is used in DLB, and our proofs.

2 Preliminaries

Distance Bounding. Secure *distance bounding protocols* have three phases: i) initialization phase, ii) Challenge-response phase, and iii) Verification phase. The

round-trip time of a challenge and response is used to estimate distance. The goal of a distance Upper bounding (DUB) protocol is to ensure that a prover P located at distance d_{PV} satisfies $d_{PV} \leq \mathcal{B}_U$ where $\mathcal{B}_U$ is a fixed upperbound.

The main attacks on distance bounding protocols are, i) Distance fraud attack: a far away dishonest prover tries to claim a shorter d_{PV} and be accepted by V; ii) Mafia fraud attack: an external attacker uses the communication of an honest prover to get accepted by the verifier, and iii) Terrorist attack (also known as collusion attack): a dishonest prover gets help from a helper that is close to the verifier, to get accepted by the verifier. A number of formal security models that capture above attacks, and protocols with provable security have been proposed [21, 9]. Secure DUB protocols are vulnerable to distance enlargement attack but not to distance reduction attack [5].

The goal of distance lower bounding (DLB) protocols [27] is the converse: a prover wants to prove that their distance to the verifier is larger than a given bound. Zheng *et. al.* [27] showed that one cannot simply use DUB protocols to guarantee a lower bound on the distance of the prover. They proposed a security model for DLB that is inline with the DUB security model, and constructed a DLB protocol with provable security in their model. Our construction of inRA protocol Π_{Prect} uses DLB protocol together with a DUB protocol.

Maximum Subarray Problem. Optimizing P-rect uses *maximum subarray problem (MSP)*, first proposed in [12]. The problem is to select a contiguous segment of an array that has the largest sum over all possible array segments. Efficient algorithms for MSP problem have applications in computer vision, data mining and genomic sequence analysis [24, 20, 10]. For a 2D array $a[1..m][1..n]$, the maximum sub-array M is given by [2],

$$M = \max \sum_{x=i,y=g}^{j,h} a[x][y] | 1 \leq i \leq j \leq m, 1 \leq g \leq h \leq n \tag{2}$$

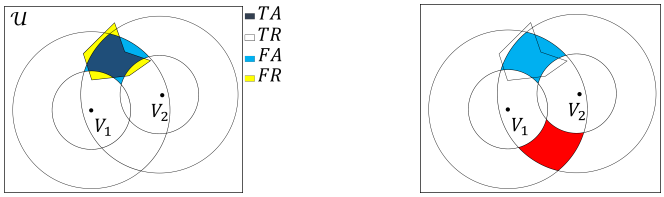
Solutions have complexity cubic or sub-cubic [19]. To find the P-rect with the lowest total error, or equivalently maximum accuracy, we will use *FindOptimalBounds* algorithm (Sec. 4) that uses the extended Kadane's algorithm [2].

3 In-Region Authentication Systems

Consider a two-dimensional planar connected (path connected) geographic area represented by an array of points, each point representing a geolocation[1]. Let $\mathcal{U}$ denote the universe of all points of interest, and $\mathcal{R} \subset \mathcal{U}$, be the *policy region.* There are multiple parties, each represented by a polynomially bounded Interactive Turing Machine (ITM), and associated with a location *loc*.

A *protocol instance* between two honest parties P and V is modelled by a probabilistic *experiment* where each party uses its algorithm on its input

[1] A point set corresponding to a geographic area can be constructed using a bitmap image of the area, at the required resolution level. Thus each point corresponds to a geographic square of size u where u is determined by the resolution of the mapping.



(a) (b)

Fig. 2. (a) The policy region $\mathcal{R}$ is the yellow arbitrary shaped region. The blue (almost) rectangular area is P-rect $\mathcal{R}_{\Pi}$ for the inRA protocol in Section 4. The dark blue area of $\mathcal{R}$ is correctly covered. The remaining yellow and blue areas are $FR_{\Pi,\mathcal{R}}$ and $FA_{\Pi,\mathcal{R}}$, respectively. (b) The upper intersection forms R_{rect} to cover $\mathcal{R}$(blue). The lower intersection forms R'_{rect} (red), an ambiguous region.

and random coin. This is shown by $P(x; r_P) \leftrightarrow V(y; r_V)$, x and y are the inputs, r_P and r_V are the random coins of the two participants, respectively. We can "enlarge" the experiment to include an adversary's algorithm, shown as: $P(x; r_P) \leftrightarrow A(r_A) \leftrightarrow V(y; r_V)$. This means that an adversary A is interfering with the communication between the honest participants.

inRA protocols. Let $\mathcal{R}$ be a connected policy region (Fig. 2). The *verifying system* consists of a set of verifiers $\mathcal{V} = \{V_0 \cdots V_{m-1}\}$, with publicly known locations. Verifiers are trusted to follow the protocol and can communicate among themselves through secure channels to exchange information and coordinate their actions. Verifiers are equipped with *directional antennas* whose signals can be received in a conic region of spece that covers $\mathcal{R}$. A prover P with location loc_P, has shared keys with the verifier set $\mathcal{V}$. The prover is not trusted.

An *in-region authentication protocol* is a protocol Π between P and $\mathcal{V}$, at the end of which $\mathcal{V}$ outputs $Out_{\mathcal{V}} = 0$ or 1, denoting reject and accept of the prover's claim, respectively. Prover does not have an output and so $Out_{\mathcal{V}}$ is the protocol output. The prover's claim is stated as the conjunction in (1).

DUB protocols can be seen as inRA protocols where the second proposition is, *P is within a distance bound from the verifier.*

Error and accuracy in inRA protocols. Consider an instance of a protocol Π between an honest prover P and the verifier set, in the absence of an adversary. Let $\mathcal{R}_{\Pi} \subset \mathcal{U}$ denote the set of points $u \in \mathcal{U}$ that Π will have $Out_{\mathcal{V}} = 1$. We define two types of errors for the protocol Π with respect to the region $\mathcal{R}$: $FA_{\Pi,\mathcal{R}}$ and $FR_{\Pi,\mathcal{R}}$, denoting false acceptance and false rejection of the protocol Π, respectively, where, (i) $FA_{\Pi,\mathcal{R}}$ is the *set of locations* that are in $R_{\Pi} \setminus \mathcal{R}$,[2] and $FR_{\Pi,\mathcal{R}}$ is the *set of locations* that are in $\mathcal{R} \setminus \mathcal{R}_{\Pi}$. *Accuracy ratio* can be defined as follows [14]:

$$Accuracy\ \ ratio = \frac{TA_{\Pi,\mathcal{R}} + TR_{\Pi,\mathcal{R}}}{TA_{\Pi,\mathcal{R}} + TR_{\Pi,\mathcal{R}} + FA_{\Pi,\mathcal{R}} + FR_{\Pi,\mathcal{R}}} \tag{3}$$

[2] $A \setminus B$ denotes the set of points that are in A and not in B.

where $TA_{\Pi,\mathcal{R}}$ and $TR_{\Pi,\mathcal{R}}$ denote the true acceptance and true rejection sets, $TA_{\Pi,\mathcal{R}}$ is the set of points in $\mathcal{R} \cap \mathcal{R}_\Pi$ and are accepted by the algorithm, and $TR_{\Pi,\mathcal{R}}$ is the set of points in $\mathcal{U} \setminus \{\mathcal{R} \cup \mathcal{R}_\Pi\}$ and are rejected by the algorithm. Now, *Error ratio* $= 1 -$ *Accuracy ratio*, and can be expressed as,

$$Error\ \ ratio = \frac{FA_{\Pi,\mathcal{R}} + FR_{\Pi,\mathcal{R}}}{TA_{\Pi,\mathcal{R}} + TR_{\Pi,\mathcal{R}} + FA_{\Pi,\mathcal{R}} + FR_{\Pi,\mathcal{R}}} \tag{4}$$

Since $\mathcal{U}$= $(TA_{\Pi,\mathcal{R}}+\ TR_{\Pi,\mathcal{R}}+\ FA_{\Pi,\mathcal{R}}+\ FR_{\Pi,\mathcal{R}})$ is constant, to minimize error one needs to minimize $(FA_{\Pi,\mathcal{R}}+\ FR_{\Pi,\mathcal{R}})$. In our work we use *error* $E_{\Pi,\mathcal{R}}$ given by,

$$\text{Error:} \quad E_{\Pi,\mathcal{R}} = FA_{\Pi,\mathcal{R}} + FR_{\Pi,\mathcal{R}} \tag{5}$$

Note that one can attach weights to points in $FA_{\Pi,\mathcal{R}}$ or $FR_{\Pi,\mathcal{R}}$ to reflect their importance in a particular application. In this paper we assume the same significance for the two types of errors. For $\mathcal{R} = TA_{\Pi,\mathcal{R}}+\ FR_{\Pi,\mathcal{R}}$, we can write,

$$\begin{aligned} FA_{\Pi,\mathcal{R}}+\ FR_{\Pi,\mathcal{R}} &= FA_{\Pi,\mathcal{R}} + (\mathcal{R} - TA_{\Pi,\mathcal{R}}) \\ &= \mathcal{R} - (TA_{\Pi,\mathcal{R}} - FA_{\Pi,\mathcal{R}}). \end{aligned}$$

$\mathcal{R}$ is fixed and so *minimizing* $(FA_{\Pi,\mathcal{R}}+\ FR_{\Pi,\mathcal{R}})$ *is equivalent to maximizing* $(TA_{\Pi,\mathcal{R}} - FA_{\Pi,\mathcal{R}})$. We say that in our $\mathcal{R}$ coverage problem, error is minimized by minimizing $(FA_{\Pi,\mathcal{R}}+\ FR_{\Pi,\mathcal{R}})$, or equivalently, accuracy is maximized by maximizing $(TA_{\Pi,\mathcal{R}} - FA_{\Pi,\mathcal{R}})$. Therefore, we define Accuracy $A_{\Pi,\mathcal{R}}$ as:

$$\text{Accuracy:} \quad A_{\Pi,\mathcal{R}} = TA_{\Pi,\mathcal{R}} - FA_{\Pi,\mathcal{R}} \tag{6}$$

Definition 1 (in-Region Authentication). *An in-region authentication (inRA) protocol* Π *is a tuple* $\Pi = (Gen, P, \mathcal{V} = \{V_0 \cdots V_{m-1}\}, \mathcal{R})$ *where:*

1. $\mathcal{X} \leftarrow Gen(1^s, r_k)$ *is a randomized key generation algorithm that generates a vector* $\mathcal{X} = \{x_0, ..., x_{m-1}\}$ *of* n *secret keys, where* x_i *is the prover's shared secret key with* V_i*, and* r_k *denoting the random coins of* Gen*.* s *is the security parameter.*
2. $P(\mathcal{X}; r_P)$*, is a ppt.(probabilistic polynomial time) ITM (Interactive Turing Machine) running the prover algorithm with random input* r_P *and the secret key vector* $\mathcal{X} = \{x_0, ..., x_m - 1\}$*.*
3. $\mathcal{V} = (V_0, ..., V_{m-1})$ *is a set of verifiers, each verifier* $V_i(x_i; r_{V_i}) \in \mathcal{V}$ *is a ppt. ITM running algorithm with random input* r_{V_i} *and shared secret* x_i*. We write* $\mathcal{V}(\mathcal{X}, r_\mathcal{V})$ *to denote the set of the verifiers' algorithms.*
4. $\mathcal{R}$ *is a set of points corresponding to a contiguous region. This is the* policy region*.*

The protocol satisfies the following properties:

- ***Termination****:* $(\forall s)$ $(\forall \mathcal{Z})$ $(\forall (r_k, r_\mathcal{V}))$ $(\forall loc_\mathcal{V})$ *if* $\mathcal{X} \leftarrow Gen(1^s, r_k)$ *and* $(\mathcal{Z} \longleftrightarrow \mathcal{V}(\mathcal{X}; r_\mathcal{V}))$ *is the execution where* $\mathcal{Z}$ *is any set of prover algorithms, then* $\mathcal{V}$ *halts in polynomial number of computational steps* $(Poly(s))$*;*

– ***p-Completeness:*** $(\forall s)$ $(\forall(loc_{\mathcal{V}}, loc_P))$ *such that* $loc_P \in \mathcal{R}$ *we have*

$$\underset{r_k, r_P, r_{\mathcal{V}}}{Pr}\left[Out_{\mathcal{V}} = 1 : \begin{array}{l}\mathcal{X} \leftarrow Gen(1^s, r_k)\\ P(\mathcal{X}; r_P) \leftrightarrow \mathcal{V}(\mathcal{X}; r_{\mathcal{V}})\end{array}\right] \geq p. \tag{7}$$

Similar definition of termination and completeness is used for DB prortocols [21, 27, 1].

3.1 inRA Security

We consider a prover, possibly malicious, who may receive help from a helper who is in $\mathcal{R}$ but does not have a secret key.

The adversary attempts to prove that their location is inside $\mathcal{R}$ (while they are actually outside) and their success chance must be negligible even if they know the shared key. We use a game-based approach in defining security, and define security in terms of the success chance of an adversary in the following security games against a challenger. Each game starts with a setup phase where the challenger sets the keys and locations of participants. This is followed by the adversary corrupting some of the participants (depending on the game), engaging them in a learning phase and finally the attack phase. We omit the details because of space and outline the steps of each game in the definition of each attack. In the following, a dishonest prover is denoted by P^*.

in-Region Fraud (inF). In this attack, a corrupted prover P^* who has the secret key and is in $\mathcal{U} \setminus \mathcal{R}$ wants to prove that they are inside $\mathcal{R}$.

Definition 2 (inF-resistance). *An inRA protocol Π is α-resistant to in-region fraud if* $(\forall s)(\forall P^*)(\forall loc_{\mathcal{V}})$ *such that* $loc_P \notin \{\mathcal{R} \cup FA_{\Pi,\mathcal{R}}\}$*, and* $(\forall r_k)$ *we have,*

$$\underset{r_{\mathcal{V}}}{Pr}\left[Out_{\mathcal{V}} = 1 : \begin{array}{l}\mathcal{X} \leftarrow Gen(1^s, r_k)\\ P^*(\mathcal{X}) \leftrightarrow \mathcal{V}(\mathcal{X}; r_{\mathcal{V}})\end{array}\right] \leq \alpha. \tag{8}$$

The above definition also captures a special type of attack - *in-region hijacking* (follows from a similar type of attack in DB protocols - *distance hijacking*). A dishonest prover P^* located outside $\mathcal{R}$ uses the inRA communications of unaware honest provers (inside $\mathcal{R}$) to get authenticated as an honest prover.

in-Region Man-in-the-Middle (inMiM). A corrupted participant who does not have a key but is inside $\mathcal{R}$, interacts with multiple provers P's and the verifier set $\mathcal{V}$, and uses transcripts of these protocols to succeed in the inRA protocol.

Definition 3 (inMiM-resistance.). *An inRA protocol Π is β-resistant to inMiM attack if,* $(\forall s)(\forall m, l, z)$ *that are polynomially bounded,* $(\forall \mathcal{A}_1, \mathcal{A}_2)$ *that are polynomially bounded, for all locations s.t.* $loc_{P_j} \notin \{\mathcal{R} \cup FA_{\Pi,\mathcal{R}}\}$*, where* $j \in \{q+1, ..., t\}$*, we have*

$$Pr\left[Out_{\mathcal{V}} = 1 : \begin{array}{c}\mathcal{X} \longleftarrow Gen(1^s, r_k)\\ P_1(\mathcal{X}), ..., P_q(\mathcal{X}) \longleftrightarrow \mathcal{A}_1 \longleftrightarrow \mathcal{V}_1(\mathcal{X}), ..., \mathcal{V}_z(\mathcal{X})\\ P_{q+1}(\mathcal{X}), ..., P_t(\mathcal{X}) \longleftrightarrow \mathcal{A}_2(View_{\mathcal{A}_1}) \longleftrightarrow \mathcal{V}(\mathcal{X})\end{array}\right] \leq \beta. \tag{9}$$

The attacker is a pair of algorithms $(\mathcal{A}_1, \mathcal{A}_2)$, where $\mathcal{A}_1$ denotes the learning phase during which the attacker interacts with the protocol-runs of q provers that can be anywhere, and provides this view to $\mathcal{A}_2$ in the second stage of the attack. Definition 3 is general and captures other attack settings that are traditionally referred to as mafia fraud and impersonation attack, in DB protocols. Mafia fraud is an MiM attack as defined above but without a learning phase. In impersonation attack the attacker uses multiple possibly concurrent interactions with the verifiers to make the verifier output 1.

in-Region Collusion Fraud (inCF). Arguably the strongest attack and involves the collusion of a corrupted prover who is in $\mathcal{U} \setminus \mathcal{R}$, and a helper who is inside $\mathcal{R}$. In collusion fraud the assumption is that the corrupted prover does not want their long-term secret key to be learnt by the helper as otherwise the helper would have a better chance to succeed in other attacks individually. The prover however attempts to use the helper's location to succeed in the attack. In the following definition of rCF-resistance, success of the attacker in inCF implies that - the attacker in a MiM attacker as defined above (and realized by the helper), will also succeed. $P^{(*)}(\mathcal{X})$ denotes honest or dishonest prover.

Definition 4 (inCF-resistance.). *An inRA protocol Π is (γ, η)-resistant to collusion fraud if* $(\forall s)$ $(\forall P^*)$ $(\forall loc_{\mathcal{V}_0}$ *s.t.* $loc_{P^*} \notin \{\mathcal{R} \cup FA_{\Pi,\mathcal{R}}\}$ $(\forall \mathcal{A}^{CF} ppt.)$ *s.t.*

$$Pr\left[Out_{\mathcal{V}_0} = 1 : \begin{array}{l} \mathcal{X} \longleftarrow Gen(1^s) \\ P^{(*)}(x) \longleftrightarrow \mathcal{A}^{CF} \longleftrightarrow \mathcal{V}_0(\mathcal{X}) \end{array}\right] \geq \gamma, \tag{10}$$

over all random coins, there is a two stage attacker $(\mathcal{A}_1, \mathcal{A}_2)$ as defined in MiM with the additional relaxation that in the learning phase, the attacker can interact with the malicious prover also, such that,

$$Pr\left[Out_{\mathcal{V}} = 1 : \begin{array}{l} \mathcal{X} \leftarrow Gen(1^s) \\ P_1^{(*)}(\mathcal{X}), ..., P_q^{(*)}(\mathcal{X}) \longleftrightarrow \mathcal{A}_1 \longleftrightarrow \mathcal{V}_1(\mathcal{X}), ..., \mathcal{V}_z(\mathcal{X}) \\ P_{q+1}(\mathcal{X}), ..., P_r(\mathcal{X}) \longleftrightarrow \mathcal{A}_2(View_{\mathcal{A}_1}) \longleftrightarrow \mathcal{V}(\mathcal{X}) \end{array}\right] \geq \eta. \tag{11}$$

The above definition of inCF captures a widely used attack model for DB protocols, which we call *in-Region Terrorist fraud (inTF)* in which P^*, with $loc_{P^*} \notin \{\mathcal{R} \cup FA_{\Pi,\mathcal{R}}\}$, uses a helper who does not have the secret key, to succeed in an instance of the protocol.

We do not consider jamming attacks blocking all communication. A *secure inRA protocol* provides security against inF, inMiM and inCF.

4 Pseudo-rectangle (P-rect) Cover Approach to inRA

We assume the setting of Section 3 and describe our approach using *basic inRA protocol* that uses two verifiers V_0, V_1 with (publicly known) location loc_{V_0} and loc_{V_1}. The prover P shares the secret keys x_0 and x_1 with V_0, V_1, respectively.

4.1 Basic (two-verifier) P-rect Approach

Protocol communication. We assume radio signal travel at the speed of light and the round trip time of a challenge and response can provide a reliable estimate of distance. There are two collaborating verifiers who interact with the prover using, *slow communication* that is used for time-insensitive messages over reliable channels, and *fast communication* that are time sensitive messages that are used for estimating distance and are sent over the physical channel that is noisy. For simplicity, we do not consider noise. Our results however can be easily extended to noisy channels by modifying the protocol parameters (thresholds). Verifiers are equipped with omnidirectional and directional antennas, although in each run of the protocol we require only one of them to use their directional antenna for communication with the prover. Communication between the verifiers takes place over a secure and reliable channel and is not time sensitive.

P-rectangle. For a fixed pair of verifiers, V_0 and V_1, with lower and upper bound pairs, $\{\ell_{V_0}, u_{V_0}\}, \{\ell_{V_1}, u_{V_1}\}$, respectively, a P-rect is defined as the set of points $x \in \mathcal{U}$ that satisfy the following inequalities:

$$d(x, loc_{V_0}) \leq u_{V_0},\ d(x, loc_{V_0}) \geq \ell_{V_0},\ d(x, loc_{V_1}) \leq u_{V_1},\ d(x, loc_{V_1}) \geq \ell_{V_1}$$

where $d(.,.)$ is the Euclidean distance. Consider the two pairs of concentric circles, centred at loc_{V_0} with radii $\{\ell_{V_0}, u_{V_0}\}$ and at loc_{V_1} with radii $\{\ell_{V_1}, u_{V_1}\}$, respectively. The intersection of the four circles defines two P-rects (Fig. 2b).

We denote the two mirrored rectangles by $R_{rect}(loc_{V_0}, loc_{V_1}, \ell_{V_0}, u_{V_0}, \ell_{V_1}, u_{V_1})$ and $R'_{rect}(loc_{V_0}, loc_{V_1}, \ell_{V_0}, u_{V_0}, \ell_{V_1}, u_{V_1})$. We use R_{rect} and R'_{rect} when parameters are known from the context. These P-rects are formed when V_0 and V_1 each executes a pair of DUB and DLB protocols with corresponding upper and lower bounds. To distinguish between the two, one of the verifiers can use a directional challenge towards the target region $\mathcal{R}$. The inRA protocol Π_{rect} below uses a P-rect to cover $\mathcal{R}$. We quantify the error and prove security of this protocol.

Protocol Π_{rect}. For given values of $loc_{V_0}, loc_{V_1}, \ell_{V_0}, u_{V_0}, \ell_{V_1}, u_{V_1}$, the protocol bounds the prover within- $R_{rect}(loc_{V_0}, loc_{V_1}, \ell_{V_0}, u_{V_0}, \ell_{V_1}, u_{V_1})$ (See Fig 3).

Initialization phase. Prover P and verifiers V_0, V_1 have shared secret $x_i, i = 0, 1$ and security parameter k at the start of the protocol. Prover picks four independently generated nonces $N^l_{p_i}, N^u_{p_i}$, $i = \{0, 1\}$, each of length k, and sends a pair of nonces to each verifier $V_i, i = \{0, 1\}$. Each verifier V_i picks two independently generated nonces of the same length, $N^l_{v_i}, N^u_{v_i}$, and two random strings A^u_i, A^l_i, each of length $2n$ ($2n$ corresponds to number of rounds in fast-exchange phase) and calculates, $M^u_i = A^u_i \oplus f_x(N^l_{p_i}, N^l_{v_i})$ and $M^l_i = A^l_i \oplus f_x(N^u_{p_i}, N^u_{v_i})$. f is a Pseudo Random Function (PRF). $N^u_{p_i}, N^u_{v_i}, M^u_i, M^l_i$ are sent to the prover who decrypts and stores A^u_i, A^l_i. These are the response tables of the distance upper and lower bound challenges for the respective verifiers, in the fast-exchange phase. All communications between the prover and the verifiers use omnidirectional antenna in the initialization phase.

Fast-exchange (FE) phase. WLOG assume V_0 starts the FE phase and notifies V_1 to start its FE phase right after sending its last challenge [3].

[3] We assume verifiers have agreed on the order.

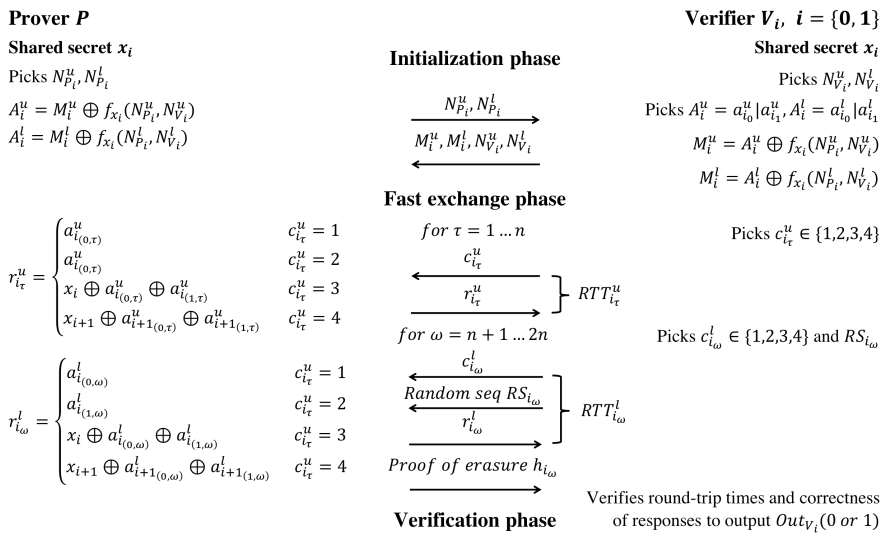


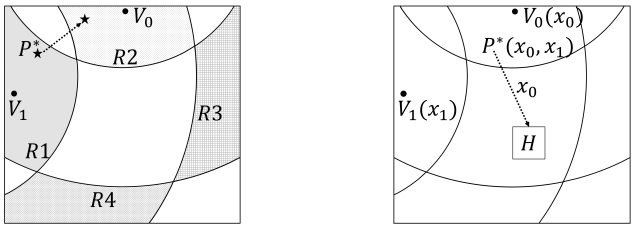
Fig. 3. inRA protocol Π_{rect} between a prover and 2 verifiers. In initialization phase prover and verifiers generate and exchange nonces $N^l_{v_i}, N^u_{v_i}$. Fast Exchange phase is $2n$ rounds of challenge $(c^u_{i_\tau}, c^l_{i_\omega})$ and responses $(r^u_{i_\tau}, r^l_{i_\omega})$ for DUB and DLB. The responses are calculated using a pseudo-random function with special properties. In verification phase, verifiers check round-trip time and correctness of responses.

V_0 will use an omnidirectional antenna to send its challenges, while V_1 will use a directional antenna with the direction and the angle of the beam chosen to cover only one of the two mirrored P-rects R_{rect} and R'_{rect} (See Fig. 2b). This means that only the points in R_{rect} will receive the challenge from V_1.

The FE phase of each verifier consists of $2n$ consecutive rounds of challenge-response ($n \in \Omega(k)$), where the first n rounds are used for distance upper bounding, and the last n rounds for distance lower bounding. In each distance upper bounding round $\tau, \tau = \{1, ..., n\}$, verifier V_i picks a challenge value $c^u_{i_\tau} \in \{1, 2, 3, 4\}$, and sends it to the prover, who must respond immediately with $r^u_{i_\tau}$, as shown in figure 3.

Note that the prover's response, when the challenge value is in the set $\{1, 2\}$, depends on the nonces of the verifier that has sent the challenge, but when the challenge value is in the set $\{3, 4\}$, their response value depends on both verifiers' nonces. This is to prevent *key-splitting attack* in which a malicious prover who is located in specific parts of the plane (outside $\mathcal{R}$), can combine parts of the secret keys of the two verifiers to succeed in their attack (more in Section 4.2). Verifiers will verify the responses at the end of the protocol and after sharing their nonces. To estimate the distance, each verifier measures the round-trip-time $RTT^u_{i_\tau}$, from sending $c^u_{i_\tau}$ to receiving $r^u_{i_\tau}$, of a round.

Rounds ω, $\omega = \{n+1, ..., 2n\}$, are for DLB protocol. In each such round the verifier V_i picks a random challenge $c^l_{i_\omega} \in \{1, 2, 3, 4\}$, together with an erasure



(a) Prover movement (b) Key splitting

Fig. 4. (a) Prover movement attack: Prover responds to DUB and DLB challenges of verifeir V_0 while in *Region 1* ($R1$), and DUB and DLB challenges of verifeir V_1 while in *Region 2* ($R2$). (b) Key splitting attack: Prover responds to DUB and DLB challenges of V_1, and also DUB challenges of V_0 while asking the helper to respond to the DLB challenges of V_0.

sequence RS_{i_ω} of length z_{i_ω} [4], that is used to prevent prover from delaying the response and claiming a farther distance. Prover has to send a response as shown in Fig. 3, as well as the proof of receiving the erasure sequence. Verifier also measures and stores the round-trip-time $RTT^l_{i_\omega}$, from sending $c^l_{i_\omega}$ to receiving $r^l_{i_\omega}$, in each round.

Verification phase. Firstly, verifiers check correctness of the responses $(r^u_{i_\tau}, r^l_{i_\omega})$, as well as the proof of erasures, h_{i_ω}. Then each verifier checks if the round-trip-time of the FE challenge-responses in each of the first n rounds satisfies: $\frac{RTT^u_{i_\tau}}{2} \leq u_{V_i}$, and each of the last n rounds satisfies $\frac{RTT^l_{i_\omega}}{2} \geq \ell_{V_i} + T(z_{i_\omega} - 1)$. $T(z_{i_\omega} - 1)$ is the maximum processing time required by the prover to store the erasure and compute the proof of erasure.If the above checks succeed, then verifier V_i outputs $Out_{V_i} = 1$. If both verifiers output 1, then P is accepted, otherwise P is rejected.

4.2 Security Analysis

Π_{rect} uses a pair of DUB and DLB protocols with two verifiers. To prove security of the protocol we first eliminate attacks that are because of the ability of the prover to change its location between its interaction with the two verifiers, or leaking part of its key to the helper such that it succeeds in lying about its location without enabling the helper to succeed in its individual attack.

Prover movement. Location verification protocols that consider prover's communication with multiple verifiers are vulnerable to attacks that involve movement of the prover. Fig. 4a shows such a scenario. A malicious prover located outside the P-rect attempts to get accepted by moving from one place to another.

[4] An erasure sequence is a pseudo-random sequence of defined length that is used in secure DLB protocols to prevent the prover from storing malicious codes in device memory to delay their responses. We follow the construction of erasure sequence of [27], which is also explained in the Appendix A.

Consider two regions: *Region 1* ($R1$) contains all the points that are within the ring centered at V_0 and inside the lower bound of V_1, and *Region 2* ($R2$) contains all the points that are within the ring centered at V_1 and inside the lower bound of V_0. Now the prover changes its location, and can succeed by responding to DUB and DLB challenges of verifier V_0 while in *Region 1*, and DUB and DLB challenges of verifier V_1 while in *Region 2*. Similar attack can take place by the prover moving between *Region 2,3*, or *Region 3,4*, or *Region 4, 1*.

Chiang *et al.* proposed a solution to prover movement [7] that uses simultaneous challenge from the verifiers. However, this requires the prover to claim a location first and this needs GPS signal (or other location determination infrastructure) and so not directly applicable to indoor area. We propose a novel approach to detecting the prover movement in which each verifier acts as an observer for the other verifier. More details below.

Let V_0 be an observer who passively records the timing of the signals for the communication between the prover and verifier V_1, and V_1 play a similar role for V_0. Let us revisit the prover movement scenario in Fig. 4a. First, we consider the prover movement between *Region 1,2*. In this case, we only consider the communication in the fast exchange phase of the DLB protocols. Notice that P^* must be in *Region 1* ($R1$) while responding to the DLB challenge from V_0, and in *Region 2* ($R2$) while responding to the DLB challenge from V_1. Consider the following time-stamps (all challenges are DLB challenges): t_0: V_0 sends challenge to P^* in *Region 1*; t_1: V_1 sends challenge to P^* in *Region 2*; T_0: V_0 receives response from P^* sent from *Region 1*; T_1: V_1 receives response from P^* sent from *Region 2*; T_0': V_0 listens to the response of P^* sent from *Region 2*; T_1': V_1 listens to the response of P^* sent from *Region 1*;

We assume the prover's processing time is known and is public. V_0, from DLB communication, will compute the distance between itself and P^* using their challenge and response round trip time as: $d_{V_0P^*} = \frac{(T_0 - t_0) \times C}{2}$, where C is the speed of radio wave. Similarly, V_1 will compute its distance to P^* as: $d_{V_1P^*} = \frac{(T_1 - t_1) \times C}{2}$. By listening to the other DLB communication, V_0 will compute the distance between itself and P^* based on the response times of P^* as: $d'_{V_0P^*} = \left(T_0' - \frac{T_1 - t_1}{2}\right) \times C$. This is because the response from P^* at *Region 2* leaves P^* at time $(T_1 - t_1)/2$, and reaches V_0 at time T_0'. Similarly, V_1 will compute the distance between itself and P^* at *Region 1*, using its listening time of the response of P^*, as: $d'_{V_1P^*} = \left(T_1' - \frac{T_0 - t_0}{2}\right) \times C$. This is because the response from P^* at *Region 1* leaves P^* at time $(T_0 - t_0)/2$, and reaches V_1 at time T_1'. The system detects movement of the prover if any of the following checks do not hold:

$$d_{V_0P^*} = d'_{V_0P^*} \ , \ d_{V_1P^*} = d'_{V_1P^*}. \tag{12}$$

The protocol immediately rejects and aborts when multiple provers are detected.

A similar approach for each type of communication, e.g., DLB or DUB, can detect the prover movement between *Region 2,3*, or *Region 3,4*, or *Region 4, 1*.

Key splitting attack. This attack is a result of using a pair of DUB and DLB protocols with two verifiers. In a key splitting attack, the prover leaks part of their key information to a helper to allow them to succeed in its attack,

without allowing the helper to have a better chance to succeed on its own. Figure 4b shows a scenario for such an attack. Here, a malicious prover P^* is located within the ring centered at V_1 and inside the lower bound of verifier V_0. P^* shares key x_0, x_1 with V_0, V_1 respectively. A helper H is located inside the P-rect. P^* gives x_0 to H. Now the prover will succeed by correctly responding to DUB and DLB challenges of V_1, and also DUB challenges of V_0 while asking the helper to respond to the DLB challenges of V_0. Note that the attack is successful because this key leakage will not directly result in a successful inMiM (H requires both keys (x_0, x_1) to succeed in inMiM) and so according to inCF Definition (Def. 4), the protocol is not secure.

We thwart this attack by including both keys (x_0, x_1) in generating the response to the challenges of each verifier. As shown in Fig. 3, upon receiving a challenge $c^u_{i_\tau} = 3$ from verifier V_i $(i = \{0, 1\})$, generating the response $r^u_{i_\tau}$ requires key and response table shared with verifier V_i . If $c^u_{i_\tau} = 4$, it requires key and response table shared with verifier V_{i+1}.

Revisiting the above key splitting scenario, to get accepted in Π_{rect}, P^* must share both keys x_0, x_1 with the helper, otherwise helper would not be able to generate the responses to the DLB challenges $c^l_i = 4$ from V_0. This will lead to a successful inMiM by H - which guarantees security (Def. 4) of our protocol.

Security against inF, inMiM and inCF. By removing the threats described above, we are ready to analyze the security of Π_{rect} against the three attacks defined in Section 3.1: inF, inMiM and inCF.

Let, Π^{DUB}_{rect} and Π^{DLB}_{rect} denote DUB and DLB protocols used in Π_{rect}. The detailed inRA protocol is presented in Fig 3. We use the constructions of [21] and [27] for DUB and DLB protocols, respectively. These protocols are provably secure against the main three attacks (distance fraud, man-in-the-middle and collusion fraud) of distance bounding protocols that have been defined consistent with the corresponding attacks of inRA in Sec. 3. Security models for these attacks in Π^{DUB}_{rect} and Π^{DLB}_{rect} are given in appendix B. Security of these component protocols does not directly lead to the security of inRA with respect to the P-rect formed by these protocols,i.e., we need to consider attack scenarios that yield from a single verifier running two different protocols (DUB and DLB).

For each verifier $V_i \in \mathcal{V}$, the response table a_u of the DUB protocol Π^{DUB}_{rect} and a_l of the DLB protocol Π^{DLB}_{rect} are independently generated from each other and for each verifier. This holds because verifiers are honest and a response tables is constructed using the randomness of the prover and corresponding verifier.

Theorem 1. *For a region $\mathcal{R}$, the protocol Π_{rect} satisfies the following:*
1: If Π^{DUB}_{rect} and Π^{DLB}_{rect} are secure against distance fraud attack with probability α_u, α_ℓ in Def. 5 and Def. 8, respectively, then Π_{rect} is secure against in-region fraud attack with probability $\alpha \geq \max(\alpha_u, \alpha_\ell)$ in Def 2.
2. If Π^{DUB}_{rect} and Π^{DLB}_{rect} are secure against man-in-the-middle attack with probability β_u, β_ℓ in Def. 6 and Def. 9 respectively, then Π_{rect} is secure against in-region man-in-the-middle with probability $\beta \geq \max(\beta_u, \beta_\ell)$ in Def. 3.
3. If Π^{DUB}_{rect} and Π^{DLB}_{rect} are secure against collusion fraud with probability (γ_u, η_u) in Def. 7 and (γ_ℓ, η_ℓ) in Def. 10 respectively, then Π_{rect} is secure against

in-region collusion fraud with probability (γ, η) *where* $\gamma \geq \max(\gamma_u, \gamma_\ell)$ *and* $\eta \geq \max(\eta_u, \eta_\ell)$ *in Def. 4.*

Proof of theorem 1 is given in appendix C.

5 Optimizing Error

The basic Π_{rect} protocol covers $\mathcal{R}$ with a P-rect. For given locations of verifiers loc_{V_0}, loc_{V_1}, and and distance bounds $\{\ell_{V_0}, u_{V_0}\}, \{\ell_{V_1}, u_{V_1}\}$), the error in the coverage can be computed. In this paper we consider the total error which is $FA + FR$. To minimize this error, one can use a two step algorithm: (i) for fixed loc_{V_0}, loc_{V_1}, find $\{\ell_{V_0}, u_{V_0}\}, \{\ell_{V_1}, u_{V_1}\}$) that minimizes the error, Denote it by $E_{min}(loc_{V_0}, loc_{V_1})$. (ii) find loc_{V_0}, loc_{V_1} that minimizes $E_{min}(loc_{V_0}, loc_{V_1})$. Both these minimizations can be solved by exhaustive search, which for an $n \times n$ size universe $\mathcal{U}$ will have the cost of $O(n^4)$ each.

In the following we provide an efficient algorithm *FindOptimalBounds*, or *FOB* for short (Alg. 1) to solve (i). Let the *size of a P-rectangle* be the number of points in the rectangle. The algorithm works as follows.

Algorithm 1. *FindOptimalBounds* algorithm to find P-rectangle with maximum accuracy for Δ

Input:
Policy region $\mathcal{R}$, Verifiers' location loc_{V_1}, loc_{V_2}, P-square size Δ
Output:
P-rect with maximum *accuracy* for Δ

$R_{rect} \leftarrow initR_{rect}(R, loc_{V_0}, loc_{V_1})$ ▷ Initial P-rectangle, covering $\mathcal{R}$ completely
$R^{\Delta}_{rect} \leftarrow makeGrid(R_{rect}, \Delta)$ ▷ R_{rect} is subdivided into P-squares of size Δ
for each P-square $ps \in R^{\Delta}_{rect}$ **do**
    $ps.TA \leftarrow 0$; $ps.FA \leftarrow 0$
    for each point $p \in ps$ **do** ▷ Each point contributes to either TA or FA value of the P-square
        if $p \in \mathcal{R}$ **then**
            $ps.TA = ps.TA + 1$
        else
            $ps.FA = ps.FA + 1$
        end if
    end for
    $ps.accuracy = ps.TA - ps.FA$ ▷ See Expression 6 for accuracy.
end for
$OptR_{rect} \leftarrow MaxSubArray(R^{\Delta}_{rect})$ ▷ R^{Δ}_{rect}, which is a 2D array with each element representing a *ps.accuracy* value, is input to a Maximum Subarray Algorithm.
return $OptR_{rect}$

i) Selects an initial R_{rect} (Line 1). This rectangle $\mathcal{R} \subset R_{rect}$ is constructed by choosing the radii to touch the region $\mathcal{R}$;
ii) R_{rect} is subdivided into P-squares (equal size sides) of size Δ (Line 2). P-squares are used as measuring units, and is used to quantify the *accuracy* (given by expression 6 in section 3) of R_{rect} in covering $\mathcal{R}$;
iii) The P-rect that maximizes the *accuracy* (therefore minimizes total *error* - see Sec. 3) for this Δ, is found by formulating the *accuracy* as the objective function of a maximum sum sub-array problem and using an algorithm (presented in Algorithm 7, page 18 of [2]) to efficiently solve the problem (Line 3 – 14).

The output of *FOB* is $OptR_{rect}$, a contiguous 2D sub-array (P-rect) with maximum sum (Line 15), that is the *optimal P-rect* for P-squares of size Δ.

Lemma 1. *For fixed values of loc_{V_0}, loc_{V_1}, the initial P-rect in FindOptimalBounds algorithm achieves higher accuracy compared to any larger P-rect.*

Proof. Let, the initial P-rectangle be denoted by $initR_{rect}$. This rectangle is chosen to be the smallest P-rectangle that contains all points in $\mathcal{R}$. That is, $initR_{rect}$ has maximum TA. Let the false acceptance associated with this P-rectangle be $FA_{initR_{rect}}$. The accuracy of $initR_{rect}$ is given by, $A_{initR_{rect}} = TA_{max} - FA_{initR_{rect}}$ Let R_{rect} be a P-rectangle that is larger than $initR_{rect}$ and fully covers $\mathcal{R}$. The accuracy of R_{rect} is expressed as - $A_{R_{rect}} = TA_{R_{rect}} - FA_{R_{rect}}$. Because $initR_{rect}$ is the "smallest" P-rectangle that covers $\mathcal{R}$, R_{rect} must have larger false acceptance. That is, $FA_{R_{rect}} > FA_{initR_{rect}}$.

Because $TA_{max} \geq TA_{R_{rect}}$, we conclude that, $A_{initR_{rect}} > A_{R_{rect}}$

Theorem 2 (Optimality). *Let the maximum sub-array algorithm return a contiguous 2D sub-array with the largest sum. Then the FindOptimalBounds algorithm returns the P-rectangle with maximum accuracy, for loc_{V_0}, loc_{V_1} and P-square size Δ.*

Proof. A P-rectangle can be expressed as a 2D array with each point being an element of that array. *FOB* algorithm is initialized with a 2D array $initR_{rect}$ of size $m \times n$ (unit Δ). For maximum accuracy, using Lemma 1 we need not consider larger P-rectangles that contain $\mathcal{R}$. The accuracy is given by the size of the set $A_{initR_{rect}} = TA_{max} - FA_{initR_{rect}} = \mathcal{R} \cap initR_{rect} - initR_{rect} \setminus \mathcal{R}$. Thus the contribution of a point $initR_{rect}[x][y]$ to the accuracy is 1, if it is in $\mathcal{R} \cap initR_{rect}$ and -1, if it is in $initR_{rect} \setminus \mathcal{R}$.[5]

Let $OptR_{rect}$ denote the 2D sub-array with maximum sum that is returned by $MaxSubArray()$. Using expression 2 for maximum sum sub-array (see Sec. 2), the 2D array $OptR_{rect}$ can be written as-

[5] Here we consider equal weights for FA, FR. Section 6 shows a flexible way to define these errors.

$$OptR_{rect} = \max\left\{ \sum_{x=i,y=g}^{j,h} R_{rect}[x][y] | 1 \le i \le j \le m, 1 \le g \le h \le n \right\}$$
$$= \max\left\{ \sum_{x=i,y=g}^{j,h} \left(TA_{R_{rect}[x][y]} - FA_{R_{rect}[x][y]} \right) \right\}$$

The right hand side of this equation is the 2D sub-array of maximum accuracy, and this concludes the proof.

Location of the verifiers. The algorithm 1 assumes that the verifiers' location are outside $\mathcal{R}$, and satisfy the following restriction: the initial rings centered at the verifiers V_0 and V_1 must intersect pairwise. This is to ensure a well-formed P-rectangle is constructed. The restriction discards many candidate locations for the verifiers. We leave the problem of efficiently finding the location of the verifiers that results in the smallest error for future work. One can remove the restriction on the location of verifiers, including being outside region $\mathcal{R}$, by subdividing the region into smaller regions. See section 6.

Higher accuracy. One can increase the accuracy of the algorithm by subdividing $\mathcal{R}$ into sub-regions, and for each, choose verifiers' location and find upper and lower bounds (using FOB). We show this in sec 6.

6 Experimental Evaluation

The error in covering $\mathcal{R}$ with a P-rect depends on the shape of $\mathcal{R}$, the number of subregions and the distance bounds. We consider the following cases for four policy regions shown in Figure 5.
- *Direct approach:* $\mathcal{R}$ is completely covered by the P-rect formed by rings centered at V_0 and V_1 and being the narrowest rings that contain all locations of $\mathcal{R}$. The resulting P-rect is the smallest P-rectangle covering $\mathcal{R}$ completely (Fig. 6a).
- *Basic FindOptimalBounds algorithm (FOB):* Fig 6b shows the implementation of basic error optimization algorithm presented in Section 5.
- *FindOptimalBounds with adjusted verifiers' location (FOB_{loc}):* We have adjusted the verifiers' locations heuristically to observe the impact on accuracy.
- *FindOptimalBounds algorithm with partitioned regions (FOB_{part}):* We partitioned each policy region into two smaller regions, and applied *FindOptimalBounds* algorithm on each independently. Figure 6c,6d show this settings.

Experimental setup. We take images from Google Map for point-set representation of the policy region, where the pixels represent points. We use "road-map" images with zoom level of 17, and of dimension 640×640 containing the policy region $\mathcal{R}$. Each pixel represents 0.7503 meters, which is obtained using the formula for "ground resolution" [17]. Ground resolution is the distance on the ground that can be represented by a single pixel in the map. We convert it into binary image containing only the policy region and store values for all the pixels

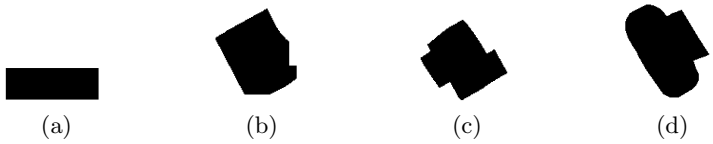


Fig. 5. Policy regions (from left to right): Building B1, B2, B3, B4 in binary image. We considered both regular shaped (B1) and relatively irregularly shaped regions (B2, B3, B4) to provide diversity to the experiment

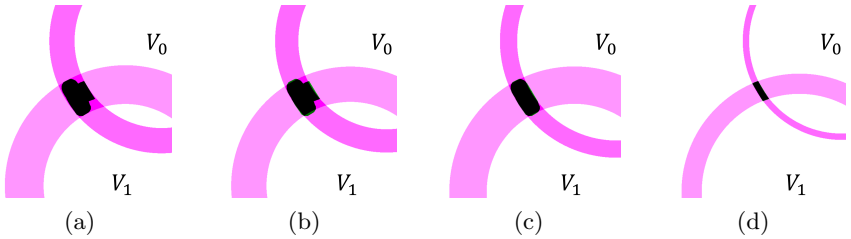


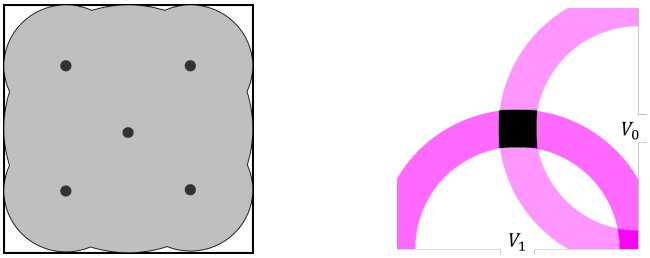
Fig. 6. (a) B4 is covered using direct approach, each ring touches two sides of the region (b) FOB approach: to reduce total error, a small amount of false rejection area is introduced (c, d) Partitiong B4 into two separate regions and applying FOB on each.

in a binary matrix. Measurements, including locations, distance, area and errors are all in pixels.

Error and coverage comparison. Table 1 compares four approaches when applied to B1, B2, B3, B4.Notice that comparatively "regular" shaped policy regions (e.g., B1 in Fig. 5a,) can be covered more accurately than other regions; if we compare the best found errors, B1 has (FA, FR) error only $(0.81, 3.89)\%$ against $(4.16, 5.58)\%$ (B2), $(3.34, 4.53)\%$ (B3) and $(5.16, 4.4)\%$ (B4). FOB_{part} algorithm reduces this irregularity to some extent reduces the total error of FOB by $7.82(B2), 15.48(B3)$ and $12.78\%(B4)$. Our algorithm trades much better than naively covering a region (the direct approach), FOB reduces total error from direct approaches by $10.84(B1), 12.68(B2), 10.78(B3)$ and $4.31\%(B4)$.

Comparison to existing approaches. Computing optimal bounds for verifiers so that the two types of errors are optimized - is only attempted once in existing literature on in-region verification and localization methods, by Sastry *et al.*[16].They have placed 5 verifiers inside a 100m by 100m room. They were able to achieve a coverage (True Acceptance) of 93% with 7% total error.We compare by considering a policy region of 100×100 resolution in the universe of 640×640 pixels. Each pixel represents 1m, so we replicate the scenario of covering a 100m by 100m room. Using two verifiers, we achieved a 96.4% coverage (TA) and 4.1% total error. Figure 7 illustrates the two approaches.

FA, FR weight analysis. In some applications FA is more tolerable, while in others FR. A notable advantage of our error formulation (Equation 5) is that it



(a) (b)

Fig. 7. (a) Sastry *et. al.* [16]: 5 nodes manually placed inside the region to provide a coverage (TA) of 93% (b) *FindOptimalBounds*: 2 nodes manually placed outside the region to provide a coverage (TA) of 96.4%. It also reduces total error from 7% to 4.1%.

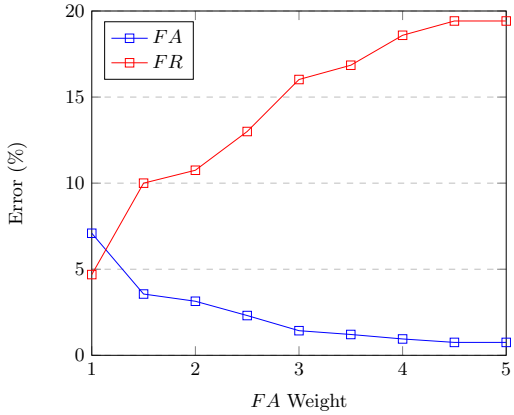


Fig. 8. False acceptance weight VS error. Increasing FA weight results in lesser FA error, and higher FR error. Service providers can select a weight that suits their security goal the most.

can be adjusted to capture requirements of different applications. We give the concept of weighted error metric: ${E_{\Pi,\mathcal{R}}}^{w} = W_{FA} \times FA_{\Pi,\mathcal{R}} + W_{FR} \times FR_{\Pi,\mathcal{R}}$.The increased weight for FA reduces FA error. For this analysis, we considered policy region $B2$ (fig. 5b) and FOB_{Loc} approach, and found that for FA weights $\{1, 2, 3, 4, 5\}$, the resulting FA errors are $\{7.09, 3.14, 1.43, 0.95, 0.75\}$%. Figure 8 shows the results. For this analysis we considered policy region $B2$ (fig. 5b) and FOB_{Loc} approach.

7 Related Work

There are hundreds of papers on location aware security and services. Because of space we only consider those that are directly relevant and consider location verification with respect to a region. As noted earlier, our goal, that is to provide

		Direct	FOB	FOB_{Loc}	FOB_{part}
B1	TA	100	94.2	94.1	96.1
	FA	20.94	4.31	2.32	0.81
	FR	0	5.79	5.89	3.89
	E	20.94	10.1	8.22	4.71
B2	TA	100	90.77	95.3	94.41
	FA	30.24	8.33	7.09	4.16
	FR	0	9.22	4.69	5.58
	E	30.24	17.56	11.78	9.74
B3	TA	100	77.63	90.61	95.47
	FA	34.13	9.5	9.69	3.34
	FR	0	13.84	9.38	4.53
	E	34.13	23.35	19.08	7.87
B4	TA	100	93.35	94.7	95.6
	FA	26.65	15.7	12.21	5.16
	FR	0	6.64	5.3	4.4
	E	26.65	22.34	21.07	9.56

Table 1. Four coverage approaches are applied to B1, B2, B3, B4. E = FA + FR is total error. Best found FA, FR and E range from 0.81 to 5.16%, 3.89 to 5.58% and 4.71 to 9.74%. Best found total coverage ranges from 94.41 to 96.1%

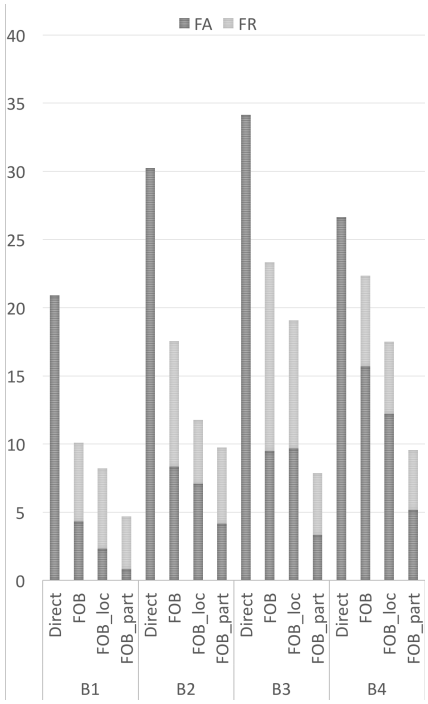


Fig. 9. FA, FR Error Comparison among different approaches when applied to B1, B2, B3, B4. The regularly shaped region B1 has the lowest FA and FR errors compared to other relatively irregular shaped regions, in all four approaches.

provably secure authentication for users inside $\mathcal{R}$ together with quantifiable error, for arbitrary region, is novel and not shared by any existing work. The system in [22] provides location verification for a region without using a secret key and without requiring user authentication.

Secure positioning in multiple verifier settings is considered in [6], who proved that security against multiple adversaries (adversaries at multiple locations) is only achievable in the bounded retrieval model. [26] use bounded retrieval model, and like us, they also take advantage of directional antennas to provide in-region security. However, they cannot provide security against adversaries that reside inside the region.

[25] proposed an in-region location verification that uses the inconsistencies between claimed location of the sensor (prover) and observations of their neighbor sensors to detect a false location claim. However, their security is dependent on other sensors' trust, which is often not desirable.

Numerous distance upper bounding protocols have been proposed to date [4, 3, 21, 9]. However the only distance lower bounding protocol with provable security against three main kind of attacks is [27]. inRA uses the formal model and protocol constructions of [21] and [27] for its DUB and DLB components.

8 Concluding Remarks

We motivated and defined the problem of in-region authentication, and defined correctness and security of inRA protocols for a region $\mathcal{R}$. We proposed an approach to constructing secure inRA protocols that uses distance bounding protocols to cover $\mathcal{R}$ with a P-rect, and gave an efficient algorithm to optimize the P-rect by minimizing the total error. We also proposed a basic two-verifier protocol with provable properties. Our approach provides flexibility to define error functions that are suitable for particular applications, and increase accuracy by choosing more verifiers.

We showed error performance of our optimization algorithm on different shaped policy region and verified improved accuracy when the region is subdivided into two. Optimizing error under real life constraints on the location of verifiers, the number of verifiers, particular error function, and optimization in three dimensional spaces are challenging directions for future research.

References

1. Ahmadi, A., Safavi-Naini, R.: Distance-bounding identifiaction. In: 3rd International Conference on Information Systems Security and Privacy (2017)
2. Bae, S.E.: Sequential and parallel algorithms for the generalized maximum subarray problem. Ph.D. thesis, University of Canterbury (2007)
3. Boureanu, I., Mitrokotsa, A., Vaudenay, S.: Secure and lightweight distance-bounding. In: Lightweight Cryptography for Security and Privacy, pp. 97–113. Springer (2013)
4. Brands, S., Chaum, D.: Distance-bounding protocols. In: Advances in Cryptology - EUROCRYPT'93. pp. 344–359. Springer (1993)
5. Čapkun, S., Hubaux, J.P.: Secure positioning of wireless devices with application to sensor networks. In: Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies. vol. 3, pp. 1917–1928. IEEE (Mar 2005)
6. Chandran, N., Goyal, V., Moriarty, R., Ostrovsky, R.: Position based cryptography. In: Advances in Cryptology-CRYPTO 2009, pp. 391–407. Springer (2009)
7. Chiang, J.T., Haas, J.J., Hu, Y.: Secure and precise location verification using distance bounding and simultaneous multilateration. In: Proceedings of the 2nd ACM conference on Wireless Network Security (WiSec '09). pp. 181–192. New York, NY, USA (2009)
8. Desmedt, Y.: Major security problems with the unforgeable(feige)-fiat-shamir proofs of identity and how to overcome them. In: Proceedings of SECURICOM. vol. 88, pp. 15–17 (1988)
9. Dürholz, U., Fischlin, M., Kasper, M., Onete, C.: A formal approach to distance-bounding rfid protocols. In: International Conference on Information Security. pp. 47–62. Springer (2011)

10. Fan, T.H., Lee, S., Lu, H.I., Tsou, T.S., Wang, T.C., Yao, A.: An optimal algorithm for maximum-sum segment and its application in bioinformatics. In: International Conference on Implementation and Application of Automata. pp. 251–257. Springer (2003)
11. Francillon, A., Danev, B., Čapkun, S.: Relay attacks on passive keyless entry and start systems in modern cars. In: NDSS (2011)
12. Grenander, U.: Pattern Analysis, Applied Mathematical Sciences, vol. 24. Springer New York, New York, NY (1978). https://doi.org/10.1007/978-1-4684-9354-2
13. Hammad, A., Faith, P.: Location based authentication (Aug 1 2017), uS Patent 9,721,250
14. Metz, C.E.: Basic principles of roc analysis. In: Seminars in Nuclear Medicine. vol. 8, pp. 283–298. Elsevier (1978)
15. Rasmussen, K.B., Castelluccia, C., Heydt-Benjamin, T.S., Čapkun, S.: Proximity-based access control for implantable medical devices. In: Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS'09). pp. 410–419. Chicago, Illinois, USA (Nov 2009)
16. Sastry, N., Shankar, U., Wagner, D.: Secure verification of location claims. In: Proceedings of the 2nd ACM Workshop on Wireless Security. pp. 1–10. New York, NY, USA (2003)
17. Schwartz, J.: Bing maps tile system. https://msdn.microsoft.com/en-us/library/bb259689.aspx, accessed: 2016-04-13
18. Singelee, D., Preneel, B.: Location verification using secure distance bounding protocols. In: IEEE International Conference on Mobile Adhoc and Sensor Systems Conference, 2005. pp. 7–pp. IEEE (2005)
19. Takaoka, T.: Efficient algorithms for the maximum subarray problem by distance matrix multiplication. Electronic Notes in Theoretical Computer Science **61**, 191–200 (2002)
20. Takaoka, T., Pope, N.K., Voges, K.E.: Algorithms for data mining. In: Business Applications and Computational Intelligence, pp. 291–315. Igi Global (2006)
21. Vaudenay, S., Boureanu, I., Mitrokotsa, A., et al.: Practical & provably secure distance-bounding. In: Proceedings of the 16th Information Security Conference (2013)
22. Vora, A., Nesterenko, M.: Secure location verification using radio broadcast. Dependable and Secure Computing, IEEE Transactions on **3**(4), 377–385 (2006)
23. Warner, J.S., Johnston, R.G.: A simple demonstration that the global positioning system (gps) is vulnerable to spoofing. Journal of Security Administration **25**(2), 19–27 (2002)
24. Weddell, S., Langford, B.: Hardware implementation of the maximum subarray algorithm for centroid estimation. In: Proc. of Twenty-first Image and Vision Computing Conference New Zealand (IVCNZ 2006). pp. 511–515 (2006)
25. Wei, Y., Guan, Y.: Lightweight location verification algorithms for wireless sensor networks. IEEE Transactions on Parallel and Distributed Systems **24**(5), 938–950 (2013)
26. Yang, R., Xu, Q., Au, M.H., Yu, Z., Wang, H., Zhou, L.: Position based cryptography with location privacy: A step for fog computing. Future Generation Computer Systems (2017)
27. Zheng, X., Safavi-Naini, R., Ahmadi, H.: Distance lower bounding. In: Information and Communications Security, pp. 89–104. Springer (2014)
28. Zickuhr, K.: Location-based services. Pew Research pp. 679–695 (2013)

A Explanation on the Erasure Sequence

The security of DLB protocol presented in [27] stands on the assumption that the prover devices are memory restricted in a way that they are not able to store malicious codes in the memory - as malicious codes potentially allow the device to delay their response to verifier's challenge. Erasure sequence solves this problem, which is a random binary sequence - sent by the verifier in each round of fast exchange phase, just after the challenge. Size of the erasure sequence is determined so that all the memory of the prover device, barring the part required by prover to execute the protocol (code to execute the protocol, responses that are going to be used in the future), is replaced by the sequence. Prover must prove to the verifier that it has actually stored the whole erasure sequence, and this is done by making the prover send a *proof of erasure* back to the verifier in each round, just after the response bit. The generation of proof of erasure is as: prover reads the erasure sequence in reverse order, then applies a cryptographic hash function on it. This design forces the prover in storing the complete erasure sequence before sending the proof to the verifier.

B Security models of Π_{rect}^{DUB} and Π_{rect}^{DLB}

Security model of Π_{rect}^{DUB} presented in [21]

Definition 5. *(DF-resistance of Π_{rect}^{DUB}) Π_{rect}^{DUB} is α_U-resistant to DF attack if $(\forall s)(\forall P^*)$ $(\forall loc_V)$ such that $d(loc_V, loc_{P^*}) > B_U)(\forall r_k)$ we have,*

$$Pr_{r_V}\left[Out_V = 1 : \begin{array}{l} x \leftarrow Gen(1^s, r_k) \\ P^*(x) \leftrightarrow V(x; r_V) \end{array}\right] \leq \alpha_U. \tag{13}$$

Here P^* is any dishonest prover and in a concurrent setting, a polynomially bounded number of honest $P(x')$ and $V(x')$ close to $V(x)$, with independent x', are allowed.

Definition 6. *(MiM-Resistance of Π_{rect}^{DUB}) Π_{rect}^{DUB} is β_U-resistant to MiM attack if $(\forall s)$ $(\forall m, l, z)$ that are polynomially bounded, $(\forall \mathcal{A}_1, \mathcal{A}_2)$ polynomially bounded, for all locations such that $d(loc_{P_j}, loc_V) > B_U$, where $j \in \{m+1, ..., l\}$, we have*

$$Pr_{r_V}\left[Out_V = 1 : \begin{array}{l} (x) \leftarrow Gen(1^s, r_k) \\ P_1(x), ..., P_m(x) \leftrightarrow \mathcal{A}_1 \leftrightarrow V_1(x), ..., V_z(x) \\ P_{m+1}(x), ..., P_l(x) \leftrightarrow \mathcal{A}_2(View_{\mathcal{A}_1}) \leftrightarrow V(x) \end{array}\right] \leq \beta_U. \tag{14}$$

Here, the attacker is represented by $(\mathcal{A}_1, \mathcal{A}_2)$, and the probability is over all random coins of the protocol. $View_{\mathcal{A}_1}$ is the final view of $\mathcal{A}_1$. This definition allows polynomially bounded number of $P(x'), P^*(x')$, and $V(x')$ with independent x', anywhere.

The attacker can have a learning phase $\mathcal{A}_1$ during which it interacts with m honest provers and z verifiers. It then uses the view of $\mathcal{A}_1$ in the attack phase, and engages as $\mathcal{A}_2$ in $l-m$ protocol instances between honest provers and the target verifier.

Definition 7. *(CF-Resistance of Π^{DUB}_{rect}) Π^{DUB}_{rect} is (γ_U, η_U)-resistant to collusion fraud if $(\forall s)$ $(\forall P^*)$ $(\forall loc_{V_0})$ such that $d(loc_{V_0}, loc_{P^*}) > B_U, \forall \mathcal{A}^{CF} ppt.$ if we have,*

$$Pr_{r_V}\left[Out_{V_0} = 1 : \begin{array}{l}(x) \leftarrow Gen(1^s)\\ P^{(*)}(x) \leftrightarrow \mathcal{A}^{CF} \leftrightarrow V_0(x)\end{array}\right] \geq \gamma_U, \tag{15}$$

then there exists an extended[6] MiM attack with $m, l, z, A_1, A_2, P_i, P_j, V_i$ that uses interaction with P and P^* both, and V in the learning phase, such that,

$$Pr_{r_V}\left[Out_V = 1 : \begin{array}{l}(x) \leftarrow Gen(1^s)\\ P_1^{(*)}(x), ..., P_m^{(*)}(x) \leftrightarrow \mathcal{A}_1 \leftrightarrow V_1(x), ..., V_z(x)\\ P_{m+1}(x), ..., P_l(x) \leftrightarrow \mathcal{A}_2(View_{\mathcal{A}_1}) \leftrightarrow V(x)\end{array}\right] \geq \eta_U. \tag{16}$$

Here $P^{(*)}$ is a prover that is either P or P^*. We have $d(loc_{P_j}, loc_V) > B_U$, for all $j \in \{m+1, l\}$. We implicitly allow a polynomially bounded number of $P(x'), P^*(x')$, *and* $V(x')$ with independent (x'), anywhere but no honest participant is close to V_0.

Security model of Π^{DLB}_{rect} presented in [27]

Definition 8. *(DF-resistance of Π^{DLB}_{rect}) Π^{DLB}_{rect} is α_U-resistant to DF attack if $(\forall s)(\forall P^*)$ $(\forall loc_V)$ such that $d(loc_V, loc_{P^*}) < B_L)(\forall r_k)$, we have,*

$$Pr_{r_V}\left[Out_V = 1 : \begin{array}{l}x \leftarrow Gen(1^s, r_k)\\ P^*(x) \leftrightarrow V(x; r_V)\end{array}\right] \leq \alpha_L. \tag{17}$$

Here P^* is any dishonest prover and in a concurrent setting, a polynomially bounded number of honest $P(x')$ and $V(x')$ far away from $V(x)$, with independent x', are allowed.

Definition 9. *(MiM-Resistance of Π^{DLB}_{rect}) Π^{DLB}_{rect} is β_L-resistant to MiM attack if $(\forall s)(\forall m, l, z)$ that are polynomially bounded, $(\forall \mathcal{A}_1, \mathcal{A}_2)$ polynomially bounded, for all locations such that $d(loc_{P_j}, loc_V) < B_L$, where $j \in \{m+1, ..., l\}$, we have*

$$Pr_{r_V}\left[Out_V = 1 : \begin{array}{l}(x) \leftarrow Gen(1^s, r_k)\\ P_1(x), ..., P_m(x) \leftrightarrow \mathcal{A}_1 \leftrightarrow V_1(x), ..., V_z(x)\\ P_{m+1}(x), ..., P_l(x) \leftrightarrow \mathcal{A}_2(View_{\mathcal{A}_1}) \leftrightarrow V(x)\end{array}\right] \leq \beta_L \tag{18}$$

[6] learning phase includes P^*, which allows the adversary to interact with it

Here, the attacker is represented by $(\mathcal{A}_1, \mathcal{A}_2)$, and the probability is over all random coins of the protocol. $View_{\mathcal{A}_1}$ is the final view of $\mathcal{A}_1$. This definition allows polynomially bounded number of $P(x'), P^*(x')$, and $V(x')$ with independent x', anywhere.

The attacker can have a learning phase $\mathcal{A}_1$ during which it interacts with m honest provers and z verifiers. It then uses the view of $\mathcal{A}_1$ in the attack phase, and engages as $\mathcal{A}_2$ in $l - m$ protocol instances between honest provers and the target verifier.

Definition 10. *(CF-Resistance of Π^{DLB}_{rect}) Π^{DLB}_{rect} is (γ_L, η_L)-resistant to collusion fraud if* $(\forall s)$ $(\forall P^*)$ $(\forall loc_{V_0}$ *such that* $d(loc_{V_0}, d(loc_{V_0}, loc_{P^*}) < B_L)$ $(\forall \mathcal{A}^{CF}$ *ppt.) such that*

$$Pr_{r_V}\left[Out_{V_0} = 1 : \begin{array}{c}(x) \leftarrow Gen(1^s) \\ P^{(*)}(x) \leftrightarrow \mathcal{A}^{CF} \leftrightarrow V_0(x)\end{array}\right] \geq \gamma_L, \tag{19}$$

then there exists an extended MiM attack with $m, l, z, A_1, A_2, P_i, P_j, V_i$ that uses interaction with P and P^* both, and V in the learning phase, such that,

$$Pr_{r_V}\left[Out_V = 1 : \begin{array}{c}(x) \leftarrow Gen(1^s) \\ P_1^{(*)}(x), ..., P_m^{(*)}(x) \leftrightarrow \mathcal{A}_1 \leftrightarrow V_1(x), ..., V_z(x) \\ P_{m+1}(x), ..., P_l(x) \leftrightarrow \mathcal{A}_2(View_{\mathcal{A}_1}) \leftrightarrow V_x\end{array}\right] \geq \eta_L. \tag{20}$$

Here $P^{(*)}$ is a prover that is either P or P^*. We have $d(loc_{P_j}, loc_V) < B_L$, for all $j \in \{m+1, l\}$. We implicitly allow a polynomially bounded number of $P(x'), P^*(x')$, *and* $V(x')$ with independent (x'), anywhere but no honest participant is far away from V_0.

C Proofs

Theorem 1.

Proof (inF Security). We prove that if there is an inF adversary against the inRA protocol, then either there is a DF adversary against the DUB protocol, or there is a DF adversary against the DLB protocol. More specifically-
If there exists a PPT malicious prover P^* such that $loc_{P^*} \notin \mathcal{R}$, and has an inF success probability $\alpha^{'} > \alpha$ in inRA protocol $\Pi_{\mathcal{R}}$, then

1. we can construct a PPT malicious prover P^*_u such that $d(loc_V, loc_{P^*_u}) > B_U$, and has a DF success probability $\alpha^{'}_u > \alpha_u$ in DUB protocol Π^{DUB}_{rect}; or
2. we can construct a PPT malicious prover P^*_ℓ such that $d(loc_V, loc_{P^*_u}) < B_L$, and has a DF success probability $\alpha^{'}_\ell > \alpha_\ell$ in DLB protocol Π^{DLB}_{rect}.

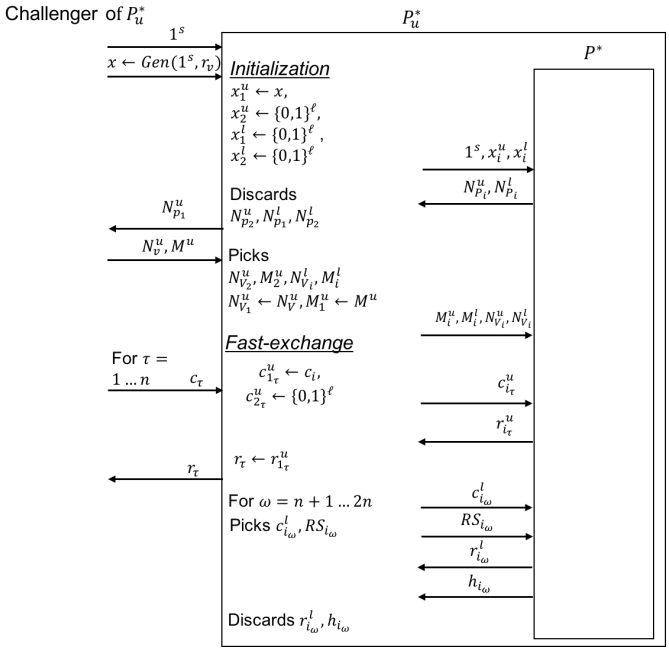


Fig. 10. inF security of Π_{rect} - Proof by reduction.

To prove 1, we assume that there exists a malicious inF prover P^* against the inRA protocol, and $d(loc_V, loc_{P^*}) > B_L$.

The malicious prover P_u^* (that is outside the upper bound B_U) must convince a challenger $\mathcal{C}$ (that acts as the verifier of P_u^* in a Π_{rect}^{DUB} protocol instance), that it is inside the distance upper bound B_U from $\mathcal{C}$. In order to do this, P_u^* interacts with P^* and acts as a challenger of P^* (i.e., acts as both verifiers V_0, V_1 for P^* in a Π_{rect} protocol instance). In other words, P^* is used as a subroutine of P_u^*. Figure 10 shows the setup and the interactions among $\mathcal{C}$, P_u^* and P^*.

The challenger $\mathcal{C}$ sets the system and sends to P_u^*, the security parameter 1^s and the secret key $x \leftarrow Gen(1^s, r_V)$.

During the initialization phase of Π_{rect}^{DUB} with $\mathcal{C}$, P_u^* does the following:

1. Set $x_1^u = x$. This is the secret key shared with the verifier V_0 in distance upper bounding. The other secret keys required by P^* are selected randomly by P_u^*: $x_2^u, x_1^l, x_2^l \in \{0,1\}^\ell$, ℓ is the length of secret key x.
2. Sends security parameter 1^s, secret keys x_i^u, x_i^l (for $i = \{1,2\}$) to P^* which acts as a subroutine.
3. Receives $N_{p_i}^u, N_{p_i}^\ell$ generated by P^* as a part of its own initialization phase of Π_{rect}; each of these is a k-bit binary string.
4. Discards all but $N_{p_1}^u$ and sends it to $\mathcal{C}$.
5. Receives N_v^u, M^u, of length k and $2n$, respectively, from $\mathcal{C}$.

6. Sets $N^u_{V_0} = N^u_V, M^u_1 = M^u$, and Randomly selects binary strings $N^u_{V_1}, N^l_{V_0}$, $N^l_{V_1}$, each of length k, and M^u_2, M^l_1, M^l_2, each of length $2n$, respectively, and sends all these strings to P^*.

In the fast-exchange phase of Π^{DUB}_{rect}, P^*_u does the following -

1. In each of the n rounds of the fast exchange phase Π^{DUB}_{rect}, receives challenge bit $c_\tau, 1 \leq \tau \leq n$ from $\mathcal{C}$, sets $c^u_{1_\tau} = c_\tau$. $c^u_{2_\tau}$ is set randomly. It sends $c^u_{1_\tau}, c^u_{2_\tau}$ to P^*. Upon receiving the response bit $r^u_{1_\tau}, r^u_{2_\tau}$ from P^*, sends $r^u_{1_\tau}$ back to $\mathcal{C}$, discards $r^u_{2_\tau}$.
2. In order to simulate the challenges of DLB instance for P^*, P^*_u does the following:
   runs for another n rounds with the delay between any two challenges chosen the same as that of a DLB instance Π^{DLB}_{rect}. It randomly generates two challenge bits $c^l_{1_\tau}, c^l_{2_\tau}$, and two binary strings RS_1, RS_2, in each round (RS_i is the erasure sequence for DLB corresponding to verifier V_i); it sends $c^l_{1_\tau}, c^l_{2_\tau}$ to P^* immediately followed by RS_1, RS_2; and receives $r^l_{i_\tau}, h_{i_\tau}$, $(i = \{1, 2\})$ and discards all.

Now, the steps given above clearly takes polynomial time to execute. Therefore, since P^* takes polynomial time (as assumed), P^*_u's running time is polynomial as well.

P^*_u is indistinguishable from a real challenger for P^* as it (P^*_u) forwards all the messages of its challenger $\mathcal{C}$ to P^*, and the rest of the messages (required for V_0's DLB instances in P^* and V_1's DUB and DLB instances) are randomly chosen with the same length and completely indistinguishable from the original messages.

The response table a^u_1 generated by P^* is the only meaningful response table here. The other tables a^l_1, a^u_2, a^l_2 do not affect P^* in generating the correct response for V_0's DUB instance in time, as response tables are independent from each other and from other verifiers.

As per assumptions, P^* has a inF success probability of $\alpha' > \alpha$ in inRA, which in this case, is only the success probability of its DUB instance with verifier V_0. According to inF definition [Def 2], it follows that all the responses $r^u_{1_\tau}$ from P^* must be correct and reach verifier in time. Therefore, all the responses r_τ from P^*_u must be correct as well, and reach the verifier (challenger $\mathcal{C}$) in time. It implies that according to DUB DF definition [Def 5], P^*_u has a DF success probability of $\alpha'_u > \alpha_u$ in DUB.

For the proof of 2, we construct the PPT malicious prover P^*_ℓ using a similar method.

Proof (inMiM Security).

We show that, if there exists a PPT adversary $\mathcal{A}$ who has an inMiM success probability $\beta' > \beta$ in inRA protocol Π_{rect}, then

1. either we can construct a PPT adversary $\mathcal{A}_u$ who has a MiM success probability $\beta'_u > \beta_u$ in DUB protocol Π^{DUB}_{rect}; or

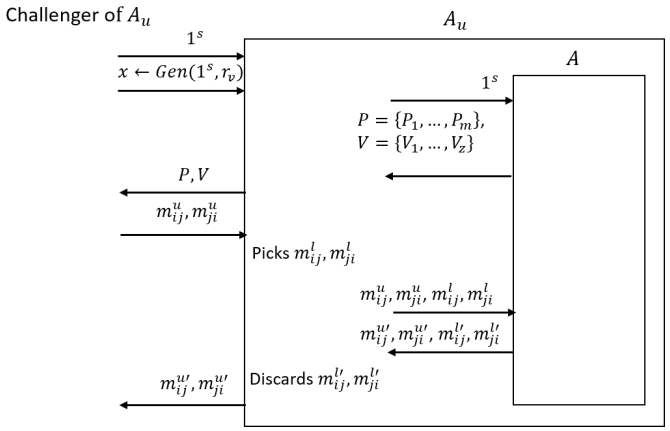


Fig. 11. inMiM security of Π_{rect} - Proof by reduction: Learning Phase

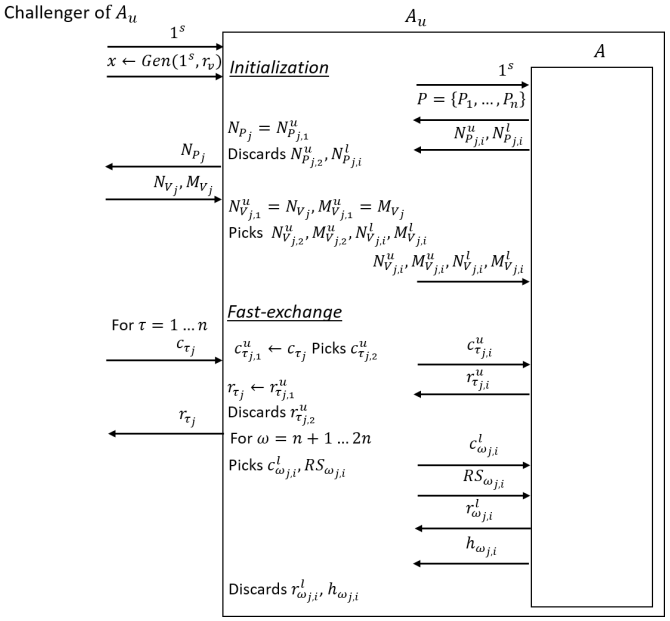


Fig. 12. inMiM security of Π_{rect} - Proof by reduction: Attack Phase

2. we can construct a PPT adversary $\mathcal{A}_\ell$ who has a MiM success probability $\beta'_\ell > \beta_\ell$ in DLB protocol Π^{DLB}_{rect}.

To prove 1, assume there is an adversary $\mathcal{A}$ as above, and suppose it cannot be used to construct $\mathcal{A}_\ell$.

The adversary $\mathcal{A}_u$, who does not have a key but is inside the distance bound B_U from a challenger $\mathcal{C}$ and interacts with multiple honest provers, aims to make $\mathcal{C}$ (that acts as a verifier for $\mathcal{A}_u$) output *accept* in a Π^{DUB}_{rect} protocol instance. The adversary has access to $\mathcal{A}$ as described above, In order to achieve its goal, $\mathcal{A}_u$ interacts with $\mathcal{A}$ and acts as a challenger of $\mathcal{A}$ (i.e., acts as both verifiers V_0, V_1 for $\mathcal{A}$ in a Π_{rect} protocol instance). In other words, $\mathcal{A}$ is used as a subroutine of $\mathcal{A}_u$. Figure 11 and 12 shows the setup and the interactions among $\mathcal{C}$, $\mathcal{A}_u$ and $\mathcal{A}$.

The challenger $\mathcal{C}$ sets up the system and provides the security parameter 1^s to $\mathcal{A}_u$.

In the learning stage of Π^{DUB}_{rect}, $\mathcal{A}_u$ does the following-

1. Sends security parameter 1^s to $\mathcal{A}$.
2. Receives the set of m provers $\mathcal{P} = \{P_1, ..., P_m\}$ and the list of z set of verifiers $\mathbb{V} = \{\mathcal{V}_1, ..., \mathcal{V}_z\}$ from $\mathcal{A}$. We assume that each verifier set $\mathcal{V}_i \in \mathbb{V}$ consists of a pair of verifiers.
3. Sends $\mathcal{P}, \mathbb{V}$ to challenger.
4. Receives message m^u_{ij} sent by prover P_i to Verifier V_j Also, receives message m^u_{ji} sent by verifier V_j to prover P_i.
5. Randomly selects strings m^l_{ij} and m^l_{ji}. Sends all these messages (m^u_{ij},m^u_{ji},m^l_{ij} and m^l_{ji}) to $\mathcal{A}$.
6. Receives tampered message $m^{u'}_{ij}, m^{l'}_{ij}$ from $\mathcal{A}$ originally sent by prover P_i to verifier V_j. Also, receives tampered message $m^{u'}_{ji}, m^{l'}_{ji}$ from $\mathcal{A}$ originally sent by verifier V_j to prover P_i. Discards $m^{l'}_{ji}, m^{l'}_{ij}$ and sends the remaining messages to the challenger (to corresponding verifiers and provers).

In the initialization phase of the attack stage in Π^{DUB}_{rect}, $\mathcal{A}$ does the following. In the following, i, j are used to index the verifier and the prover, respectively.

1. Sends security parameter 1^s to $\mathcal{A}$.
2. Receives a set of θ provers $\mathcal{P} = \{P_1, ..., P_\theta\}$ (a different set of provers from the learning phase) from $\mathcal{A}$.
3. Receives $N^u_{P_{j,i}}, N^\ell_{P_{j,i}}$, $(m+1 \leq i \leq \ell)$ from $\mathcal{A}$, each of these is a k-bit binary string. $N^u_{P_{j,i}}$ is the DUB nonce that is generated by the prover P_j, and is intended to reach the verifiers V_i, $i = \{1, 2\}$, $j = \{1, .., \theta\}$.
4. Sets $N_{P_j} = N^u_{P_{j,1}}$ and sends to challenger $\mathcal{C}$. Discards the rest of the nonces.
5. Receives N_{V_j}, M_{V_j} of length k and $2n$ respectively, from the challenger; these are intended to reach prover P_j.
6. Sets $N^u_{V_{j,1}} = N_{V_j}$ and $M^u_{V_{j,1}} = M_{V_j}$. Randomly selects binary strings $N^u_{V_{j,2}}$, $N^\ell_{V_{j,i}}$ of length k and $M^u_{V_{j,2}}, M^\ell_{V_{j,i}}$ of length $2n$ respectively and sends all of these to $\mathcal{A}$.

In the fast-exchange phase of attack stage in Π_{rect}^{DUB}, $\mathcal{A}_u$ does the following -

1. For each round τ, $\tau = \{1, ..., n\}$ of the fast exchange phase of Π_{rect}^{DUB} for prover P_j $(1 \leq j \leq \theta)$, $\mathcal{A}_u$ receives a challenge bit c_{τ_j} from $\mathcal{C}$.
2. Sets $c^u_{\tau_{j,1}} = c_{\tau_j}$, picks $c^u_{\tau_{j,2}}$ randomly. Sends both challenge bits to $\mathcal{A}$.
3. Receives responses $r^u_{\tau_{j,i}}$ from $\mathcal{A}$. Sets $r_{\tau_j} = r^u_{\tau_{j,1}}$ and sends back to $\mathcal{C}$. Discards $r^u_{\tau_{j,2}}$.
4. In order to simulate the challenges of the Π_{rect}^{DLB} instance for $\mathcal{A}$, $\mathcal{A}_u$ does the following:
   runs for another n rounds (rounds indexed with ω) with the delay between the two challenges chosen to be the same as that of a Π_{rect}^{DLB} instance; randomly generates challenge bits $c^{\ell}_{\omega_{j,i}}$ and a binary string $RS_{\omega_{j,i}}$ in each round ($RS_{\omega_{j,i}}$ is the erasure sequence); sends $c^{\ell}_{\omega_{j,i}}$ to the $\mathcal{A}$, immediately followed by $RS_{\omega_{j,i}}$; receives responses $r^{\ell}_{\omega_{j,i}}, h_{\omega_{j,i}}$ from $\mathcal{A}$, and discards both.

Now, the steps given above clearly take polynomial time to execute, therefore, because A takes polynomial time (as assumed), A_u's running time is also polynomial.

A_u is indistinguishable from a real challenger for A as A_u forwards all the messages of its challenger $\mathcal{C}$ to A, and the rest of the messages (required for V_0's Π_{rect}^{DLB} instances in $\mathcal{A}_{\sqcap}$ and V_1's Π_{rect}^{DUB} and Π_{rect}^{DLB} instances) are randomly chosen with the same length and completely indistinguishable from the original messages.

The response table a^u_1 generated by A is the only meaningful response table here. The other tables a^l_1, a^u_2, a^l_2 do not affect A in generating correct response for V_0's Π_{rect}^{DUB} instance in time, as response tables are independent from each other and from other verifiers.

As per the assumptions, A has a inMiM success probability of $\alpha' > \alpha$ in inRA, which in this case, is the success probability of only its Π_{rect}^{DUB} instance with verifier V_0. According to inRA inMiM definition [Def 3], it follows that all the responses $r^u_{1_\tau}$ from A must be correct and reach verifier in time. Therefore, all the responses r_{τ_j} from A_u must be correct as well, and reach the verifier (challenger $\mathcal{C}$) in time. It implies that according to Π_{rect}^{DUB} MiM definition [Def 6], A_u has a MiM success probability of $\beta'_u > \beta$ in Π_{rect}^{DUB}.

For the proof of 2, we construct the PPT adversary A_ℓ using a similar method.

Proof (inCF Security). Lets assume that as per the inCF resistance definition of inRA (Definition 4), we have an experiment $exp^{inCF} = P^*(x) \longleftrightarrow \mathcal{A}^{inCF} \longleftrightarrow \mathcal{V}_0(\mathcal{X})$, where P^* is a malicious prover who is outside the region $\mathcal{R}$. If $c_1, ..., c_n$ are some random challenges from one of the verifiers V, let us define $View_i$ to be the view of $\mathcal{A}^{CF}$ before receiving c_i from V. We also define a random variable W_i, where $W_i = w_i$ is all the information $\mathcal{A}^{CF}$ receives from P^*. It is important to note that w_i must be received before it is too late to send the response r_i to V. This w_i is exploited in the later experiment, where the adversary $(\mathcal{A}_1, \mathcal{A}_2)$ will run a MiM attack.

As assumed, for each verifier $V_i \in \mathcal{V}$, the response tables a_u of the DUB protocol Π_{rect}^{DUB} and a_l of the DLB protocol Π_{rect}^{DLB} are independent from each

other and from other verifiers. We also assume that, at the end of the initialization of an instance of inRA protocol Π_{rect}, verifier V_0, V_1 and prover P have response tables that are, i) possibly different, and ii) in all cases are independent of adversary's combined view $(N_p^u, N_v^u, N_p^\ell, N_v^\ell, M^u, M^\ell)$.

Above assumptions enable us to divide $View_i$ into $View_i^u$ and $View_i^l$; this is because of the independence of the random keys and the choice of randomness during the initialization and fast exchange phase. Also, the information w_i received from P^* can be separated into w_i^u and w_i^l following the same argument.

Now, in the inCF experiment the attacker $\mathcal{A}^{inCF}$ succeeds if, (i) they can correctly respond to the upper bounding challenges from V given that A_1, A_2 will not succeed in the later MiM experiment; and also ii) respond to the lower bounding challenges from V given that A_1, A_2 will not succeed in the later MiM experiment.

For responding to upper bounding challenges, the relevant view is $View_i^u$ and the relevant information from P^* is w_i^u. Using Definition 7 of CF in Π_{rect}^{DUB}, the success probability of the adversary in Π_{rect} with this view, in the attack instance between the prover P^* and the verifier V, is the same as the success probability of a CF attacker in Π_{rect}^{DUB}, given that the later inMiM attack will have success probability the same as that of MiM attack in Π_{rect}^{DUB}.

A similar argument shows that for responding lower bounding challenges, the relevant view is $View_i^l$ and relevant information from P^* is w_i^l. Using Definition 10 of CF in Π_{rect}^{DLB} the success probability of the adversary in Π_{rect} with this view, in the attack instance between prover P^* and the verifier V, is the same as success probability of a CF attacker in Π_{rect}^{DLB}, given that the later inMiM attack will have success probability same as that of MiM attack Π_{rect}^{DLB}.

However, in a particular attack scenario in Π_{rect}, a malicious prover will always be able to succeed in at least one of the two sub-protocols Π_{rect}^{DUB}, Π_{rect}^{DLB}. A malicious prover can either be inside the lower bound B_L or outside the upper bound B_U for one of the verifiers. If it is inside the lower bound B_L, all it needs to do is behaving honestly (follows the protocol) in the distance upper bounding sub-protocol Π_{rect}^{DUB} and successfully prove itself to be inside the upper bound B_U (i.e., by definition, $B_L \leq B_U$). If the prover is outside the upper bound B_U, it behaves honestly in the distance lower bounding sub-protocol Π_{rect}^{DLB} and successfully prove itself to be outside the lower bound B_L.

This concludes the proof.