Protocol for Blind Random Selection of Peers in a Pool of Participants: The Among Us Protocol

Peter Parker, ChatGPT
peterparker303e24@proton.me
Ethereum Account: 0x5a610334D65f6A83237d287A17dB1baa314eF6E6

January 17, 2024

Abstract

In a system comprising two parties, denoted as A and B, along with a network of N anonymous participants, a protocol is established to enable A and B to randomly select k participants. Subsequently, only the chosen k participants possess the capability to self-identify as specially selected. Importantly, neither A, nor B, nor the remaining $N - k$ participants know the identities of the selected individuals with any significant probability.

1 Introduction

The Among Us Protocol presents a novel solution to a challenge encountered in the creation of a decentralized jury voting system. The primary objective is to facilitate the democratic resolution of conflicts between two untrusted parties while ensuring a robust mechanism that minimizes the risk of tampering with both the jury composition and the votes cast by selected participants.

Through the internet, participants are anonymous, making user identity verification challenging. Within this online environment, a user can adopt multiple identities, introducing the potential for one participant to wield various personas within the participant pool. Consequently, both entities A and B possess the ability to engage as voters in the conflict resolution system. Moreover, in the presence of a participant pool, each party has an interest in selecting a jury member corresponding to their concealed identity. The protocol incorporates robust measures to prevent potential exploits. Specifically, it guards against the revelation of the selected jury members to either party, preventing the leveraging of bribes or blackmail. These protective barriers are integral to the protocol's design, ensuring the integrity of the conflict resolution process.

The protocol additionally facilitates the transmission of a confidential message from party B to the participants randomly chosen. Notably, party B can obtain the secret using dishonest behavior, but the network is able to expose any such dishonest behavior. Furthermore, the selected participant may choose to disclose the secret message. Hence, the term 'secret message' within the context of this protocol should be taken lightly.

For the protocol's outcome to maintain security, specific preconditions must be satisfied. A and B both hold onto secret information that must not be shared among the network to retain the integrity of the protocol. Additionally, both parties utilize local randomness which either party can manipulate, so the parties should have opposing utilities so that randomness is the optimal strategy for choosing the values.

The protocol relies on cryptographic tools, incorporating universal hash functions along with asymmetric public and private key encryption. To achieve the intended outcome of the protocol, it is imperative to employ effective algorithms for these cryptographic components.

The outcome of the protocol is to randomly select special individuals from a group of participants, ensuring that each participant remains unaware of the selection status of others. Thus, it is fitting to name this protocol, 'The Among Us Protocol', after the video game, 'Among Us', as it closely mirrors the rules of selecting an imposter.

2 Protocol

Let A have an asymmetric encryption public key p_A and private key s_A, and similarly, let B have a public key p_B and private key s_B. Consider N as the total number of participants in the network. The network initiates with a pre-established, ordered list of participant IDs denoted as:

$$[a_1, a_2, \ldots, a_N]. \tag{1}$$

For each participant a_i at position i on the list, the corresponding public and private keys are denoted as p_i and s_i, respectively.

Let b represent a set of random padding bytes with a length of λ, ensuring uniqueness across all instances of b. The parameter λ should be sufficiently large to render the computational task of determining the random bytes b computationally challenging, even when provided with a known plaintext and public key. This precautionary measure aims to prevent any party from deducing small or structured data through inference.

Let m_i be a randomly generated bitmask for participant a_i with a length denoted as l. The length of the bitmask, l, must be agreed upon by the network before the protocol initiation. In the scenario where party A intends to transmit a secret message, l should be at least equal to the length of the secret message plus λ.

To initiate the protocol, participants in the network broadcast their randomly generated bitmasks. Each bitmask undergoes a multi-step encryption process: first, encryption with party A's public key, followed by padding with random bytes, and finally, encryption with party B's public key, coupled with a digital signature. The resulting encrypted bitmasks are organized as:

$$[(a_1, p_B(b, p_A(m_1))), (a_2, p_B(b, p_A(m_2))), \ldots, (a_N, p_B(b, p_A(m_N)))], \tag{2}$$

Upon completion of the broadcast of encrypted bitmasks by all participants, or when a predetermined deadline is reached, A can proceed to the next step of the protocol.

Let k represent the number of selected participants in the network. A selects k unique random numbers, each denoted as n_i, chosen from the range 1 to N. Subsequently, the binary representation of this list is padded with b and then hashed. The resulting hash value is broadcasted to the network along with A's digital signature:

$$\text{hash}(b, [n_1, n_2, \ldots, n_k]). \tag{3}$$

Party B proceeds by decrypting each message in (2), discarding the b values while retaining the information for future use. Subsequently, B shuffles the order of elements in the list. The newly ordered list is then encrypted using party A's public key, and the result is broadcasted to the network along with B's digital signature. Let $\tilde{m}_i$ represent the i^{th} bitmask in the newly shuffled list:

$$p_A([p_A(\tilde{m}_1), p_A(\tilde{m}_2), \ldots, p_A(\tilde{m}_N)]). \tag{4}$$

At this point, A gains access to all bitmasks m_i, with the sender of each bitmask concealed due to B's decryption and shuffling of the order. The discarded padding bytes render it infeasible for A to reconstruct the original list and discern the senders based on the bytes data.

A proceeds by randomly selecting k elements from the previously generated list and decrypting their corresponding bitmask values. Let $\bar{m}_i := \tilde{m}_{n_i}$, resulting in the list:

$$[\bar{m}_1, \bar{m}_2, \ldots, \bar{m}_k]. \tag{5}$$

Let x_i denote the secret message, prefixed with b, that A intends to transmit exclusively to the selected participants. Each message destined for the k participants may vary in content, but must differ in padding bytes.

Subsequently, A broadcasts the following list of hashes, without padding, using their digital signature:

$$[\text{hash}(x_1), \text{hash}(x_2), \ldots, \text{hash}(x_k)]. \tag{6}$$

Let $\hat{}$ denote the XOR binary operator. Party A compiles a list that will be broadcasted to the network along with their digital signature:

$$[\bar{m}_1 \hat{} x_1, \bar{m}_2 \hat{} x_2, \ldots, \bar{m}_k \hat{} x_k]. \tag{7}$$

At this stage, there are k random and unique participants in the network, each possessing access to a single secret message. The participant a_i can verify that they have obtained the secret message by XORing each entry in the list with their generated bitmask m_i. If a secret message has been encoded the participant would be able to see it at this point. Subsequently, the participant hashes each entry and compares it to the hash of each broadcasted entry in (6). Successful matching confirms the correct decryption of the secret message and indicates special selection.

The participant is now eligible to vote by composing a message containing any necessary information for the voting system. The broadcasted message must include the decrypted message x_i, their bitmask value m_i, the respective padding b, and the participant's digital signature. The vote is valid only if the hash(x_i) value matches an entry in (6), and $p_B(b, p_A(m_i))$ matches the value they had broadcasted earlier. All peers in the network can compute and verify these values. If not, the broadcasted vote is deemed illegitimate, signaling a potential leakage of the bitmask key by A or the selected participant.

The next phase of the protocol initiates once all k messages have been decrypted and all k votes have been broadcasted to the network, or until a predetermined deadline. The specific implementation of this step may vary depending on the use case of the system employing the protocol. Upon completion of the voting phase, the implementation may determine a verdict for the conflict between parties A and B based on any previously agreed-upon voting scheme. However, it's important to note that the results cannot be confirmed until the conclusion of the next step.

Subsequently, B broadcasts (4) unencrypted, including its respective padding bytes b. The network, in turn, can verify that reapplying the public key p_B will yield a list containing all the encrypted bitmasks of all participants. Any discrepancy in this computation indicates potential tampering by B, and the network can present proof of fraud, leading to punishment.

$$[(b, p_A(\tilde{m}_1)), (b, p_A(\tilde{m}_2)), \ldots, (b, p_A(\tilde{m}_N))] \tag{8}$$

Utilizing the information provided above, any participant in the network can compute the encryption using B's public key and compare each value with the actual broadcasted value by each participant. When making the comparison entries must be shuffled back into order, the resulting lists are as follows:

$$[p_B(b, p_A(\tilde{m}_1)), p_B(b, p_A(\tilde{m}_2)), \ldots, p_B(b, p_A(\tilde{m}_N))] \tag{9}$$
$$[p_B(b, p_A(m_1)), p_B(b, p_A(m_2)), \ldots, p_B(b, p_A(m_N))]. \tag{10}$$

Participants in the network can also verify the validity of the random shuffle construction performed by B and ensure that it preserves the information intended for A. Any peer in the network can remove the padding bytes, encrypt the list with A's public key, and compare the value to (4):

$$p_A([p_A(\tilde{m}_1), p_A(\tilde{m}_2), \ldots, p_A(\tilde{m}_N)]). \tag{11}$$

Next, A broadcasts (3) unhashed:

$$(b, [n_1, n_2, \ldots, n_k]). \tag{12}$$

Peers in the network can verify that the value matches the hash. If the value does not match the hash, it indicates potential tampering by A, and peers can provide evidence of their dishonesty, leading to punishment.

Subsequently, A broadcasts the unencrypted ordered list of all bitmasks $\tilde{m}_i$, all selected bitmasks $\bar{m}_i$, and the corresponding x_i values:

$$[\tilde{m}_1, \tilde{m}_2, \ldots, \tilde{m}_N] \tag{13}$$

$$[\bar{m}_1, \bar{m}_2, \ldots, \bar{m}_k] \tag{14}$$

$$[x_1, x_2, \ldots, x_k]. \tag{15}$$

Peers in the network can now reconstruct (4):

$$p_A([p_A(\tilde{m}_1), p_A(\tilde{m}_2), \ldots, p_A(\tilde{m}_N)]). \tag{16}$$

If the reconstruction does not match that which was sent by B to the network, it indicates potential tampering by A. Consequently, the network can provide evidence of fraudulent activity by A, leading to punishment. Peers in the network can also verify that A has chosen the corresponding indices from (12) to obtain selection bitmasks (14); any deviation signals potential tampering by A in the selection process.

Additionally, peers in the network can now reconstruct (6) and verify whether A has correctly broadcasted the selection hashes. An incorrect message indicates fraudulent behavior in the selection process, and A can be proven guilty by the network, leading to punishment.

If, at this point, no infractions in the protocol have been identified, the network can solidify the election results, and the protocol is considered complete.

It's important to note that the protocol is agnostic to the system in which the votes and verdict are determined. The Among Us Protocol provides a mechanism for a decentralized network of peers to reach a democratic verdict between two parties. The jury of participants is selected from a larger pool, and both A and B agree that this selection is a random and anonymous process to the parties and other network members. The protocol's design ensures the prevention of manipulation in the selection of jury participants by either party, and it anonymizes voters' identities to safeguard against bribery or blackmail during the voting process.

3 Applications

The Among Us Protocol was crafted as a tool for a broader system, shaping the design specifications to meet specific requirements. Notably, the voting scheme necessitated two parties with opposing interests and a large pool of participants, from which a subset would constitute the jury for conflicts between the two. While an ideal democratic process involves everyone voting on each issue, practicality limits such broad participation. Hence, a system must determine who holds the voting power, a crucial influence that individuals may seek through malicious means. This becomes particularly crucial in an anonymous and untrusted internet environment. To mitigate manipulation within the network, fairness is imperative. The protocol addresses this by allowing both parties in conflict resolution to introduce randomness. Furthermore, ensuring voter anonymity from conflicting parties is vital to minimize potential bribery and blackmail. The conceptual framework of Among Us imposters served as an analogy for a solution meeting these conditions, and the only difference was to apply it in a decentralized manner.

While initially designed for a voting mechanism, this system can find broader application in any lottery system where participants are randomly selected. Unlike systems relying on a single trusted source for integrity, the protocol operates on the foundation of distrust towards two entities. This inherent distrust proves essential and is applicable when two parties possess opposing utilities, making it mutually beneficial for them to perform actions randomly and keep certain information confidential. This is exemplified in the context of decentralized conflict resolution, making it an excellent application for this protocol.

4 Jury and Vote Tampering

While there exist multiple avenues through which A, B, and other peers on the network could potentially tamper with the voting process in this protocol, this section outlines certain assurances despite the potential participant interactions.

A may possess a masked identity in the peer network, prompting them to prefer choosing the bitmask key associated with their identity. However, A is required to disclose the hash of the selected bitmask indices before gaining access to the scrambled list of bitmasks. Notably, A gains no significant advantage in selecting their own bitmask over choosing a bitmask associated with any other peer. If A were to disregard their own announcement or withhold the contents corresponding to the announcement hash, it would provide evidence of dishonest conduct by A.

If B possesses a masked identity in the peer network, they might be inclined to choose the bitmask key associated with their identity. B could potentially replace all received bitmasks with generated ones and encrypt each with A's public key. Given that A randomly selects an unknown bitmask, while B generated all bitmasks, A ends up using bitmasks known only to B. Consequently, B can decrypt all secret messages, deducing the values x_i that yield matching hashes. However, B faces a limitation in voting, as the encrypted bitmask values won't match the values earlier announced by B. Peers in the network can thus expose B for tampering with the protocol.

Random peers within the network can potentially disrupt the system by generating a trivial bitmask, such as an all-0 byte value for m_i. If this peer happens to be the one specially chosen, anyone can decipher the secret message upon its announcement. The protocol's reliance on the correct padding b for the individual casting the vote ensures that other peers cannot steal their vote. Additionally, A lacks the capability to guarantee that the specially selected peer refrains from disclosing the 'secret message' despite protocol constraints.

Random peers within the network can also disrupt the system by announcing an incorrectly formatted bitmask or encryption data. To address this, the protocol implementation could incorporate checks for edge cases, allowing either A or B to disregard such participants. In the scenario where random peers provide incorrect data, the protocol could proceed to the next available bitmask in the list. In the event of a malicious disregard by A or B toward a valid participant, the participant can announce their unencrypted data, demonstrating to the network that, after encryption using public keys, it matches their announced value. Such evidence could lead to punishment against A or B for their manipulation.

Given the decentralized and anonymous nature of the protocol over the internet, there's a risk that A, B, or another entity could obscure their identity and engage in the system as a substantial number of participants. This conduct would significantly increase the entity's probability of being specially selected. To maintain the integrity of the system using this protocol, it's crucial to make it challenging for a single attacker to dominate the network. The risk can be mitigated by ensuring the participant pool is vast, or that a single entity faces computational or financial infeasibility in attempting to overrun the network.

Fundamentally, this protocol relies on A and B having opposing utilities, making it impractical in scenarios where A and B can conspire together for a common goal. An alternative approach involves modifying the protocol into a more intricate procedure, wherein all peers in the network contribute to the shuffling of the bitmask keys. This variant facilitates the random selection of participants, ensuring that even if numerous peers conspire together, the outcome remains unknown and random. The proof of this modified protocol is left as an exercise for the reader.

5 Conclusion

The Among Us Protocol provides a decentralized solution to a unique problem, incorporating a set of constraints that leverage randomness and anonymity. In this protocol two parties A and B contribute randomness and safeguard secrets to ensure anonymity. The protocol relies on cryptographic tools, including asymmetric public and private key cryptography and universal hash functions. It finds application in systems requiring a decentralized random lottery mechanism among a pool of participants. Therefore, a jury selection system for voting on the resolution of a conflict between two opposing parties stands out as a fitting application for this protocol.