(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
31 October 2002 (31.10.2002)

PCT

(10) International Publication Number
WO 02/085475 A2

(51) International Patent Classification[7]: A63F 3/00

(21) International Application Number: PCT/IB02/01868

(22) International Filing Date: 27 March 2002 (27.03.2002)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
01/04954 11 April 2001 (11.04.2001) FR

(71) Applicant *(for all designated States except US)*: **FXTOP** [FR/FR]; 13 rue Lantiez, F-75017 Paris (FR).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: **PELE, Laurent** [FR/FR]; 13, rue Lantiez, F-75017 Paris (FR).

(81) Designated States *(national)*: AU, CA, GB, IL, JP, NZ, US, ZA.

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii)) for all designations*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii)) for all designations*

— *of inventorship (Rule 4.17(iv)) for US only*

Published:

— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 02/085475 A2

(54) Title: SELECTION OF CARDS FOR A LONG-DISTANCE GAME

(57) Abstract: This invention consists of a process of blind selection of X cards from a pack of K cards represented by numbers C_1, C_2 CK, where the two players are separated by a long distance without any third party. 2 devices "Player A" and "Player B" share the same prime number N. "Player A" selects number A1 randomly so that A1 and N-1 are relatively prime. "Player A" computes Vi=Ci^A1 modulo N for every i number between 1 and K. ; shuffles the cards and sends them to "Player B" "Player B" selects numbers B1 and B2 randomly so that they are both relatively prime with N-1 "Player B" selects X different cards from the pack of K cards received and computes $Jb_i=V_i$^B1 modulo N for each of them. Then "Player B" sends that to "Player A" "Player A" computes A1prime so that A1prime*A1=1 modulo N-1 "Player A" computes Vi=JB_i^A1prime modulo N for the X cards and sends that to "Player A" "Player B" computes B1prime so that B1prime*B1=1 modulo N-1 "Player B" computes Vi=JB_i^B1prime modulo N for the X cards and sees the decrypted values of cards (the Vi numbers correspond to some of the C_j numbers). He can choose the cards he wants to keep and the ones he wants to put back in the pack.

SELECTION OF CARDS FOR A LONG-DISTANCE GAME

Technical area :

When most people play cards, they usually shuffle the pack of cards. A player picks a card in the pack. He doesn't know which card he will pick in the pack but he knows that the card he will pick is not already in his hand or in the hands of the other players. This is true even if the player doesn't know which cards the other players have in their hand. When a player puts back a card in the pack, this card could be given to another player later.

Some card games are not open to any audience. For example, poker, even without betting money, is forbidden in some countries or areas, except when it is played privately between members of the same family above a certain age.

In some circumstances, the members of that family are allowed to play poker together and bet money. Unfortunately, members of the same family don't often live at the same location and have to travel a long distance to see each other. It is difficult for isolated members of a family to find a game partner that belongs to their family, who likes playing that game and who doesn't live too far away.

In some cases, players decide to connect to the Internet to find other players that like playing the game (for example cards, dice, dominoes etc.) they like.

It isn't possible yet to play long distance cards or dominoes because of hidden technical properties of the pack. One solution seems to be to use an intermediary third party that shuffles the cards but that third party may be corrupted (for example one player could pay him to know which cards belong to the other players or to have a good hand). The intermediary third party may also make mistakes in card shuffling and may have insufficient security so that the cards of the other players or of the pack could be revealed to one or several players. In poker, players should only rely on themselves and should take care not to disclose their game to anybody else.

Another problem of using an intermediary third party to shuffle the cards is that this third party often requires money to give that service and that could be expensive. It is also sometimes difficult to choose game partners with that intermediary (for example people from the same family). In some places, that intermediary third party may also be forbidden by law.

In some money games, some people designated to shuffle the cards earned a lot of money with particular card shuffling and were in collusion with one of the players. So the way of shuffling cards is also important otherwise some of the players may cheat, and cards should be shuffled by several independent people. A good solution to avoid that kind of fraud is when the pack is shuffled by all the players so none of the players are able to say that the cards have been shuffled improperly.

To conclude this part, either the intermediary third party is motivated by earning money or it isn't and doesn't have to take the appropriate security measures.

The consequence is that the players should think they can only rely on themselves and their own security measures.

The present invention explains a technical solution to select, shuffle and exchange cards (or dominoes or dice) in a long distance game without any intermediary third party.

The main technical problem is to solve the following : how can a player pick a card (or a domino) from the pack in a long distance game without knowing which card it is (otherwise he would choose it and cheat). Other players should also not know which card has been picked.

Another technical problem, is how a player can put back a card in the pack so that another player may pick it later. The following rule must be respected so the long distance game is like a real one : the sum of the cards in each hand plus the cards in the pack equal all the cards at any time in the game. And all the cards in the game must appear only once.

The present invention describes a cryptographic process to select and shuffle cards : each card is represented by a different number. Each player picks a card in the pack without knowing the cards in the pack or the other players cards but he cannot choose a card that already belongs to him or to another player.

Cards are shuffled at every turn so that none of the player can guess which card has been selected by another player.

The principle of the invention is the following for a 2 player-game :

Player A encrypts the pack of cards, Player B chooses cards from that pack, encrypts the selected cards and send them to Player A, Player A decrypts those cards with his key (Player A can't know the cards selected by Player B as they are still encrypted with Player B's key). Then Player B decrypts the cards with his key so that he can see the real value of the cards he has selected.

This process uses the mathematical properties of power function on large numbers modulo P where P is a large prime number (assuming also that P-1 has a large prime factor).

That power function mod P is commutative so [(X power A mod P) power B mod P is equal to (X power B mod P) power A mod P.

The security of the process is based on the difficulty to solve the discrete logarithm mod P when P is a large prime number (and P-1 has a large prime factor).

Previous technique :

None of the cryptographic methods known so far (like public key cryptography or symmetric cryptography) have been able to solve the drawing of a card from the other ones in a pack, without revealing it.

The problem of putting back a card in the pack (so that players can pick it in later turns) is also a problem that hasn't had a solution so far.

Description of the invention :

The process of selection of X cards for a long distance card game between 2 or more players selecting cards from the pack where none of the players know the content of the pack and none of the players share the same card.

With the use of :

- an integer number N strictly superior to 1
- a pack of K cards represented by the numbers C_1, C_2 to C_K where K is an integer strictly superior to 1 and strictly inferior to N. Numbers C_1, C_2 to C_K have values between 2 and N-1
- a number PhiN equal to the Euler totient function of N (eg PhiN = N-1 when N is a prime number)
- 2 devices called « Player A » and « Player B » with appropriate computation, data storage and data transmission capabilites

Step 1

T_A_1, T_A_2 to T_A_K represent the pack

« Player A » stores the respective values C_1, C_2 to C_K in T_A_1, T_A_2 to T_A_K

« Player A » shuffles T_A_1, T_A_2 to T_A_K (for example by doing several swaps between T_Ai and T_Aj, where i and j are numbers randomly selected between 1 and K)

« Player A » selects number A1 so that A1 and PhiN are relatively prime.

A1 is kept secretly by « Player A »

« Player A » computes Vi = T_Ai ^A1 modulo N (sign ^ represents the power function) for every i number between 1 and K, and stores Vi in T_Aj.

« Player A » shuffles the now encrypted T_A_1, T_A_2 to T_A_K

« Player A » sends the group of T_A_1, T_A_2 to T_A_K to « Player B »

Step 2

« Player B » receives a group of K values and stores them in T_B_1, T_B_2 to T_B_K

« Player B » shuffles T_B_1, T_B_2 to T_B_K

« Player B » select X different cards randomly from T_B_1, T_B_2 to T_B_K and stores their values in JB_B_1, JB_B_2 to JB_B_K. Non selected cards are then stored in T_B_1, T_B_2 to T_B_{K-X}

« Player B » selects 2 numbers B1 and B2 randomly so that B1 and PhiN are relatively prime and B2 and PhiN are relatively prime.

B1 and B2 are kept secretly by « Player B »

« Player B » computes Vi = T_Bi ^B2 modulo N for every i number between 1 and K-X and

stores Vi in T_Bi

« Player B » shuffles T_B_1, T_B_2 to T_B_{K-X}

« Player B » computes Vi = JB_Bi ^B1 modulo N for every i number between 1 and X, and stores Vi in JB_Bi.

« Player B » shuffles JB_B_1, JB_B_2 to JB_B_X

« Player B » sends to « Player A » the first group of K-X values T_B_1, T_B_2 to T_B_{K-X} and the second group of X values JB_B_1, JB_B_2 to JB_B_X

Step 3

« Player A » receives the K-X values of the first group and stores them in T_A_1, T_A_2 to T_A_{K-X} then stores X values of the second group in JB_A_1, JB_A_2 to JB_A_X

« Player A » shuffles T_A_1, T_A_2 to T_A_{K-X}

« Player A » shuffles JB_A_1, JB_A_2 to JB_A_X

« Player A » computes A1prime so that A1*A1prime =1 modulo PhiN

« Player A » computes Vi = JB_A i ^A1prime modulo N for every i number between 1 and X, and stores Vi in JB_Ai

« Player A » selects 2 numbers A2 and A3 randomly so that A2 and PhiN are relatively prime and A3 and PhiN are relatively prime.

A1prime, A2 and A3 are kept secretly by « Player A »

« Player A » selects X different cards randomly from T_A_1, T_A_2 to T_A_{K-X} and stores their values in JA_A_1, JA_A_2 to JA_A_K. Non selected cards are then stored in T_A_1, T_A_2 to T_A_{K-2*X}

« Player A » shuffles JA_A_1, JA_A_2 to JA_A_X

« Player A » computes Vi = JA_A i ^(A1prime*A2) modulo N for every i number between 1 and X, and stores Vi in JA_A i

« Player A » shuffles JA_A_1, JA_A_2 to JA_A_X

« Player A » computes Vi = T_Ai ^(A1prime*A3) modulo N for every i number between 1 and K-2*X, and stores Vi in T_Ai

« Player A » shuffles T_A_1, T_A_2 to T_A_{K-2*X}

« Player A » sends to « Player B » the first group of K-2*X values T_A_1, T_A_2 to T_A_{K-2*X}, the second group of X values JB_A_1, JB_A_2 to JB_A_X and a third group of X values JA_A_1, JA_A_2 to JA_A_X

Step 4

« Player B » receives the K-2*X values of the first group and stores them in T_B_1, T_B_2 to T_B_{K-2*X} then stores X values of the second group in JB_B_1, JB_B_2 to JB_B_X

and finally stores X values of the third group in JA_B_1, JA_B_2 to JA_B_X

« Player B » shuffles T_B_1, T_B_2 to T_B_{K-2*X}

« Player B » shuffles JB_B_1, JB_B_2 to JB_B_X

« Player B » computes B1prime so that B1*B1prime =1 modulo PhiN

« Player B » computes Vi = JB_Bi ^B1prime modulo N for every i number between 1 and X, and stores Vi in JB_Bi

« Player B » sees at that point the decrypted values of his cards in JB_Bi [they correspond to a selection of values from C_1, C_2 to C_K because C_i^(A1*B1*A1prime*B1prime) modulo N is equal to C_i]

« Player B » computes B2prime so that B2*B2prime =1 modulo PhiN

B1prime and B2prime are kept secretly by « Player B »

« Player B » computes Vi = JA_Bi ^A1prime modulo N for every i number between 1 and X, and stores Vi in JA_Bi.

« Player B » shuffles X memories JA_B_1, JA_B_2 to JA_B_X.

« Player B » send to « Player A » the group of X values JA_B_1, JA_B_2 to JA_B_X.

Step 5

« Player A » receives the X values of the group and stores them in JA_A_1, JA_A_2 to JA_A_X

« Player A » computes A2prime so that A2*A2prime =1 modulo PhiN

A2prime is kept secretly by « Player A »

« Player A » computes Vi = JA_Ai ^A2prime modulo N for every i number between 1 and X, and stores Vi in JA_Ai.

« Player A » sees at that point the decrypted values of his cards in JA_Ai [they correspond to a selection of values from C_1, C_2 to C_K because C_i^(A1*B1*A1prime*A2*B1prime*A2prime) modulo N is equal to C_i]

An additional feature of this process can be seen when a player puts a card back in the pack. For example, in some card games like poker, a player can choose a few cards (YB for example) he doesn't want and pick other ones in the pack. A constraint is that the cards he picks should not be any of the ones he has put back in the pack !

This feature can be achieved by adding the following operations to steps 4 and 5 of the previous process :

At the end of step 4 :

« Player B » shuffles JB_B_1, JB_B_2 to JB_B_X.

« Player B » chooses YB cards (YB is an integer between 0 and X) from the X cards of JB_B_1, JB_B_2 to JB_B_X he wants to replace and stores them in RB_B_1, RB_B_2 to RB_B_{YB}.

« Player B » selects 2 numbers B3 and B4 randomly so that B3 and PhiN are relatively prime and B4 and PhiN are relatively prime.

B3 and B4 are kept secretly by « Player B »

« Player B » computes Vi = RB_Bi ^B4 modulo N for every i number between 1 and YB, and stores Vi in RB_Bi.

« Player B » shuffles T_B_1, T_B_2 to T_B_{K-2X}.

« Player B » selects YB different cards randomly from T_B_1, T_B_2 to T_B_{K-2X} and stores their values in PB_B_1, PB_B_2 to PB_B_{YB}. Non selected cards are then stored in T_B_1, T_B_2 to $T\_B_{K-2*X-YB}$.

« Player B » computes Vi = T_Bi ^(B4*B2prime) modulo N for every i number between 1 and K-2*X-Y, and stores Vi in T_Bi.

« Player B » computes Vi = PB_Bi ^(B3*B2prime) modulo N for every i number between 1 and YB, and stores Vi in PB_Bi.

« Player B » shuffles PB_B_1, PB_B_2 to PB_B_{YB}

« Player B » shuffles RB_B_1, RB_B_2 to RB_B_{YB}

« Player B » send to « Player A » the second group of K-2*X-YB values T_B_1, T_B_2 to $T\_B_{K-2*X-YB}$, the third group of YB values PB_B_1, PB_B_2 to PB_B_{YB} and the fourth group of YB values RB_B_1, RB_B_2 to RB_B_{YB}

At the end of step 5 :

« Player A » receives the K-2*X-YB values of the second group and stores them in T_A_1, T_A_2 to $T\_A_{K-2*X-YB}$, then stores the YB values of the third group in PB_A_1, PB_A_2 to PB_A_{YB} and then stores the YB values of the fourth group in RB_A_1, RB_A_2 to RB_A_Y

« Player A » shuffles PB_A_1, PB_A_2 to PB_A_{YB}

« Player A » shuffles RB_A_1, RB_A_2 to RB_A_{YB}

« Player A » shuffles JA_A_1, JA_A_2 to JA_A_X

« Player A » shuffles T_A_1, T_A_2 to $T\_A_{K-2X-YB}$

« Player A » selects 2 numbers A4 and A5 randomly so that A4 and PhiN are relatively prime and A5 and PhiN are relatively prime.

A4 and A5 are kept secretly by « Player A »

« Player A » computes Vi = RB_Ai ^A5 modulo N for every i number between 1 and YB, and stores Vi in RB_Ai

« Player A » computes A3prime so that A3*A3prime =1 modulo PhiN

A3prime is kept secretly by « Player A »

« Player A » computes Vi = PB_Ai ^A3prime modulo N for every i number between 1 and YB, and stores Vi in PB_Ai (« Player B » will be able to see those cards when it's their turn)

« Player A » chooses the YA cards (YA is an integer between 0 and X) from JA_A_1, JA_A_2 to JA_A_X he wants to replace and stores their values in RA_A_1, RA_A_2 to RB_A_{YA}

« Player A » computes Vi = RA_Ai ^A5 modulo N for every i number between 1 and YA, and stores Vi in RA_Ai

« Player A » selects YA different cards randomly from T_A_1, T_A_2 to $T\_A_{K-2X-YB}$ and stores their values in PA_A_1, PA_A_2 to PA_A_{YA}. Non selected cards are then stored in T_A_1, T_A_2 to $T\_A_{K-2*X-YB-YA}$

« Player A » computes Vi = PA_Ai ^(A3prime*A4) modulo N for every i number between 1 and YA, and stores Vi in PA_Ai

« Player A » computes Vi = T_Ai ^(A3prime*A5) modulo N for every i number between 1 and K-2*X-YB-YA and stores Vi in T_Ai

« Player A » shuffles PA_A_1, PA_A_2 to PA_A_{YA}

« Player A » shuffles RA_A_1, RA_A_2 to RA_A_{YA}

« Player A » adds to the K-2*X-YB-YA values stored in T_A_1, T_A_2 to $T\_A_{K-2*X-YB-YA}$ the YB values stored in RB_A_1, RB_A_2 to RB_A_{YB}. After this combination, the pack has K-2*X-YA cards in T_A_1, T_A_2 to $T\_A_{K-2*X-YA}$

« Player A » shuffles T_A_1, T_A_2 to $T\_A_{K-2X-YA}$

After that operation, all the K-2*X-YA cards stored in the T_Ai are encrypted with B3*A5 and the game can continue with more turns by replacing cards back in the pack and selecting other ones

Indication of the way the invention can have an industrial application :

Computations and storage of data can be done by micro-computers or smart / chip cards. The transfer of data can be done between two computers that are connected together or over a computer network, (for example via a private or public communication infrastructure like the Internet).

The invention is particularly adapted for impartial and confidential long distance card selecting without any third party.

It is possible to extend the process to games with more than 2 players (each player should shuffle and encrypt the cards before the first player selects some, each player has a set of keys for each of the other players).

It is almost the same process for any game with a pack (like cards or dominoes), the process of card selection is independent from the game rules.

The process can be used for a long distance throw of a die. For example, assuming that the die has six faces, you can apply the above process where the pack of cards has 6 different numbers and one of the players chooses one. The security of this « long distance die throwing » is in the fact that each player shuffles (and encrypts) the numbers of the pack, so that nobody can assume there is a cheat in the shuffling.

The process can also be applied to blind tests - for validating laboratory tests or industrial tests, for example.

Software can be specifically designed to implement this process for the above and other applications.

5

CLAIMS

1) A process of selection of X cards for a long distance card game between 2 players selecting cards from the pack where none of the players know the content of the pack and none of the players share the same card.

With the use of :

- an integer number N strictly superior to 1
- a pack of K cards represented by the numbers C_1, C_2 to C_K where K is an integer strictly superior to 1 and strictly inferior to N. Numbers C_1, C_2 to C_K have values between 2 and N-1
- a number PhiN equal to the Euler totient function of N (eg PhiN = N-1 when N is a prime number)
- 2 devices called « Player A » and « Player B » with appropriate computation, data storage and data transmission capabilites

Step 1

T_A_1, T_A_2 to T_A_K represent the pack

« Player A » stores the respective values C_1, C_2 to C_K in T_A_1, T_A_2 to T_A_K

« Player A » shuffles T_A_1, T_A_2 to T_A_K (for example by doing several swaps between T_Ai and T_Aj, where i and j are numbers randomly selected between 1 and K)

« Player A » selects number A1 so that A1 and PhiN are relatively prime.

A1 is kept secretly by « Player A »

« Player A » computes Vi = T_Ai ^A1 modulo N (sign ^ represents the power function) for every i number between 1 and K, and stores Vi in T_Aj.

« Player A » shuffles the now encrypted T_A_1, T_A_2 to T_A_K

« Player A » sends the group of T_A_1, T_A_2 to T_A_K to « Player B »

Step 2

« Player B » receives a group of K values and stores them in T_B_1, T_B_2 to T_B_K

« Player B » shuffles T_B_1, T_B_2 to T_B_K

« Player B » select X different cards randomly from T_B_1, T_B_2 to T_B_K and stores their values in JB_B_1, JB_B_2 to JB_B_K. Non selected cards are then stored in T_B_1, T_B_2 to T_B_{K-X}

« Player B » selects 2 numbers B1 and B2 randomly so that B1 and PhiN are relatively prime and B2 and PhiN are relatively prime.

B1 and B2 are kept secretly by « Player B »

« Player B » computes Vi = T_Bi ^B2 modulo N for every i number between 1 and K-X and stores Vi in T_Bi

« Player B » shuffles T_B_1, T_B_2 to T_B_{K-X}

« Player B » computes Vi = JB_Bi ^B1 modulo N for every i number between 1 and X, and stores Vi in JB_Bi.

« Player B » shuffles JB_B_1, JB_B_2 to JB_B_X

« Player B » sends to « Player A » the first group of K-X values T_B_1, T_B_2 to T_B_{K-X} and the second group of X values JB_B_1, JB_B_2 to JB_B_X

Step 3

« Player A » receives the K-X values of the first group and stores them in T_A_1, T_A_2 to T_A_{K-X} then stores X values of the second group in JB_A_1, JB_A_2 to JB_A_X

« Player A » shuffles T_A_1, T_A_2 to T_A_{K-X}

« Player A » shuffles JB_A_1, JB_A_2 to JB_A_X

« Player A » computes A1prime so that A1*A1prime =1 modulo PhiN

« Player A » computes Vi = JB_A i ^A1prime modulo N for every i number between 1 and X, and stores Vi in JB_Ai

« Player A » selects 2 numbers A2 and A3 randomly so that A2 and PhiN are relatively prime and A3 and PhiN are relatively prime.

A1prime, A2 and A3 are kept secretly by « Player A »

« Player A » selects X different cards randomly from T_A_1, T_A_2 to T_A_{K-X} and stores their values in JA_A_1, JA_A_2 to JA_A_K. Non selected cards are then stored in T_A_1, T_A_2 to T_A_{K-2*X}

« Player A » shuffles JA_A_1, JA_A_2 to JA_A_X

« Player A » computes Vi = JA_A i ^(A1prime*A2) modulo N for every i number between 1 and X, and stores Vi in JA_A i

« Player A » shuffles JA_A_1, JA_A_2 to JA_A_X

« Player A » computes Vi = T_Ai ^(A1prime*A3) modulo N for every i number between 1 and K-2*X, and stores Vi in T_Ai

« Player A » shuffles T_A_1, T_A_2 to T_A_{K-2*X}

« Player A » sends to « Player B » the first group of K-2*X values T_A_1, T_A_2 to T_A_{K-2*X}, the second group of X values JB_A_1, JB_A_2 to JB_A_X and a third group of X values JA_A_1, JA_A_2 to JA_A_X

Step 4

« Player B » receives the K-2*X values of the first group and stores them in T_B_1, T_B_2 to T_B_{K-2*X} then stores X values of the second group in JB_B_1, JB_B_2 to JB_B_X

and finally stores X values of the third group in JA_B_1, JA_B_2 to JA_B_X

« Player B » shuffles T_B_1, T_B_2 to T_B_{K-2*X}

« Player B » shuffles JB_B_1, JB_B_2 to JB_B_X

« Player B » computes B1prime so that B1*B1prime =1 modulo PhiN

« Player B » computes Vi = JB_Bi ^B1prime modulo N for every i number between 1 and X, and stores Vi in JB_Bi

« Player B » sees at that point the decrypted values of his cards in JB_Bi [they correspond to a selection of values from C_1, C_2 to C_K because C_i^(A1*B1*A1prime*B1prime) modulo N is equal to C_i]

« Player B » computes B2prime so that B2*B2prime =1 modulo PhiN

B1prime and B2prime are kept secretly by « Player B »

« Player B » computes Vi = JA_Bi ^A1prime modulo N for every i number between 1 and X, and stores Vi in JA_Bi.

« Player B » shuffles X memories JA_B_1, JA_B_2 to JA_B_X.

« Player B » send to « Player A » the group of X values JA_B_1, JA_B_2 to JA_B_X.

Step 5

« Player A » receives the X values of the group and stores them in JA_A_1, JA_A_2 to JA_A_X

« Player A » computes A2prime so that A2*A2prime =1 modulo PhiN

A2prime is kept secretly by « Player A »

« Player A » computes Vi = JA_Ai ^A2prime modulo N for every i number between 1 and X, and stores Vi in JA_Ai.

« Player A » sees at that point the decrypted values of his cards in JA_Ai

2) A process of selection of X cards for a long distance card game between 2 players as in claim 1 when a player can put a card back in the pack.

This process is achieved by adding the following operations to steps 4 and 5 to the process of claim 1:

At the end of step 4 :

« Player B » shuffles JB_B_1, JB_B_2 to JB_B_X.

« Player B » chooses YB cards (YB is an integer between 0 and X) from the X cards of JB_B_1, JB_B_2 to JB_B_X he wants to replace and stores them in RB_B_1, RB_B_2 to RB_B_{YB}.

« Player B » selects 2 numbers B3 and B4 randomly so that B3 and PhiN are relatively prime and B4 and PhiN are relatively prime.

B3 and B4 are kept secretly by « Player B »

« Player B » computes Vi = RB_Bi ^B4 modulo N for every i number between 1 and YB, and stores Vi in RB_Bi.

« Player B » shuffles T_B_1, T_B_2 to T_B_{K-2X}.

« Player B » selects YB different cards randomly from T_B_1, T_B_2 to T_B_{K-2X} and stores their values in PB_B_1, PB_B_2 to PB_B_{YB}. Non selected cards are then stored in T_B_1, T_B_2 to $T\_B_{K-2*X-YB}$.

« Player B » computes Vi = T_Bi ^(B4*B2prime) modulo N for every i number between 1 and K-2*X-Y, and stores Vi in T_Bi.

« Player B » computes Vi = PB_Bi ^(B3*B2prime) modulo N for every i number between 1 and YB, and stores Vi in PB_Bi.

« Player B » shuffles PB_B_1, PB_B_2 to PB_B_{YB}

« Player B » shuffles RB_B_1, RB_B_2 to RB_B_{YB}

« Player B » send to « Player A » the second group of K-2*X-YB values T_B_1, T_B_2 to $T\_B_{K-2*X-YB}$, the third group of YB values PB_B_1, PB_B_2 to PB_B_{YB} and the fourth group of YB values RB_B_1, RB_B_2 to RB_B_{YB}

At the end of step 5 :

« Player A » receives the K-2*X-YB values of the second group and stores them in T_A_1, T_A_2 to $T\_A_{K-2*X-YB}$, then stores the YB values of the third group in PB_A_1, PB_A_2 to PB_A_{YB} and then stores the YB values of the fourth group in RB_A_1, RB_A_2 to RB_A_Y

« Player A » shuffles PB_A_1, PB_A_2 to PB_A_{YB}

« Player A » shuffles RB_A_1, RB_A_2 to RB_A_{YB}

« Player A » shuffles JA_A_1, JA_A_2 to JA_A_X

« Player A » shuffles T_A_1, T_A_2 to $T\_A_{K-2X-YB}$

« Player A » selects 2 numbers A4 and A5 randomly so that A4 and PhiN are relatively prime and A5 and PhiN are relatively prime.

A4 and A5 are kept secretly by « Player A »

« Player A » computes Vi = RB_Ai ^A5 modulo N for every i number between 1 and YB, and stores Vi in RB_Ai

« Player A » computes A3prime so that A3*A3prime =1 modulo PhiN

A3prime is kept secretly by « Player A »

« Player A » computes Vi = PB_Ai ^A3prime modulo N for every i number between 1 and YB, and stores Vi in PB_Ai (« Player B » will be able to see those cards when it's their turn)

« Player A » chooses the YA cards (YA is an integer between 0 and X) from JA_A_1, JA_A_2 to JA_A_X he wants to replace and stores their values in RA_A_1, RA_A_2 to RB_A_{YA}

« Player A » computes Vi = RA_Ai ^A5 modulo N for every i number between 1 and YA, and stores Vi in RA_Ai

« Player A » selects YA different cards randomly from T_A_1, T_A_2 to $T\_A_{K-2X-YB}$ and stores

their values in PA_A_1, PA_A_2 to PA_A_{YA}. Non selected cards are then stored in T_A_1, T_A_2 to T_$A_{K-2*X-YB-YA}$

« Player A » computes Vi = PA_Ai ^(A3prime*A4) modulo N for every i number between 1 and YA, and stores Vi in PA_Ai

« Player A » computes Vi = T_Ai ^(A3prime*A5) modulo N for every i number between 1 and K-2*X-YB-YA and stores Vi in T_Ai

« Player A » shuffles PA_A_1, PA_A_2 to PA_A_{YA}

« Player A » shuffles RA_A_1, RA_A_2 to RA_A_{YA}

« Player A » adds to the K-2*X-YB-YA values stored in T_A_1, T_A_2 to T_$A_{K-2*X-YB-YA}$ the YB values stored in RB_A_1, RB_A_2 to RB_A_{YB}. After this combination, the pack has K-2*X-YA cards in T_A_1, T_A_2 to T_$A_{K-2*X-YA}$

« Player A » shuffles T_A_1, T_A_2 to T_$A_{K-2X-YA}$

3) process according to claims 1 or 2 where there are more than 2 players

4) process according to claims 1 , 2 or 3 where one of the devices is a smart card

5) process according to claims 1, 2 , 3 or 4 where one of the devices is a computer