Practical Construction for Secure Trick-Taking Games Even With Cards Set Aside

Rohann Bella
INSA Centre Val de Loire,
Laboratoire d'informatique
fondamental d'Orléans, France

Xavier Bultel
INSA Centre Val de Loire,
Laboratoire d'informatique
fondamental d'Orléans, France
0000-0002-8309-8984

Céline Chevalier
CRED, Université Paris-Panthéon-Assas,
DIENS, École normale supérieure,
PSL Université,
CNRS, INRIA, Paris, France

Pascal Lafourcade
Université Clermont-Auvergne,
CNRS, LIMOS, Clermont-Ferrand, France
0000-0002-4459-511X

Charles Olivier-Anclin
be ys Pay
Université Clermont-Auvergne,
CNRS, LIMOS, Clermont-Ferrand, France
0000-0002-9365-3259

***Abstract*—Trick-taking games are traditional card games played all over the world. There are many such games, and most of them can be played online through dedicated applications, either for fun or for betting money. However, these games have an intrinsic drawback: each player plays their cards according to several secret constraints (unknown to the other players), and if a player does not respect these constraints, the other players will not realize it until much later in the game.**

In 2019, Bultel and Lafourcade proposed a cryptographic protocol for Spades in the random oracle model allowing peer-to-peer trick-taking games to be played securely without the possibility of cheating, even by playing a card that does not respect the secret constraints. However, this protocol requires a custom proof of shuffle with quadratic complexity in the number of cards, which makes the protocol inefficient in practice. We improve their work in several ways. First, we extend their model to cover a broader range of games, such as those implying a set of cards set aside during the deal (for instance French Tarot or Skat). Then, we propose a new efficient construction for Spades in the standard model with significantly improved efficiency. Finally, the secure protocol is extended to achieve French Tarot with comparable efficiency.

I. Introduction

Trick-taking Games: With the development of computers, many traditional games have been adapted into electronic versions. The emergence of the Internet has naturally made it possible to play these games online with opponents from all over the world. This is particularly the case for card games, and it is now possible to play Poker, Bridge, Blackjack, Ramis, Triomphe, Écarté, Euchre or Tarot with human opponents at any time and any place, thanks to the use of dedicated applications on computers or smartphones. While these applications allow users to play for fun, many of them offer to play for money. In this case, there are several security issues to consider, since an application that allows players to cheat would illegitimately make honest players lose money. For this reason, several works, initiated in the seminal paper of Goldwasser and Micali [7], have proposed cryptographic protocols allowing to play cards securely.

Trick-taking games are a family of card games that all have the same structure: the cards are dealt to the players, then the game is divided into several rounds; in each round, players take turns playing a card, and the player with the highest value card wins the round. However, players cannot play any card from their hand and must follow several rules.

Unlike other card games, trick-taking games allow players to cheat without it being immediately detectable: since the players' cards are hidden, it is not possible to know if a player respects the rules at the time it plays its card. The cheating is detected later in the game, when the cheater plays a card it is not supposed to have. Since such cheating is possible with a physical deck of cards, the classical cryptographic card game protocols do not prevent it. In [3], Bultel and Lafourcade introduce the secure trick-taking game protocols, which allows to detect when a player does not respect the rules of the game, without learning anything from its cards. Unfortunately, their security model cannot be applied to games in which not all cards are used by the players. This excludes some famous games, such as the French Tarot or Skat, considered as the national card game of Germany, among others.

Furthermore, the card distribution mechanism of the protocol in [3] suffers from two drawbacks inherent to its design. The first issue is that this approach is highly dependent on the random oracle model, the second is that the shuffle proof proposed in [3] is not efficient since its complexity is in $\mathcal{O}(n^2)$ in the number of cards.

Contributions: In [2], we first extend the security model from [3] to cover the French Tarot. French Tarot being the most complex of the games with cards set aside, it is easy to simplify our model to adapt it to other games having this property. Then, we propose two new secure trick-taking protocols based on a common idea (as in [3], for the sake of clarity, we base one of our protocols on Spades, but it can be adapted to any game having the same structure, the other is based on Tarot for similar reasons). The shuffle is done on a partially homomorphic encryption scheme, and there are

many efficient zero-knowledge proofs to prove the correctness of such a shuffle in the literature with linear complexity in the number of ciphertexts [1], [6], [9]. This allows us to instantiate our protocols much more efficiently than in [3], and to propose practical yet secure trick-taking protocols.

II. Trick-Taking Games and the Particularity of French Tarot

Our approach of trick-taking games could be applied to all aforementioned games and with any number of cards or players. To formalise it, we have chosen one of them: Spades. This allows us to keep notations simple and intelligible. Due to space limitation, we let the reader refer to [5], [10] for the rules of Spades and Tarot.

One particular case has never been addressed: the case where a set of cards is set aside during the deal, such as the dog (*chien*) in French Tarot. The dealing of this game generates another hand: While played with 4 players, 6 cards are put aside in a fifth hand until the bets are over. Once the cards are dealt, the *bids* start. The *taker* (the player that bets the highest) then plays against the 3 other players and needs to obtain a certain amount of points in its tricks to win. A player that does not bid *passes*. If all players pass, new cards are dealt.

III. Cryptographic Tools

Our trick-taking protocol relies on the Decisional Diffie-Hellman hypothesis (DDH) stating that for a group $\mathbb{G}$, given $(g, g^a, g^b, g^z) \in \mathbb{G}^4$, there exists no polynomial-time algorithm able to decide whether $z = a{\cdot}b$ or not. Our schemes use the ElGamal encryption specified through three algorithms: KeyGen, Enc and Dec. ElGamal is IND-CPA secure (indistinguishable under chosen plaintext attack) under the DDH hypothesis [11], moreover it is *randomizable*, which means that there exists an algorithm Rand that changes a ciphertext c into a new ciphertext c' of the same plaintext. This is put in use for the shuffle of the cards. Our construction also use Non-Interactive Zero-Knowledge Proofs of Knowledge (NIZKP) [8]. Let $\mathcal{R}$ be a binary relation and s, w two elements verifying $(s, w) \in \mathcal{R}$. A (NIZKP) is a cryptographic primitive allowing a prover knowing a witness w to show that w and s verify the relation $\mathcal{R}$ leaking no information on w.

IV. Models for Trick-Taking Game Revisited

Formal Definitions of Trick-Taking Scheme and Protocol

Trick-taking schemes and protocols were formalised in [3], but their definitions miss the French Tarot. we have extended them in [2], to cover this additional game while staying consistent with the existing. We introduce a new definition covering both the existing and our work. To cover the dog in French Tarot, we also add up an algorithm named $\mathsf{MakeDog}$.

Trick-Taking Game Scheme: First, we define a trick-taking scheme, which contain the algorithms for a play.

Definition 1: A *trick-taking scheme* (fully defined in [2]) between m participants encompass the following algorithms:

Init: Setups global parameters.

KeyGen: Returns a key pair for the players.

$\mathsf{DeckGen}$: It is a protocol between the players returning an encrypted and shuffled deck of cards.

$\mathsf{GetHand}$: It returns the hand of the player executing it.

Play: It allows to produce the values needed to play a card.

Verif: It checks the values of the play of a player.

$\mathsf{GetSuit}$: It returns the leading suit for the current turn.

An additional algorithm can be added to trick-taking schemes to support a dog:

$\mathsf{MakeDog}$: This is a protocol between the players outputting an updated deck of cards based on the previously shuffled one and a designated player.

Trick-taking Protocol: Fully defined in [2], it instantiates the order of execution of a trick-taking protocol and the communication it requires. These are similar to the real ones.

Security Properties

We now recall the security model of trick-taking protocols introduced in [3] and extended in [2]. The model proposed in [3] being too specific to the design of the related protocol, these properties were adjusted to make them more generic. Security is defined by five properties:

Theft and cheating resistance: A protocol is *theft-resistant* when a player cannot play a card that is not in its hand. A protocol is *cheating-resistant* when a player cannot play a card that does not follow the rules of the game

Unpredictability: It ensures that the cards are dealt at random even when players try to influence the shuffle.

Hand-privacy: It ensures that the players do not know the hand of the other players at the beginning of the game.

Game-privacy: A protocol is *game-private* when at each step of the game phase, the players learn nothing else than the previously played cards. This property is defined by a real/simulated experiment.

Particularity of Dog's Security: One would expect any set of cards set aside to behave as one of the player's hands: it should not be possible to steal (covered by theft resistance), to predict (unpredictability), to influence (theft-resistance) nor learn the cards in the dog (hand and game privacy) at the end of the shuffle. Moreover, the rules sometimes disallow to place some cards in the dog during the $\mathsf{MakeDog}$ algorithm. The latter is ensured through a property that we call *Dog security*.

Despite fitting the model in terms of required properties, games with dogs do not allow us to rely completely on what exists and we had to produce an *ad hoc* model to prove its security. A less *ad hoc* model is still an open problem.

V. Our Spades protocol

Our new Spades protocol is based on the randomisation of ElGamal. We highlight the main steps of the protocol of [2], assuming a deck D contains 52 cards, and each of the 4 players' hands 13 cards.

Setup: Each player P_i has to produce keys $(\mathsf{sk}_i = \mathsf{sk}_i, \mathsf{pk}_i = (\mathsf{ek}_i, \mathsf{ZK}\{\mathsf{sk}\colon \mathsf{ek} = g^{\mathsf{sk}}\}))$, initialise a canonical deck $D = (\mathsf{id}_1, \ldots, \mathsf{id}_{52})$ (canonical deck means a deck with a predefined order common to all participants) and to compute

a common product key $\mathsf{pk} = \prod_{i=1}^{4} \mathsf{ek}_i$. The spirit of our work is to ensure to all players that their fellows behave correctly at any step of the game. Hence, we require a ZKP to prove that they generated their keys correctly.

Shuffle and Distribution: The first player generates an *ad hoc* ElGamal ciphertext $(g, \mathsf{pk} \cdot \mathsf{id}_j)$ of all cards with the common key pk. All players sequentially apply a random permutation to the ciphertexts $(c_j)_{1 \leq j \leq 52}$, then randomise them using the Rand algorithm of the ElGamal encryption. For each of these steps a ZKP is required.

Cards are now shuffled and distributed in between the players. For $i \in \{1, 2, 3, 4\}$, player i receives the ciphertexts of indices in $\{13 \cdot (i-1)+1, 13 \cdot i\}$. Before being able to recover their hands, a second phase is necessary. Each of the players P_i has to produce 39 values $\theta_{(i,j)}$ for the 39 ciphertexts it has not been attributed (*i.e.,* $j \in [\![1, 52]\!] \backslash [\![13 \cdot (i-1)+1, 13 \cdot i]\!]$). With these values, participants can now recover their cards. Once again, a ZKP is required for the outputted values. Once they all received the $\theta_{(i,j)}$ from the 3 other users alongside a required proof, players check this latest one and then parse each of the ciphertexts $c = (x, y)$ of their cards. They divide the y values by a product of the three $\theta_{(i,j)}$ associated to the card. Hence, retrieving a ciphertext for their cards encrypted only with their keys. Each entity obtains its cards by decrypting its values, the cards remain oblivious to the other players as they are still encrypted with the player's key.

Playing a Card: To play the card id_j, player P_i has to prove that the card id_j it picks is in his/her hand, this by producing a proof of decryption proving that $\mathsf{id}_j = \mathsf{Dec}_{\mathsf{sk}_i}(c_{j'})$ for one $j' \in \{13 \cdot (i-1)+1, 13 \cdot i\}$. If at the time of playing id_j is not of the leading suit and not the last card of his/her hand, it has to compute a second proof showing that none of his/her cards corresponds to the leading suit.

This proof ensures that the encrypted cards c_j of the players are not of the leading suit, which proves that the player follows the rules of Spades. This proof must be verified by all the other players to prevent double play a card.

Security: This Spades protocol relies on the unpredictability of the randomness introduced by the players, security of the ZKP and the DDH hypothesis. Each of the five properties have been proven based on these hypothesis in [2].

Theorem 1: Given a secure proof of knowledge our protocol is theft-resistant, cheating-resistant, hand-private, unpredictable, and game-private under the DDH assumption.

VI. Our French Tarot's Protocol

A protocol that contains a dog can also be achieved through a modified version of the shuffle, an instantiation of a Tarot protocol is presented in [2]. Adapted from our previously presented Spades scheme of Section V, we need to address the $\mathsf{MakeDog}$ algorithm based on the rules of this game. On a high level, the dog remains encrypted while the hands are recovered by the players. Once all bets have been done and players agreed on the taker, remaining cards are unciphered and the latter proves that it does not place unauthorised cards in the dog. Then, the rest of the game remains unchanged. We advise the reader to refer to [2] for an in-depth description.

VII. Efficiency Analysis

Users of online card games want efficient and reliable tools. Waiting in between plays would inevitably lead to a degraded game experience. We compare our protocol to the only other secure Spades protocol [3]. We have implemented our protocol in Rust. Most of the computations are due to the numerous zero-knowledge proofs. A first improvement is that we can implement two proof designs for the shuffle.

(1): Proof *1-out-of-n* [4] based on Schnorr's proof, complexity $\mathcal{O}(n^2)$.

(2): Groth proof of shuffle from [9], complexity $\mathcal{O}(n)$.

Since the second design proves shuffle of homomorphically encrypted elements, it cannot be used in [3]. On the whole, a full execution of $\mathsf{DeckGen}$ can be performed with Design (1) in about 20 seconds of processor time for all 4 players (equivalent to an instantiation of [3]), against 3 seconds with Design (2) at equivalent security level. It takes 178 milliseconds (ms) to recover each hand. A play takes about 270 ms of time and its verifications only 185 ms. This analysis shows that this protocol is the most efficient secure Spades protocol existing to date. Our Tarot protocol has a computation overhead of the same magnitude. Furthermore, these timings being of the same order of magnitude than 1 RTT (Round-Trip Time), the overhead brought by securing the protocol seems acceptable.

References

[1] Stephanie Bayer and Jens Groth. Efficient zero-knowledge argument for correctness of a shuffle. In *Advances in Cryptology – EUROCRYPT*. Springer, 2012.

[2] Rohann Bella, Xavier Bultel, Céline Chevalier, Pascal Lafourcade, and Charles Olivier-Anclin. Practical construction for secure trick-taking games even with cards set aside. In *Financial Cryptography and Data Security*. LNCS. Springer, 2023.

[3] Xavier Bultel and Pascal Lafourcade. Secure trick-taking game protocols - how to play online spades with cheaters. In Ian Goldberg and Tyler Moore, editors, *FC 2019*. Springer, Heidelberg, 2019. https://eprint.iacr.org/2019/375.

[4] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Advances in Cryptology — CRYPTO*. Springer, 1994.

[5] Fédération française de Tarot. Règlement officiel du jeu de tarot, 2012. http://www.fftarot.fr/assets/documents/R-RO201206.pdf.

[6] Jun Furukawa and Kazue Sako. An efficient scheme for proving a shuffle. In *Advances in Cryptology — CRYPTO*. Springer, 2001.

[7] Shafi Goldwasser and Silvio Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, STOC. ACM, 1982.

[8] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on computing*, 1989.

[9] Jens Groth. A verifiable secret shuffle of homomorphic encryptions. In *Journal of Cryptology*. Springer, 2010.

[10] Barry Rigal. *Card games for dummies*. John Wiley & Sons, 2022.

[11] Yiannis Tsiounis and Moti Yung. On the security of elgamal based encryption. In *PKC*. Springer, 1998.