Secure Large-Scale Bingo *

Antoni Martínez-Ballesté, Francesc Sebé and Josep Domingo-Ferrer
Universitat Rovira i Virgili,
Dept. of Computer Engineering and Maths,
Av. Països Catalans 26, E-43007 Tarragona, Catalonia,
e-mail {anmartin, fsebe, jdomingo}@etse.urv.es

Abstract

When designing a secure Internet version for keno or bingo for a large group of players, a multicast communication model is likely to be used for source-to-players communication. For a large group of players, this can potentially overwhelm the source, a problem which is known as implosion. This paper presents a scheme for large-scale bingo which avoids implosion in the many-to-one communication between the players and the hall. The resulting scheme aggregates information using a probabilistic public-key additive privacy homomorphism and guarantees the security of communications.

Keywords: *Virtual gaming, Multicast, Many-to-one communications.*

1 Introduction

With the widespread use of Internet, plenty of virtual casinos offer electronic versions of classic games such as roulette, blackjack, etc. Some of these are large group games, like keno or bingo. When designing a secure Internet version for keno or bingo for a large group of players, a multicast communication model is likely to be used for source-to-players (one-to-many) communication.

A multicast network consists of one source sending a single stream of data to a set of multicast routers. The data stream floods the multicast tree in order to efficiently reach every customer subscribed to the current session. Before a multicast transmission begins, the session is advertised to potential customers. Replies are made by means of protocols such as IGMP, which provide the source with information about who is interested in the upcoming session. With this information, an optimal multicast tree is built.

Using multicast transmission for one-to-many communication results in a dramatical reduction of the bandwidth and memory used in intermediate routers.

1.1 The implosion problem

For a large group of players, reverse many-to-one communication can potentially overwhelm the source, a problem which is known as implosion. In addition to requiring solutions to implosion, some many-to-one applications also have security requirements.

1.2 Contribution and plan of this paper

Moving the bingo game from physical casinos to virtual Internet halls requires many-to-one communication between the players and the virtual hall. If the game is played on a large scale, special care must be devoted to avoiding implosion of communications on the side of the virtual hall. This paper presents a scheme for large-scale bingo which avoids implosion in the many-to-one communication between the players and the hall. The resulting scheme aggregates information using a probabilistic public-key additive privacy homomorphism and guarantees the security of communications.

Section 2 describes a scheme for sending bits from many receivers to the source. This scheme is used in Section 3, where a secure large-scale Internet bingo is described. In Section 4 the security of the scheme is proven.

*This work has been partly supported by the Spanish Ministry of Science and Technology and the European FEDER Fund under project TIC2001-0633-C03-01 "STREAMOBILE".

Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04)
0-7695-2108-8/04 $ 20.00 © 2004 IEEE

IEEE COMPUTER SOCIETY

2 Many-to-one transmission in multicast networks

In this section, a scheme for sending bits from many receivers to the source via the multicast tree is described. In a multicast scenario there are three parties, which are also involved in the protocol:

- The source, which multicasts some information to all players in the session.
- The intermediate routers, which aggregate the information and send it to their parent routers.
- The players, who receive some information via their multicast connection and can send some information to the source.

In this tool, a public-key probabilistic additive privacy homomorphism (PH) is used. A canditate PH for the application described here is [Okam98]. Note that, since a probabilistic cryptosystem is being used, the same cleartext message can result in different encrypted messages.

2.1 The basic construction

The construction consists of a set-up protocol to be run before any transmissions are started, and a transmission protocol to be run for each symbol transmission.

Protocol 1 (Set-up)

1. *The source chooses parameters l, u, where l will be used below and u is the number of users.*
2. *The source generates $2u$ intervals as follows:*

$$\mathbf{I}_1 = [1, 2^l - 1]$$

$$\mathbf{I}_j = [I_j^{min}, I_j^{max}] = [(2^j - 2)(2^l - 1) + 2^{j-1} - 1,$$
$$(2^j - 1)(2^l - 1) + 2^{j-1} - 1]$$

   for $j = 2$ to $2u$.
3. *The source generates u keys k_i, for $i = 1$ to u, corresponding to a block cipher (*e.g. *AES).*
4. *The source generates a key pair for a probabilistic additive public-key privacy homomorphism such that its cleartext space is $CT = \{0, 1, 2, \cdots, p-1\}$ where p should be larger than $2I_{2u}^{max} + 1$. After some manipulation, it can be checked that the lower bound on p is*

$$p > (2^{2u} - 1)(2^{l+1} - 1) \qquad (1)$$

   *(*E.g.*, for $u = 500$ users, p should have $O(10^3)$ bits.)*
5. *The source multicasts the public key PK of the PH and I_{2i-1}^{min}, I_{2i-1}^{max}, I_{2i}^{min}, I_{2i}^{max} for $i = 1$ to u. In addition the source secretly sends k_i to each user U_i, who should keep this key confidential (storing it in a tamper-resistant device such as a smart card would seem appropriate).*

After set-up, the normal operation of the scheme consists of many-to-one transmissions of binary symbols. In order to collect a binary symbol from each user, the following four-step protocol is used:

Protocol 2 (Many-to-one bit transmission)

1. Transmission request. *A challenge message is multicast by the source to all users. This challenge contains a random value v.*
2. Message generation.
   (a) *When a user U_i receives the challenge message, she computes*

$$S_{2i-1} = I_{2i-1}^{min} + \mathcal{H}(v||k_i)$$

$$S_{2i} = I_{2i}^{min} + \mathcal{H}(v+1||k_i) \qquad (2)$$

   where $\mathcal{H}$ is a one-way collision-free hash function yielding an l-bit integer as output. This condition on the output of $\mathcal{H}$ ensures that $S_{2i-1} \in \mathbf{I}_{2i-1}$ and $S_{2i} \in \mathbf{I}_{2i}$, which in turn guarantees that the sequence $\mathcal{S} = \{S_j\}$ for $j = 1, \cdots, 2u$ is super-increasing. *On the other hand, condition (1) ensures that no overflow in CT will occur when adding encrypted terms of the super-increasing sequence over the ciphertext space CT'.*
   (b) *Next, each U_i generates her message as follows:*
      i. *If she wants to transmit a 0 bit value, she generates the message*

$$M_i = E_{PK}(S_{2i-1})$$

      where $E_{PK}(\cdot)$ stands for the encryption function of the probabilistic additive public-key privacy homomorphism used.
      ii. *If she wants to transmit a 1 bit value, she generates*

$$M_i = E_{PK}(S_{2i}))$$

Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04)
0-7695-2108-8/04 $ 20.00 © 2004 IEEE

IEEE COMPUTER SOCIETY

iii. In case of transmitting ternary symbols, a third symbol (other than 0 and 1) could be transmitted if U_i sent

$$M_i = E_{PK}(S_{2i-1} + S_{2i})$$

Finally U_i sends M_i up to her parent router.

3. Message aggregation. *Intermediate routers receive messages from their child routers/users and do the following:*

   (a) *Once all expected messages $\{M_i\}_i$ have been received, the router aggregates them as $M = \sum'_i M_i$, where $\sum'$ stands for the ciphertext operation of the privacy homomorphism corresponding to cleartext addition.*

   (b) *The router sends M up to its parent router.*

4. Symbol extraction. *When the previous process completes, the source finally receives an aggregated message M, from which the transmitted symbols are extracted as follows:*

   (a) *The source constructs the super-increasing sequence $\mathcal{S} = \{S_j\}$ for $j = 1$ to $2u$ using, for each user U_i, using equations (2).*

   (b) *The source decrypts M using its private key to recover a value T which is used to solve the super-increasing knapsack problem [Merk78] and obtain the sequence $\mathcal{S}' = \{S_a, S_b, \ldots, S_j\}$ that yields the symbols sent by the users. Specifically, there are four possible cases for each user U_i:*

      i. If $S_{2i-1} \notin \mathcal{S}'$ and $S_{2i} \notin \mathcal{S}'$, then U_i sent nothing.

      ii. If $S_{2i-1} \in \mathcal{S}'$ and $S_{2i} \notin \mathcal{S}'$, then U_i sent a bit value 0.

      iii. If $S_{2i-1} \notin \mathcal{S}'$ and $S_{2i} \in \mathcal{S}'$, then U_i sent a bit value 1.

      iv. If $S_{2i-1} \in \mathcal{S}'$ and $S_{2i} \in \mathcal{S}'$, then this is an error condition in a binary transmission.

3 A secure large-scale bingo

3.1 A description of conventional bingo

Conventional, *i.e.* physical, bingo is played in a large hall. Players meet at the hall and the game begins. In this way, many bingo games are played one after the other. A *bingo game* proceeds as follows:

1. There are 99 possible bingo numbers: $\mathcal{B} = \{1..99\}$. Each of these numbers is represented by a ball in a large rotating bin. Each ball is painted with its unique bingo number.

2. Every player receives a bingo card with 15 different numbers. These numbers are distributed in three rows, with five numbers each.

3. An announcer spins the bin and selects a ball (or a computer randomly selects a number). This number is announced to the audience.

4. Then each player checks her card to see whether the announced number appears on it.

5. This is repeated until a bingo or a line are called out. There are two winning configurations:

   - **Line.** If all the numbers in a row of a card have been selected, its owner calls out line. The game pauses while the card is verified. If a line has been completed, the player receives about 8 percent of the total bet. Then, the game goes on, but nobody else can call out a line.
   - **Bingo.** When all the numbers on a card have appeared, its owner must call out bingo. The game pauses while the card is verified. If a whole card has been completed, the game ends. Note that the game always ends, because someone will complete a card sooner or later.

The above description corresponds to European and, more specifically, Spanish bingo. In other versions of bingo (*e.g.* American), the winning configurations may vary, but the basic operation of the game stays the same.

3.2 A protocol for secure large-scale bingo

An electronic version of the above bingo rules is described next. Let u be the number of players, where $u \gg 1$.

3.2.1 Enrollment

The source announces an *enrollment period* in order to allow new players to join. When a player is interested in joining a virtual bingo hall, she registers to the source using a unicast communication. The parameters used in Protocols 1 and 2 are supplied by the source. This would require sending renewed parameters to players already in the system.

Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04)
0-7695-2108-8/04 $ 20.00 © 2004 IEEE

IEEE COMPUTER SOCIETY

3.2.2 Game start

In conventional physical bingo, cards are generated and supplied by the source. In the electronic version, each player randomly creates her bingo card in the following way:

1. Every player U randomly chooses 15 different numbers from $\mathcal{B}$ in order to fill her bingo card $\mathcal{C}_U$. These numbers are sorted in ascending order and split in three rows of five numbers each.

   $$\mathcal{L}_l^U = \{b_{l,1}, b_{l,2}, b_{l,3}, b_{l,4}, b_{l,5}\}$$

   for l=1..3

   $$\mathcal{C}_U = \{\mathcal{L}_1^U, \mathcal{L}_2^U, \mathcal{L}_3^U\}$$

2. The player computes a one-way hash of each line l, in the following way

   $$\mathcal{H}_l^U = \mathcal{H}(b_{l,1}, b_{l,2}, b_{l,3}, b_{l,4}, b_{l,5}, salt_l)$$

   where $salt_l$ is a random number, and sends the resulting hash to the source. In this way, the player commits to the selected values. These hash values are sent to the source one bit at a time using Protocol 2. This procedure avoids information implosion at the source. Moreover, no user-dedicated secure connections are needed.

3. The use of Protocol 2 guarantees that the source knows the identity of the new bingo game players, as shown below.

4. Note that during this step, players may perform a micropayment that allows them to play. Otherwise, a subscription model may be used for payment.

3.2.3 Game operation

Once the game has started, bingo numbers are successively and randomly chosen by the source and checked by players, until the game ends:

1. A bingo number b is randomly chosen. This number cannot appear again during the current bingo game (due to bingo rules). b is multicast to every player.

2. Every player checks whether b is in her bingo card. If so, b is marked.

3. A transmission request is multicast by the source to all players. Player U_i generates a message M_i, according to the following message-to-symbol mapping:

   - S_{2i-1} = player has received the number, but has not completed a line or bingo.
   - S_{2i} = all five numbers of $\mathcal{L}_0^{U_i}$, $\mathcal{L}_1^{U_i}$ or $\mathcal{L}_2^{U_i}$ have appeared (line).
   - $S_{2i-1} + S_{2i}$ = all numbers of $\mathcal{C}_{U_i}$ have appeared (bingo).

   Messages are sent and aggregated following Steps 3a and 3b of Protocol 2.

4. The source finally receives an aggregated message M, from which the transmitted symbols are extracted.

   - If a user U_i has sent a line message, a unicast connection is set up between the source and U_i. The latter sends a message to the source containing the bingo numbers of the winning line, and the $salt_l$ value used for that line. The source then checks that the hash U_i committed to at the beginning of the game corresponds to the sent line. Once all messages from line-winning players have been checked, no more line messages will be accepted (due to bingo rules).
   - If a user U_i has sent a bingo message, U_i sends to the source the three lines of $\mathcal{C}_{U_i}$ and the corresponding $salt_l$ values for checking.

4 Security

We next state the security properties of our scheme, which are proven below.

Property 1 (Confidentiality) *If a secure probabilistic additive public-key PH is used in which there is a negligible probability of obtaining the same ciphertext as a result of two independent encryptions of the same cleartext, then an intruder cannot determine the symbol transmitted by a user in Protocol 2.*

Property 2 (Authentication) *If a secure public-key PH and a one-way collision-free hash function with l-bit output are used, the following holds:*

1. *the probability of successfully impersonating another user when sending a bit value to the source is 2^{-l};*

Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04)
0-7695-2108-8/04 $ 20.00 © 2004 IEEE

IEEE COMPUTER SOCIETY

2. *substituting a false message M' for a legitimate message $M \neq M'$ in the current transmission is at least as difficult as impersonation;*

3. *substituting a message M' for a legitimate message $M \neq M'$ in future transmissions using information from the current transmission is infeasible.*

Proof (Property 1): Assume the intruder captures a message M sent by U_i during Protocol 2. This message is either $E_{PK}(S_{2i-1})$, $E_{PK}(S_{2i})$ or $E_{PK}(S_{2i-1} + S_{2i})$. Decryption of M is not possible because the PH is secure and the intruder does not have access to the private key.

Exhaustive search of the cleartext carried out by M is the other attack strategy to be examined. Now, exhaustive search of the sequence values S_{2i-1} or S_{2i} by encrypting candidate values and comparing the result to M is not feasible because the PH is probabilistic and there is a negligible probability that two independent encryptions of the same cleartext yield the same ciphertext. Therefore, the above comparison (and thus exhaustive search) will fail with overwhelming probability. □

Proof (Property 2): By the same argument as for Property 1, we need only prove this property for the basic construction. In the impersonation attack, an intruder who wants to impersonate user U_i tries to generate a message $E_{PK}(S_{2i-1})$, $E_{PK}(S_{2i})$ or $E_{PK}(S_{2i-1} + S_{2i})$, Now, the intruder needs to compute S_{2i-1} or S_{2i}. Each term S_j of the super-increasing sequence $\mathcal{S}$ is pseudo-randomly chosen within an interval $\mathbf{I}_j$ containing 2^l integer values. The choice is made using a one-way collision-free hash function of the challenge and the secret key k_i unknown to the intruder, as shown in equations (2). Thus, the probability of the intruder randomly hitting S_j is at most 2^{-l}. Remark that exhaustive search is not feasible, since there is no way of checking whether the right S_j has been hit (there is no way for the intruder to make sure whether the message generated with the candidate S_j is correct).

A substitution attack can be mounted in the current transmission or in future transmissions:

- In the current transmission, assume the intruder wants to substitute a false message M' for an authentic message M sent by U_i, with $M' \neq M$. Without loss of generality, let $M = E_{PK}(S_{2i})$; the intruder wants to transform M into $M' = E_{PK}(S_{2i-1})$ or $M' = E_{PK}(S_{2i-1} + S_{2i})$. This requires the following steps: i) recover S_{2i} from M; ii) compute S_{2i-1} with knowledge of S_{2i}; iii) compute M'. Thus, even if decrypting M at step i) was easy (which it is not), solving step ii) is as difficult as mounting a successful impersonation attack (see above).
- A second possibility is for an internal intruder to use information derived from a current transmission of a message by U_i to alter future messages sent by U_i. But this is infeasible, because in subsequent executions of Protocol 2, a different super-increasing sequence will be used to encode the messages which does not depend on the current super-increasing sequence (see equations (2)).

□

5 Conclusion

Moving the bingo game from physical casinos to a large-scale virtual scenario requires efficient and secure many-to-one communication between players and the virtual bingo hall. This paper has presented a scheme for large-scale virtual bingo which avoids implosion in the many-to-one communication. The resulting scheme aggregates information using a probabilistic public-key additive privacy homomorphism and guarantees the security of communications.

References

[Merk78] R. C. Merkle and M. Hellman, "Hiding information and signatures in trapdoor knapsacks", in *IEEE Transactions on Information Theory*, vol. 24, no. 5, pp. 525-530, 1978.

[MSEC03] Multicast Security Working Group (MSEC WG). `http://www.securemulticast.org`

[Okam98] T. Okamoto and S. Uchiyama, "A new public-key cryptosystem as secure as factoring", in *Advances in Cryptology - EUROCRYPT'98*, ed. K. Nyberg, LNCS 1403, Berlin: Springer-Verlag, pp. 308-318, 1998.

[Quin01] B. Quinn and K. Almeroth, *IP Multicast Applications: Challenges and Solutions*, Internet RFC 3170, 2001.

Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04)
0-7695-2108-8/04 $ 20.00 © 2004 IEEE

IEEE COMPUTER SOCIETY