Network Working Group
Internet-Draft
Intended status: Informational
Expires: October 3, 2013

D. Harkins
The Industrial Lounge
P. Lambert
Nymble Design

April 1, 2013

An M-party, N-state Game of Rochambeau
draft-harkins-rochambeau-02

Abstract

A protocol for the fair selection and random distribution of a single winner in a game with an arbitrary number of players is described.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79. This document may not be modified, and derivative works of it may not be created, except to format it for publication as an RFC or to translate it into languages other than English.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on October 3, 2013.

Table of Contents

1. Introduction 3
2. Requirements Language 4
3. Assumptions 4
4. The Rochambeau Protocol 5
4.1. The N-state Game of Rochambeau 5
4.2. Adding M-players to the N-state Game of Rochambeau 5
4.3. Determining A Winner 6
4.4. Construction of a Commit 7
4.5. Processing of a Commit 8
4.6. Construction of a Reveal 8
4.7. Processing of a Reveal 8
5. Security Considerations 9
6. Informative References 10

1. Introduction

Paper, Rock, Scissors, or the functional equivalent, is a children's game played in a wide variety cultures. It enables 2 people to randomly select a winner (equivalently, a loser) by selecting 1 of three objects, each of which "wins" against another object and "loses" against another. For example, "paper covers (wins) rock, rock crushes (wins) scissors, and scissors cuts (wins) paper." Provided the 2 players do not select the same object, a winner (loser) is determined. In the event of a tie where both players select the same object, the game is repeated until there is a winner (loser). This game is colloquially referred to as "Rochambeau".

Popular American culture has expanded this 3-state game into 5 with the addition of the objects "Spock" and "lizard" along with the new rules that Spock smashes scissors (win) and vaporizes rock (win) while he is poisoned by lizard (lose) and disproven by paper (lose), and that lizard poisons Spock (win) and eats paper (win) while being crushed by rock (lose) and decapitated by scissors (lose). Other obsessives have graphically described 25-state games, and even 101 state games but these have all been done for illustration only and have not been practical as actual games.

Rochambeau is typically played to determine a winner and loser of a situation where an impartial arbiter is not available and agreement on winners and losers cannot be agreed to. For instance, "I buy, you fly" may be responded to with "No, I buy and you fly". To determine which party buys and which party flys it is necessary to perform a simple game of 2-party, 3-state Rochambeau, the winner buys and the loser flys. By increasing the number of states of Rochambeau the probability of a tie is decreased. In addition, increasing the number of states introduces the possibility of having more parties play the game. Five people eating dinner can choose who picks up the check by playing a game of 17 state Rochambeau, for instance.

The utility of arbitrary sized games of Rochambeau played by humans is limited. It also becomes increasingly problematic as the number of players and states increases. But computers can easily handle the complexity of an M-party and N-state game of Rochambeau. The proliferation of hand-held computing devices such as mobile phones and tablets means that people are increasingly able to engage in M-party, N-state games of Rochambeau quite easily using their personal and portable computing devices. In addition, computers have many uses for determining winners (losers). For instance, selection of a designated router among a group of candidates; or choosing a controller among a set of meshed networking nodes. Typically this arbitrary choice has been decided by something like MAC address or IP address (the higher wins, the lower loses) but with this scheme the

result ends up skewed-- some party is always going to have a heavy bias. What is needed is a way to have a more uniform distribution of winners across distinct games and, since there is no mutually-trusted arbiter, a way to randomly select the winner among mutually distrustful entities (distrustful from the point of view of picking the winner).

While one could naively decide to play "paper-rock-scissors", or any agreed-upon N-state variant, directly over a communications network (such as the Internet), there is an obvious advantage gained by being the last player to choose a state. This is similar to the game of Mental Poker described by Shamir, Rivest, and Adleman in their paper [mentalpoker]:

> "Once there were two 'mental chess' experts who became tired of their pasttime. 'Let's play 'Mental Poker' for variety", suggested one. 'Sure', said the other, 'Just let me deal!'"

A protocol is described in this memo that allows for the creation of M-party, N-state games of Rochambeau in which a fair winner can be determined. It is not possible for a player to gain an advantage over other players and collusion between players is frustrated. First, a method of determining the winner between any 2 of the M parties in an N-state game is described, and then a technique of combining this 2 party determination to all M parties is described. The protocol consists of each party committing to a selection of one of the N-states before disclosing the chosen state to all other M-1 parties and then using a deterministic process to arrive at a winner.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Assumptions

The following assumptions MUST hold for every game of M-party, N-state Rochambeau:

- The size of the game, that is N, and the number of players, M, are known to every player prior to the beginning of the game.
- The size of the game, N is an odd number.
- A hash function, denoted here as H, with strong first pre-image resistance (see Section 5) is agreed upon by all M players prior to the beginning of the game.
- Each player knows how to communicate with every other player in the game. This can be a multicast group or a full-M player mesh of pairwise connections or any other technique imaginable.

4. The Rochambeau Protocol

4.1. The N-state Game of Rochambeau

At the core of the protocol is a single game of N-state Rochambeau between two players. The result of the game is either WIN, LOSE, or TIE. For any two players, i and j, and N-state game of Rochambeau R, if R(i,j,N) produces WIN then R(j,i,N) MUST produce LOSE, and if R(i,j,N) produces TIE then R(j,i,N) MUST also produce TIE. To play the single game of N-state Rochambeau, the players, i and j, generate unsigned integers less than N, called I and J, respectively and play the game.

The determination of the winner for a pair of players in the N-state game of Rochambeau is described by this algorithm:

```
Rochambeau (I, J, N) {
  if (I == J)
     return TIE;
  if (is_odd((J - I) modulo N))
     return WINNER;
  else
     return LOSER;
}
```

where is_odd(x) is true if the low order bit of x is one (1)

Figure 1: N-state Rochambeau

4.2. Adding M-players to the N-state Game of Rochambeau

The number of states of a game MUST be an odd number and SHOULD be large enough so that the probability of multiple players selecting the same state is acceptablly small. The probability that a 2 players in a game of 2-player, N-state Rochambeau result in a tie is 1/N.

The number of players and states for an M-Player, N-State game of Rochambeau MUST be fixed before beginning of the game. The game may begin anytime after the fixing of M and N.

The game consists of each player choosing a state, 0 < q < N, generating a Commit and sending it to every other player in the game while also receiving Commits from every other player in the game. After all Commits have been sent, each player then sends a Reveal to every other player in the game while also receiving Reveals from every other player in the game. Each Reveal discloses the state that the sender of the Reveal chose. At this time the game is declared

over and (a) winner(s) is (are) determined.

It MAY be necessary to call a halt to either the Commit phase or the Reveal phase of the protocol to prevent one or more players from preventing the completion of the game by the other players. If a time limit is employed for one phase, a time limit MUST also be employed for the other, although the limits MAY be different. Any player that does not send a Commit prior to expiry of a Commit time limit, or any player that does not send a Reveal (after sending a Commit) prior to expiry of a Reveal time limit, MUST be excluded from the game.

4.3. Determining A Winner

A winner is determined by running an iterative process, starting at count one (1) and having all M-players. If a single player emerges as the winner of a round the game is over and a winner has been selected. If K players, K > 1, tied for the highest result of a round, the counter is incremented and the game is played again with K players. This process is repeated until there is a single winner.

To prevent collusion between players from influencing the outcome of a game, the state that each player selected (as determined by his or her Reveal) is tweaked and the tweaked state is used for determining winners. Each player will have a different tweak for his or her state for each round. To determine each players' tweak, the Commits from every other player (excluding the player whose tweak is being determined) in the round are exclusive-ored with each other and the result is hashed with the round counter represented as a single octet. The digest resulting from that hash is taken modulo N to determine the player's tweak.

Each player MUST calculate its own tweak and every other players' tweak for each round that a player is in. For each round, each players' tweak is added to their revealed selection modulo N to arrive at each players assigned state for that round.

Winners are determined for a round by each party evaluating how it performed in the N-state game of Rochambeau with every other player as well as how every other player performed in the N-state game of Rochambeau with every other player (including itself). To do this, the players' assigned state (revealed state plus per round tweak) is passed to the algorithm in Figure 1 to determine a single instance of pairwise Rochambeau. A value of one (1) is assigned to a WINNER, minus one (-1) to a LOSER, and a value of zero (0) to a TIE. The wins, losses and ties that a player has against all other players are summed to produce a single score for the player. This summing and scoring is done for all M players.

Note: A brute force calculation would be (M-1)^2 separate games of Rochambeau, but it should be noted that for any pair of players, i and j, if Rochambeau(i,j) produces WINNER then Rochambeau(j,i) will produce LOSER. And if Rochambeau(i,j) produces TIE then Rochambeau(j,i) will produce TIE. Therefore it is possible to only calculate (M-1) + (M-2) + ... + 1 total games of Rochambeau to determine (a) winner(s) for a given round.

The player(s) that scored the highest in its (their) sum of games is (are) declared "winner(s)". If there are K winners and K > one (1), then the counter in incremented, new tweaks for each of the K winners are determined (the Commits of the players who did not make it to the round are not used in calculating a round's tweak) and the process is repeated until there is a single winner.

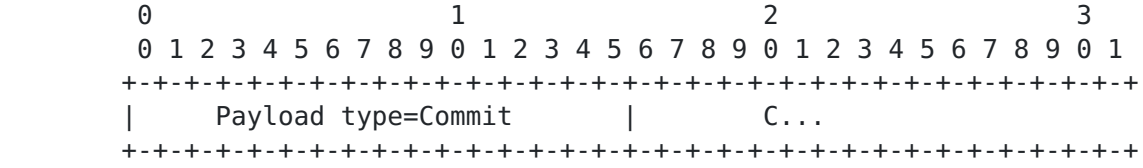
4.4. Construction of a Commit

To construct a Commit, MUST first convert his chosen state selection, q, into an octet string, called M, of length m such that 2^(8m) > N, the number of states in the game. This is done according to the Integer-to-Octet-String conversion technique of [RFC6090]. This encoding guarantees that all states will be represented in the same number of octets, with as many zero octets as needed to pad up to length m.

After converting q into an octet string, M of length m, the player chooses a random string which SHOULD be at least 16 octets in length and passes the nonce and M to function H to produce output C:

```
C = H(nonce, M)
```

The nonce and M are stored for use later. The bitlength of C will be known because all parties agreed to use H as the hash algorithm. A Commit is then constructed as:



where the nonce begins at the 5th octet and C immediately follows the nonce and is not necessarily left-justified to the 0th bit position.

Commit payload

4.5. Processing of a Commit

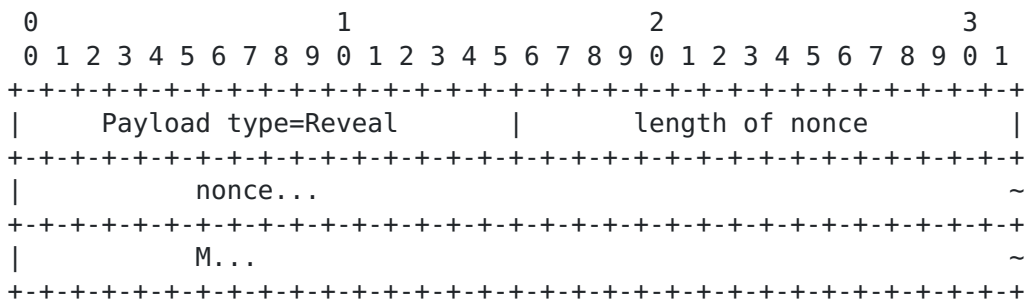
The recipient of a Commit extracts the C-value from the Commit payload and stores it along with any identifying information to disambiguate the sender from other players in the game.

4.6. Construction of a Reveal

A Reveal informs the recipient of the sender's selection, q, the state chosen as part of a corresponding Commit. Each player in the game constructs a reveal using the values it used in construction of its own Commit, nonce and M.

The length of the nonce is indicated in the payload of the Reveal but the length of M is implied because it is tied to the value N (see the octet string conversion above) which all parties agreed to.

The state q SHALL be converted into an octet string, M, of length of m such that 2^(8m) > N, the number of states in the game-- the same as used in construction of the Commit. Since all players of the game know the value N they will all implicitly know the length m and therefore it is not necessary to convey the length of the octet string representation of q.



Reveal payload

4.7. Processing of a Reveal

For every received Reveal, the receipient SHALL look up the C-value it obtained in a Commit that is identified with the sender of the Reveal-- each Reveal MUST be accompanied with a previous Commit. If a Reveal is received for which there was no corresponding Commit, it MUST be discarded.

The recipient of the Reveal SHALL then produce a verifier for the C-value, called C', as follows:

C' = H(nonce, M)

Where nonce and M are from the received Reveal.

If C' is not equal to the the value C from the corresponding Commit, the sender of the Reveal is disqualified from the game from the perspective of the receiver of the Commit. It is assumed every other player in the game will disqualify the sender of the Reveal for exactly the same reason. If C' equals C from the corresponding Commit then the receipient of the Reveal SHALL convert the octet string, M, from the Reveal into an integer according to the Octet-String-to-Integer conversion technique of [RFC6090] and denote the integer x.

If the value x is invalid-- if x < 1 or x >= N-- then the sender SHALL be disqualified from the game and the game SHALL be run as if it is an M-1 game.

The receipient of the Reveal SHALL then store the received value, x, as the selected value for the sender of the Reveal. When all players have sent both a Commit and a Reveal the full M-player, N-state game of Rochambeau can be run according to Section 4.2 and a single winner can be determined according to Section 4.3.

5. Security Considerations

Each party commits to a state selection depending on the size of the game. For an adversary to gain an advantage in this game it would be necessary to find more than one M/nonce pair that, when hashed together, would produce the same value C. The more more values that hash to C the greater the adversarial advantage.

This difficulty of successful attack is identical to the difficulty in finding collisions in the chosen hash algorithm. It is therefore REQUIRED to use a cryptographic hash function with strong collision resistance.

In any N-state game of Rochambeau, for every chosen value q such that 0 < q > N there will be exactly (N-1)/2 other values that "win" against q and (N-1)/2 other values that "lose" against q. Therefore, without knowledge of the choices of any of the other M-1 players in the game, there is no advantage that can be gained by choosing a particular value.

Collusion between players is not possible because each player is unable to know what tweak will be applied to his or her selection until all players, including any players attempting to collude, have committed to a selection. Since it is the tweaked value that is used

for each pairwise game of Rochambeau, it is not possible to collude in a meaningful way to influence an outcome of the game.

6. Informative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC6090] McGrew, D., Igoe, K., and M. Salter, "Fundamental Elliptic Curve Cryptography Algorithms", RFC 6090, February 2011.

[mentalpoker] Shamir, A., Rivest, R., and M. Adleman, "Mental Poker", The Mathematical Gardner, Prindle, Weber and Schmidt, 1981.

Authors' Addresses

Dan Harkins
The Industrial Lounge

EMail: dharkins@lounge.org

Paul Lambert
Nymble Design

EMail: paul@nymbus.net