PANGEA/Privatebet - Decentralized and generalized shuffled decks with faceup and facedown support and recovery from minority nodes disconnecting

***Abstract*—The idea is to develop a privacy enhanced peer-to-peer gaming system which allows players to play and transact funds without need of any centralization institution. The platform is resilient against collusion between any of the participating entities, player disconnection and privacy tracking. Each entity in the system associates with a key pair, where a public key represents the identity of an entity, and the use of digital signatures in every move by an entity represents the sense of ownership. PANGEA is based on $M\ of\ N$ trust, which means that at any point in time during the game, a move can't be made unless $M of N$ players agreed where M represents majority and is of $(N/2)+1$ and fund distribution is handled using an $M\ of\ N$ multi-signature transaction. It's likely that players may disconnect while in game, there can be many reasons for this, so the deck reconstruction will happen every time when there isn't M players and the value of M is adjusted to $(N-\delta)/2+1$ and then newly constructed deck will be redistributed, where δ be the number of players get disconnected. The state of the game will be maintained whenever reconstruction happens, and in every instance the game resumed from the point where it is halted. CHIPS, a bitcoin fork, is a token on this platform, and is transacted using the JUMBLR process in Komodo, which brings anonymity to CHIPS. The consensus of the game is made by recording every move on the blockchain.**

***Keywords*—Private bet, Block chain, CHIPS.**

I. Introduction

The internet or any networking is designed to connect the machines, not people or identities associated with them. In the process of establishing trust, identities are associated with the machines, which unfortunately made the whole ecosystem centralized and vulnerable to succumb to the law of the land. Even further association of identities to the machines comes up with the cost of privacy and unlawful regulations imposed by governments. The advancement in cryptography techniques made it possible to establish the trust in a decentralized environment by concealing the identities of the communicating entities. This paper proposes a generic gaming paradigm called PANGEA, which allows the game to be played fairly over a decentralized network.

When money depends on the accuracy of the numbers, it is important to make sure that it is the right number. When it comes to cards, making sure that the deck was not tampered with in any way is of utmost importance. To achieve this, PANGEA uses a method similar to coin shuffling and all participants are involved in the process of deck shuffling. If just single player is honest then the deck is properly shuffled. The limitations with the other known methods for mental poker protocols are that they require a lot of bandwidth and are not as fast as PANGEA, which uses a single $curve25519$ field multiplication for the vast majority of its encrypt and decrypt functions. This creates an order of magnitude in speed and reduction of bandwidth required.

Often, there is high possibility that the entities in the decentralized environment can collude. PANGEA was designed such that it is impossible that any of the entities can collude. Our proposed approach consists of three entities.

1) Deck Creating Vendor (DCV)
2) Blinding Value Vendor (BVV)
3) Players (P)

The primary goal of any game is to make sure its played in a fair manner and PANGEA is designed to ensure that. Non-Repudiation is provided such that none of the entities in the game can deny their actions. Since the game is played over a decentralized network where the privacy of the players needs to be preserved, a lot of math is required to enable trust in such scenarios. The software shouldnt allow any parties to collude and it should be able to reconstruct the deck and have the capability to take the game forward in case if any of the players gets disconnected in the middle. In the proposed approach, every entity in the game is involved in the shuffling process. DCV and BVV does the digital signing of the shuffled deck in order to make sure nobody can spoof their roles. Shamirs shared secret keys are used to carry the game forward based on M of N players trust.

Any move in the game is possible only if $M\ of\ N$ players are agreed to share the Shamir shards. If $(N-M+1)$ players get disconnected it is impossible to validate the move for the remaining players, in such scenarios, reconstruction and redistribution of the deck is done by the BVV in order to make the game progress.

Reconstruction and redistribution happens only if the number of active players in the game is less than the minimum number of players that are required to play the game, $i.e\ M$. If more of such scenarios happens very often, then it can impose some computing load on BVV but this computational overhead is negligible, and the players will experience a seamless transition during the reconstruction and redistribution of the deck.

II. Working Methodology

Card deck handling, fund handling and privacy handling are the three major tasks that needs to be achieved in order to ensure the fair play of the game.

A. Card Deck Creation

The initial card deck is created such that none of the parties participating in the game should be aware of which slot what card presents. The deck is shuffled such that no entity can guess the next card in the sequence. The randomness in the shuffling process ensures the fairness of the game. Trust is established among the players by providing a provision that every player in the game can verify each other player's move in the game.

All the entities in the game $i.e.$ Players, DCV, BVV and Cards are associated with a set of key pairs and the public keys of these key pairs of Players, DCV and BVV represents the identities of the corresponding entities and the private keys ($i.e$ random scalar values) of the corresponding key pairs of the Cards which represent identity of the cards in the context of the game. The Card's identity is secret and is only to be revealed as the game progresses.The scope of association of these identities to the entities are restricted to the context of the game. In the initial card deck creation process, the flow of events and the order of shuffling the deck among the entities is as follows:

$$Player \rightarrow DCV \rightarrow BVV \rightarrow Player$$

Let N be the number of players and Z be the number of distinguishable cards in the game.

1) At Player: Every entity in the game is associated with a key pair on the $curve25519$. Let p_i be the 32 byte random scalar value generated by the i^{th} player and P_i be the public point on the curve corresponds to the i^{th} player.
$i.e.$

$$P_i = p_i * G$$

Each player generates a high entropy 32 byte random numbers as many as Z and sets the second byte as the index of the card.
$i.e.$

$$R_{ij}[2] = j \quad \forall\, 1 \leq i \leq N, 1 \leq j \leq Z$$

Where R_{ij} represents the j^{th} random value generated by the i^{th} player or in simple terms it represents the identity of the j^{th} card of the i^{th} player. The set of 32 byte random values generated by i^{th} player are represented as R_i.
$Where$

$$R_i = \left\{ \; R_{ij} \middle| 1 \leq j \leq Z,\; R_{ij}[2] = j \; \right\}$$

The player performs the permutation on R_i, $i.e.$ $\sigma_i(R_i)$ and keeps the permutation as secret to the player and is represented as

$$\begin{aligned} \sigma_i &= \left\{ \; \sigma_i(j) \middle| 1 \leq j \leq Z \; \right\} \\ \sigma_i(R_i) &= \sigma_i \left(\left\{ \; R_{ij} \middle| 1 \leq j \leq Z,\; R_{ij}[2] = j \; \right\} \right) \\ &= \left\{ \; \sigma_i(R_{ij}) \middle| 1 \leq j \leq Z, \right. \\ &\qquad \left. \sigma_i\Big(R_{ij}[2]\Big) = \sigma_i(j) \right\} \end{aligned}$$

The permutation followed by blinding is done by the player in order to prevent the DCV guessing the position of the card. In this way even if the DCV and the BVV colluded also they cant guess the sequence of the final shuffled cards. In order to blind the deck, $\sigma_i(R_i)$ is multiplied with the corresponding public point of the player. i.e. P_i and is represented as $E_{P_i}\Big(\sigma_i(R_i)\Big)$.

$$E_{P_i}\Big(\sigma_i(R_i)\Big) = \left\{ \; \sigma_i(R_{ij}) * P_i \middle| 1 \leq j \leq Z \; \right\} \tag{1}$$

In order to ensure the non-repudiation, the player signs $E_{P_i}\Big(\sigma_i(R_i)\Big)$ and publish these values. The authenticated initial deck of cards of the i^{th} player is represented as A_i.

$$\begin{aligned} A_i &= \left\{ sign\Big(E_{P_i}\Big(\sigma_i(R_i)\Big), p_i\Big) \right\} \\ &= \left\{ \; sign\Big(E_{P_i}\Big(\sigma_i(R_{ij})\Big), p_i\Big) \middle| \; 1 \leq j \leq Z \right\} \end{aligned} \tag{2}$$

The authenticated vector generated by the player is the set of signatures over each and every blinded card that is submitted to DCV. These authenticated vectors can be used by any entity in the game to verify the claims made by the corresponding player.

The values published by an i^{th} player are:

$$\Big(E_{P_i}\Big(\sigma_i(R_i)\Big), A_i\Big)$$

2) At Deck Creating Vendor: Let d be a 32 byte random scalar value that is private to the DCV and is multiplied with G to get the public point D on the curve and that represents the identity of DCV.
$i.e.$

$$D = d * G$$

DCV receives $E_{P_i}\Big(\sigma_i(R_i)\Big)$ shuffled, blinded cards along with the authenticated vector A_i from each player, where $1 \leq i \leq Z$. Now DCV has the set of all the initial shuffled, blinded cards of all the players, $i.e.$ R.
$Where$

$$R = \left\{ E_{P_i}\Big(\; \sigma_i(R_i)\Big) \middle| 1 \leq i \leq N \; \right\}$$

For each card of every player, DCV generates a 32 byte random numbers and the set of all such random numbers are represented as c and the random numbers specific to an i^{th} player are represented as c_i.

$$c_i = c_{ij} \mid 1 \leq j \leq Z$$

and multiplies c with G to get the public points on the curve and which further used to blind the card deck created by DCV.
$Where$

$$\begin{aligned} C &= c * G \\ &= \left\{ \; c_{ij} * G \middle| 1 \leq i \leq N, 1 \leq j \leq Z \; \right\} \end{aligned}$$

The $X/Z(XoverZ)$ values of C, computed for an i^{th} player is

$$\Big(X_{C_i}/Z_{C_i} \Big) = \Big\{ \Big(X_{\left(c_{ij}*G\right)} \Big/ Z_{\left(c_{ij}*G\right)} \Big) \Big| 1 \leq j \leq z \Big\}$$

For each player i and for each card j, DCV multiplies each card in $E_{P_i}\Big(\sigma_i\big(R_i\big)\Big)$ with the corresponding 32 byte random value which is specifically generated for that card of the player, $i.e.$ c_{ij}

$$\begin{aligned} E_i &= c_i * E_{P_i}\Big(\sigma_i\big(R_i\big)\Big) \\ &= \Big\{ c_{ij} * \sigma_i\big(R_{ij}\big) * P_i \big| 1 \leq j \leq Z \Big\} \end{aligned}$$

Encrypted deck for the player i is represented as E_i and (X/Z) of E_i is represented as

$$\begin{aligned} E_{i_{(X/Z)}} &= \Big(X_{E_i}/Z_{E_i} \Big) \\ &= \Bigg\{ \Bigg(X_{\Big(c_{ij}*\sigma_i\big(R_{ij}\big)*P_i\Big)} \Bigg/ Z_{\Big(c_{ij}*\sigma_i\big(R_{ij}\big)*P_i\Big)} \Bigg) \\ &\qquad \Bigg| 1 \leq j \leq Z \Bigg\} \end{aligned}$$

Then hash of $\Big(E_{i_{\left(X/Z\right)}}\Big)$ is computed using SHA–256 is as follows:

$$\begin{aligned} &H\left[E_{i_{(X/Z)}}\right] \\ &= \Bigg\{ H\Bigg(X_{\Big(c_{ij}*\sigma_i\big(R_{ij}*P_i\big)\Big)} \Bigg/ Z_{\Big(c_{ij}*\sigma_i\big(R_{ij}*P_i\big)\Big)} \Bigg) \\ &\qquad \Bigg| 1 \leq j \leq Z \Bigg\} \end{aligned}$$

The Encrypted Deck of cards for the i^{th} player is blinded with the random scalar values and is represented as

$$\begin{aligned} ED_i &= \Big\{ c_{ij} * H\Big[\Big(X_{(c_{ij}*\sigma_i(R_{ij})*P_i)}/ \\ &\qquad Z_{(c_{ij}*\sigma_i(R_{ij})*P_i)} \Big)\Big] \Big| 1 \leq j \leq Z \Big\} \\ &= \Big\{ R_{ij}^{'} \Big| 1 \leq j \leq Z \Big\} \end{aligned}$$

Encrypting the deck with the random scalar values make it blind to the players and BVV. Now the DCV shuffles ED_i so that neither the player nor the BVV or the combination of them can able to get the sequence of the cards.

The shuffling of DCV is represented as

$$\sigma_{DCV} = \Big\{ \sigma_{DCV}(j) \big| 1 \leq j \leq Z \Big\}$$

and is only published if required at the end of game to validate the correctness of the game and to resolve the disputes.

Applying the shuffling on ED_i results to the final output of DCV

$$\begin{aligned} \sigma_{DCV}\left(ED_i\right) = \Big\{ &\sigma_{DCV}\Big(c_{ij} * H\Big[\Big(X_{(c_{ij}*\sigma_i(R_{ij})*P_i)}/ \\ &Z_{(c_{ij}*\sigma_i(R_{ij})*P_i)} \Big)\Big]\Big) \Big| 1 \leq j \leq Z \Big\} \end{aligned}$$

For simplicity,

$$\Big\{ \sigma_{DCV}\Big(c_{ij} * H\Big[\Big(X_{(c_{ij}*\sigma_i(R_{ij})*P_i)} \Big/ Z_{(c_{ij}*\sigma_i(R_{ij})*P_i)} \Big)\Big]\Big)\Big\} \tag{3}$$

represents the blinded, shuffled j^{th} card of the i^{th} player generated by the DCV and is represented as $R_{ij}^{'}$.
$\sigma_{DCV}\left(ED_i\right)$ is the raw deck that is created by DCV and is represented with R_{DCV}, the above equation can be rewritten as

$$R_{DCV} = \Big\{ R_{ij}^{'} \Big| 1 \leq i \leq N, 1 \leq j \leq Z \Big\} \tag{4}$$

Apart from R_{DCV}, the digital signing of the card random bytes (c) will be done that produces an authenticated vector A_{DCV} which ensure nobody can spoof the role of DCV.

The authenticated vector generated by DCV that corresponds to the i^{th} player is defined as A_{DCV_i}.

$$A_{DCV_i} = \Big\{ sign\Big(\sigma_{DCV}\left(ED_i\right), d\Big) \Big| 1 \leq j \leq Z \Big\}$$

The complete digitally signed deck is represented as A_{DCV}.

$$A_{DCV} = \Big\{ A_{DCV_i} | 1 \leq i \leq N \Big\}$$

A_{DCV} is a set of digital signatures of each and every card over R_{DCV}, this will be used by BVV to verify the authenticity of the deck that it received.

The authenticity of the permutation and the random values generated by the DCV also needs to be published which is used to resolve the conflicts. A SHA–256 hash is computed on the concatenated permuted values and is signed.

$$A_{DCV}^{'} = sign\Big(H\Big(||_{i=1}^{N}\sigma_{DCV}\left(i\right)\Big), d\Big)$$

$$A_{DCV}^{"} = sign\Big(H\Big(||_{i=1}^{N}||_{j=1}^{Z}c_{ij}\Big), d\Big)$$

Since the values of C are needed by the players in order to reveal the card, so the values of C is published to the players by DCV by signing them. The authenticator vector that constitutes the signatures of C is represented as A_C.

$$\begin{aligned} A_C &= \Big\{ A_{C_i} | 1 \leq i \leq N \Big\} \\ &= \Big\{ sign\Big(C_i, d\Big) \Big| 1 \leq i \leq N \Big\} \\ &= \Big\{ sign\Big(C_{ij}, d\Big) \Big| 1 \leq i \leq N, 1 \leq j \leq Z \Big\} \end{aligned}$$

The set of values that get published by DCV are:

$$\Big(R_{DCV}, C, A_{DCV}, A_{DCV}^{'}, A_{DCV}^{"}, A_C\Big)$$

3) At Blinding Value Vendor: Let b be a 32 byte random scalar value which is private to BVV and

$$B = b * G$$

be the corresponding public point of it on the $curve25519$. Once the BVV receives the raw deck R_{DCV} and a set of authenticated vectors $\left(A_{DCV}\ A^{'}_{DCV}\ A^{"}_{DCV}\right)$ from DCV it generates a set of binding values for each player.

Let BV_i be the set of blinding values generated for the i^{th} player and is represented as follows:

$$BV_i = \{\ b_{ij} | 1 \leq j \leq Z\ \}$$

The complete set of blinding values which are used to blind the whole deck are:

$$BV = \{\ BV_i | 1 \leq i \leq N\ \}$$

The $M\ of\ N$ Shamir shared secret shards over the blinding values are to be computed and are represented as, i.e.

$$S(b_{ij}) = \left\{\ s_k(b_{ij}) \middle| 1 \leq k \leq N\ \right\}$$

$s_k(b_{ij})$ represents the k^{th} share of the j^{th} card of the i^{th} player.s_k represents the set of all k^{th} shares of the cards which are intended for the k^{th} player and is represented as:

$$s_k = \left\{\ s_k(b_{ij}) \middle| 1 \leq i \leq N,\ 1 \leq j \leq Z\ \right\}$$

BVV encrypts the shamir secret keys with the corresponding public keys of the players, so that the only the intended player can get to reveal the shamir secret key when and where it is needed. Let $E_k(s_k)$ represents the the encrypted set of shamir secret keys intended for the k^{th} A player and is represented as:

$$E_k(s_k) = \left\{\ s_k\left(b_{ij}\right) * P_k \middle| 1 \leq i \leq N,\ 1 \leq j \leq Z\ \right\}$$

Shared secret key for each card is computed and it is the multiplication of the corresponding blinding value with the corresponding public point of the player on the $curve25519$. The set of shared secret keys that corresponds to encrypt the deck of the cards intended for the i^{th} player are generated as:

$$K_i = \{\ P_i * b_{ij} | 1 \leq j \leq Z\ \}$$

The total set of shared secret keys generated by BVV to encrypt the whole deck are:

$$K = \left\{\ K_i | 1 \leq i \leq N\ \right\} \tag{5}$$

The deck received from DCV will be shuffled by BVV, in order to prevent any collusion between players and DCV. The shuffling of BVV is represented as

$$\sigma_{BVV} = \left\{\ \sigma_{BVV}(j) \middle| 1 \leq j \leq Z\ \right\}$$

and is only published at the end of game to validate the correctness of the game and to resolve the disputes if any. The shuffling is applied on the raw deck received from $DCV\ i.e\ R_{DCV}$

$$\sigma_{BVV}\left(R_{DCV}\right) = \left\{\ \sigma_{BVV}(R^{'}_{ij}) \middle| 1 \leq j \leq Z\ \right\}$$

To blind the shuffling, the deck is encrypted with shared secret key computed in equation 5. The encrypted deck of cards intended for an i^{th} player are as follows:

$$\begin{aligned} R_{BVV_i} &= E_{K_{ij}}\left(\sigma_{BVV}\left(R_{DCV}\right)\right) \\ &= \left\{\left(\ \sigma_{BVV}\left(R_{DCV}\right) * K_{ij} \middle| 1 \leq j \leq Z\ \right)\right\} \end{aligned} \tag{6}$$

By replacing, $\left(\sigma_{BVV}\left(R_{DCV}\right) * K_{ij}\right)$ with $R^{"}_{ij}$ the above equation can be rewritten as

$$R_{BVV_i} = \{\ R^{"}_{ij} | 1 \leq j \leq Z\ \}$$

The complete deck of cards of all the players are represented as EB.

$$R_{BVV} = \{\ R_{BVV_i} | 1 \leq i \leq N\ \}$$

Once the shuffled encrypted deck is created by the BVV, it is digitally signed by BVV to ensure nobody can spoof the role of it. The digital fingerprint of the final shuffled encrypted deck is computed as follows:

$$\begin{aligned} A_{BVV} &= \left\{\ A_{BVV_i} | 1 \leq i \leq N\ \right\} \\ &= \left\{\ sign\left(R_{BVV_i}, b\right) \middle| 1 \leq i \leq N\ \right\} \\ &= \left\{\ sign\left(R^{"}_{ij}, b\right) \middle| 1 \leq i \leq N, 1 \leq j \leq Z\ \right\} \end{aligned}$$

The authenticity of the permutation and the blinding values generated by the BVV also needs to be signed and these values are used to verify the correctness of the game and to resolve the conflicts.

$$A^{'}_{BVV} = sign\left(H\left(||_{j=1}^{N}\sigma_{BVV}(j)\right), b\right)$$

$$A^{"}_{BVV} = sign\left(H\left(||_{i=1}^{N}||_{j=1}^{Z}b_{ij}\right), b\right)$$

Since the blinding values are needed to reveal decrypt the card, BVV should provide a mechanism to check the authenticity and correctness of the blinding values.The $Hash$ of the blinding values signed by BVV is as follows:

$$\begin{aligned} A_{BV} &= \left\{\ A_{BV_i} | 1 \leq i \leq N\ \right\} \\ &= \left\{\ sign\left(H(BV_i), b\right) \middle| 1 \leq i \leq N\ \right\} \\ &= \left\{\ sign\left(H(b_{ij}), b\right) \middle| 1 \leq i \leq N, 1 \leq j \leq Z\ \right\} \end{aligned}$$

The set of values that get published by BVV are

$$\left(R_{BVV}, A_{BVV}, A^{'}_{BVV}, A^{"}_{BVV}, A_{BV}\right)$$

4) At Player: Raw deck is the set of the values published by DCV and BVV, which are available to the player before the game get starts. The raw deck of an i^{th} player is as follows:

$$\left(R_{BVV_i}, C_i\right)$$

The authenticated vectors needed to check the deck integrity of an i^{th} player are

$$\left(A_{BVV_i}, A_{BV_i}, A_{C_i}\right)$$

The authenticated vectors which are needed for an i^{th} player to verify the authenticity of the permutations made by the DCV and BVV and to verify the correctness of the game when disputes arise are:

$$\left(A^{'}_{DCV_i}, A^{"}_{DCV_i}, A^{'}_{BVV_i}, A^{"}_{BVV_i}\right)$$

The revealing and correctness of the raw deck as the game progresses demonstrated in the later sections of this paper.

B. Handling of the Deck

The cards in the deck will get revealed as the game progresses, even if the players possess the raw deck they can't get to reveal the cards since they are blinded by the blinding values and which are split into shamir secret shares and distributed among the players, in this way the deck got locked and it only get unlocks upon the approval of M players. Where M represents the majority which usually be a minimum of $\left(N/2+1\right)$

Since initially all players have been applied it's own permutation on the deck, so the the raw deck generated by the BVV for each player have the cards in different sequences. Once all the players received the raw deck from BVV they have to undone the initial permutation applied on the raw deck, in this way all the players have the deck which has the permutation applied by only DCV and BVV. So the deck of cards that each player possesses same sequence, but none knows which card in what slot.

Lets say if its j^{th} turn of the i^{th} player, the player picks up the j^{th} card from the published deck R_{BVV_i} in sequence and follows the below procedure in order to reveal the card. Lets say player gets a card $R^{"}_{ij}$ such that

$$R^{"}_{ij} \in R_{BVV_i} \quad and$$

By substituting R_{BVV_i} from equation 6 the above equation can be rewritten as

$$\begin{aligned} R^{"}_{ij} &= E_{K_{ij}}\left(\sigma_{BVV}\left(R_{DCV_i}\right)\right) \\ &for \quad some \quad 1 \le j \le Z \\ &= E_{K_{ij}}\left(\sigma_{BVV}\left(R_{DCV_{ij}}\right)\right) \end{aligned} \tag{7}$$

Since $K_{ij} = P_i * b_{ij}$, in order to compute the K_{ij} player needs the corresponding blinding value b_{ij}. Since b_{ij} is split and distributed using $MofN$ shamir secret key sharing approach, player i requests the rest of the players, including DCV, for Shamir secret key in order to reconstruct the blinding value.

Each player encrypts the corresponding shamir shard with the i^{th} player (requested player) public key and send it to them, in this way none of the players be aware of the shamir shards except i^{th} player, it means none of the players can reconstruct the corresponding value. In case if i^{th} player failed to receive at least M shards, then in such scenario BVV steps in and the reconstruction and redistribution of the deck happens and which is mentioned in detail in the next sections.

In ideal scenario the shamir shards received by the i^{th} player are for the j^{th} card in the sequence are:

$$\left\{ \; s_k\left(b_{ij}\right) \middle| 1 \le k \le N \; \right\}$$

Using these values player i reconstruct the blinding value b_{ij} and computes K_{ij} by multiplying it with P_i. Using K_{ij} player decrypts $R^{"}_{ij}$ in equation 7 as follows:

$$\begin{aligned} D_{K_{ij}}\left(R^{"}_{ij}\right) &= D_{K_{ij}}\left(E_{K_{ij}}\left(\sigma_{BVV}\left(R_{DCV_{ij}}\right)\right)\right) \\ &= \sigma_{BVV}\left(R_{DCV_{ij}}\right) \end{aligned} \tag{8}$$

Substitute equation 3 and 4 in 8

$$\begin{aligned} D_{K_{ij}}\left(R^{"}_{ij}\right) &= \sigma_{BVV}\left(R_{DCV_{ij}}\right) \\ &= \sigma_{BVV}\left(R^{'}_{ij}\right) \\ &= \Bigg\{\sigma_{BVV}\Bigg(\sigma_{DCV}\Bigg(c_{ij}* \\ &\quad H\Big[\left(X_{(c_{ij}*\sigma_i(R_{ij})*P_i)} \Big/ Z_{(c_{ij}*\sigma_i(R_{ij})*P_i)}\right)\Big]\Bigg)\Bigg)\Bigg\} \end{aligned} \tag{9}$$

Since the i^{th} player have the knowledge of C_i,p_i and R_{ij}. The following set of values $(say\ \alpha_i)$ is computed:

$$\begin{aligned} \alpha_i &= \Bigg\{H\Bigg[\Bigg(X_{\left(c_{ij'}*\sigma_i\left(R_{ij'}\right)*P_i\right)} \Big/ \\ &\quad Z_{\left(c_{ij'}*\sigma_i\left(R_{ij'}\right)*P_i\right)}\Bigg)\Bigg]^{-1} for some 1 \le j^{'} \le Z\Bigg\} \\ &= \left\{ \; \alpha_{ij'} \middle| 1 \le j^{'} \le Z \; \right\} \end{aligned}$$

α_i is multiplied with $D_{K_{ij}}\left(R^{"}_{ij}\right)$ in 9 and the result is

represented as $c_i^{'}$.

$$
\begin{aligned}
c_i^{'} &= D_{K_{ij}}\left(R_{ij}^{"}\right) * \alpha_i \\
&= \Bigg\{ \sigma_{BVV}\Bigg(\sigma_{DCV}\Bigg(c_{ij} * \\
&H\Big[\Big(X_{(c_{ij}*\sigma_i(R_{ij})*P_i)} \Big/ Z_{(c_{ij}*\sigma_i(R_{ij})*P_i)} \Big) \Big] \Bigg) \Bigg) \Bigg\} * \\
&\Bigg\{ H\Bigg[\Bigg(X_{\left(c_{ij'}*\sigma_i\left(R_{ij'}\right)*P_i\right)} \Bigg/ \\
&Z_{\left(c_{ij'}*\sigma_i\left(R_{ij'}\right)*P_i\right)} \Bigg) \Bigg]^{-1} for some 1 \leq j^{'} \leq Z \Bigg\}
\end{aligned}
\tag{10}
$$

Multiplying $c_i^{'}$ with G gives

$$C_i^{'} = c_i^{'} * G$$

Compare $C_i^{'}$ with C^i and

$$
\begin{aligned}
&for \quad some \quad j = j^{'}, where \quad 1 \leq j \leq Z \\
&H\Bigg[\Bigg(X_{\left(c_{ij'}*\sigma_i\left(R_{ij}\right)*P_i\right)} \Bigg/ Z_{\left(c_{ij'}*\sigma_i\left(R_{ij}\right)*P_i\right)} \Bigg) \Bigg] * \\
&H\Bigg[\Bigg(X_{\left(c_{ij}*\sigma_i\left(R_{ij}\right)*P_i\right)} \Bigg/ Z_{\left(c_{ij}*\sigma_i\left(R_{ij}\right)*P_i\right)} \Bigg) \Bigg]^{-1} = 1
\end{aligned}
\tag{11}
$$

and if we found a match in C_i and for that $j = j^{'}$ the Hash and inverse of the Hash gets nullified in equation 10 and we get $c_{ij}^{'}$ as

$$c_{ij}^{'} = \Big(\sigma_{BVV}\Big(\sigma_{DVV}\big(j\big) \Big) \Big) \tag{12}$$

From equation 12 it is clear that, the card in the sequence is the combined shuffled result of DCV and BVV.

C. Non Repudiation and Correctness of the Game

PANGEA provides the provision to authenticate and validate every move of the game in order to ensure the fair play of the game.

The authenticated vectors of all the participating entities are published among the entities to resolve any disputes and to validate the published deck of values of the corresponding entities.
Each player possess the following authenticated vectors, from different entities in the game.
From players:

$$\Big\{ A_i | 1 \leq i \leq N \Big\}$$

From DCV:

$$\Big(A_{DCV}, A_{DCV}^{'}, A_{DCV}^{"}, A_C \Big)$$

From BVV:

$$\Big(A_{BVV}, A_{BVV}^{'}, A_{BVV}^{"}, A_{BV} \Big)$$

As the game finishes every player reveals the cards received by revealing the initial 32 byte random numbers which are associated with them, where the second byte of that 32 byte random number represents the card index value. Any player in the game can verify the revealed cards of any other player and DCV, which act as a dealer, verifies the revealed cards and evaluates the game.

Since all the entities in the game possesses the authenticator vectors, so any player can verify and ensure the fair play of the game.
Lets say i^{th} player is claiming it received j^{th} card in k^{th} turn, i.e. card of index j. In order to prove the claim i^{th} player publishes R_{ij}.

The verifying entity, does the following steps to verify the correctness of the card.

1) Check $R_{ij}[2] = j$, to ensure that the claimed card index is correct.
2) Verifies the signature of the card R_{ij} against the values present in A_i and if this is successful it ensures that the card is valid.
3) In order to verify the position of the card, first the authenticity of the permutations done DCV and BVV should needs to be verified. Since at the end of the game $\sigma_{DCV}, \sigma_{BVV}$ are published, the authenticity of them will be verified against the corresponding signatures $A_{DCV}^{'}$ and $A_{BVV}^{'}$.
4) Apply the inverse permutation $\sigma_{DCV}, \sigma_{BVV}$ on $k\big(turn\big)$ to get to initial published authenticated vector of that card in A_i $i.e.$
$$s = A_i\Big[\sigma_{DCV}^{-1}\Big(\sigma_{BVV}^{-1}\big(k\big) \Big) \Big]$$
5) Verifies the signature of the card R_{ij} against the signature s identified above and if this successful both the card and position of the card in the deck are valid otherwise not.

The authenticator vectors A_C and A_{BV} are used during the progress of the card if any player failed to reveal the card from the raw deck.

D. Funds Handling

Each hand (re)distributes the total funds the table has put into chips according to the result of each game. Under normal conditions, the majority of players sign an $M\ of\ N$ multisig transaction to release the funds at the conclusion of a table. However, if more than $N - M$ players refuse to sign, then the funds would not be distributed correctly, they would be stuck. In the event where more than $N - M$ players end up without any chips, they have no financial incentive to stay online to approve the $M\ of\ N$ multisig transaction.

One way to mitigate this is to have an $M\,of\,N$ transaction signed after each round, so at most the result of a single game is unaccounted for. However, we still have the case of there being nobody left to cosign with the last man standing from financial self-interest. Also, it is common for online game players to simply disconnect out of frustration (admit it, you have done this too!), so relying on the losing players to approve a payout to the winner is not likely to be reliable. Even if the $M\ of\ N$ values are reduced as the number of players at the

table is reduced, this can be avoided as we certainly do not want to get to a $1\,of\,2$ multisig for obvious reasons.

The ideal solution is to have a blockchain-enforced payout. This requires each change of gamestate to be recorded in the blockchain and the blockchain to be able to determine the proper allocation of funds. The method of doing a $M\ of\ N$ after each game will reduce the blockchain bloat as only a game that doesnt have sufficient signers needs to be blockchain interpreted. The optimal scenario is to use bi-directional multisig payment channels for normal play and have a backup blockchain mechanism that can be invoked by any single player in the event of the funds being stuck due to not enough signers.

E. Privacy Handling

A totally separate issue is privacy, which in the case of online games is needed as some govts impose juridical restrictions on these and made online gaming illegal. This has nothing to do with hundreds of millions of dollars in campaign financing and other funds paid by Las Vegas casinos to politicians. I am assured that it is purely to protect the innocents from, well, not sure what the online players are being protected from. So, I think a way to play privately is quite important. To that effect, the psock capability allows a single node to publish an IP address and if that node is not playing, but just participating in the creation of the card deck, it will allow all the other players to play in realtime without posting their IP address. By using JUMBLR secret funds to purchase chips, the identity of the source of the funds is not linked. If you are in a totalitarian regime that is monitoring your IP traffic, then unfortunately you would need to take further protective actions, ie. dont play from any IP address that can be correlated to you. As can be seen from the above, decentralized card games are one of the most difficult challenges in the crypto world and was unsolved, until PANGEA was released. The critical tech is divided between the card deck handling and the funds handling and each will be described independently.

F. Deck Recovery

PANGEA is designed on $MofN$ trust, it means that at least M Shamir shards are required to reveal any card, otherwise game halts. The deck creation process is so independent and agile that, except the BVV none of the participating entities will have to bear the computing load while reconstructing the deck.

In the proposed approach reconstruction of the partial deck takes place without compromising any of the security constraints. In the reconstruction of the deck a new value of $M\,say\,M^{'}$ is chosen such that $M^{'} = (N^{'}/2) + 2$. Where $N^{'}/2 + 1$ shards represents the majority, and plus one is added since DCV also holds a Shamir shard.

In most of the scenarios the players who get disconnected are either losers or who get disconnected abruptly due to power failure or loss of cable connection. Once the BVV identifies the reconstruction needs to be done, it finds out δ where δ is the number of players get disconnected and $\big(N - \delta\big)$ gives number of active players in the game.
If

$$\delta < (N - M + 1)$$

then in such scenario reconstruction of the deck should happen in order to continue the game.

BVV has R_{DCV} received from the DCV and a new set of blinding values will be created for all the players who are present in the game, i.e.

$$BV_{i}^{'}\ \forall\ i = 1\ to\ N^{'}$$

where $N^{'}$ represents the active players present in the game.While in reconstruction of the deck $M^{'}$ is computed as

$$M^{'} = \Big(\big(N - \delta\big)/2\Big) + 2$$

In the process of recovery of the deck, in order to maintain the state of the game the permutation applied remains unchanged.

The $MofN$ Shamir shards computed for the newly generated blinding values and are represented as, i.e.

$$S\left(b_{ij}^{'}\right) = \left\{\ s_k\big(b_{ij}^{'}\big)\Big|1 \leq k \leq \big(N - \delta + 1\big)\ \right\}$$

The Shamir shards are encrypted with the corresponding player public key and are distributed to the Players securely and DCV.

Here $b_{ij}^{'}$ represents the reconstructed deck blinding value associated with the j^{th} card of the i^{th} player. The encrypted Shamir shards of b_{ij} is represented as $E\left(b_{ij}^{'}\right)$.

$$E\left(b_{ij}^{'}\right) = \begin{cases} E_{P_k}\Big(s_k\big(b_{ij}^{'}\big)\Big)\ where\ 1 \leq k \leq (N - \delta) \\ E_D\Big(s_k\big(b_{ij}^{'}\big)\Big)\ where\ k = (N - \delta + 1) \end{cases}$$

A Shared secret key for each card is computed and it is the multiplication of the corresponding blinding value with the public point of the player on the $curve25519$. For instance, the shared secret keys which are generated to encrypt the cards correspond to the i^{th} player is represented as:

$$K_{i}^{'} = \left\{\ P_i * b_{ij}^{'}\Big|1 \leq j \leq Z\ \right\}$$

Since the reconstruction of the deck needs to preserve the state of the game, for that while reconstruction, the deck should be shuffled with an initially generated permuted values $i.e\ \sigma_{BVV}$

$$\sigma_{BVV}\Big(R_{DCV}\Big) = \left\{\ \sigma_{BVV}\Big(R_{ij}^{'}\Big)\Big|1 \leq j \leq Z\ \right\}$$

The shuffled deck is encrypted with the shared secret key computed, i.e. $E_{K_i}\Big(\sigma_{BVV}\Big(R_{DCV}\Big)\Big)$ and is represented as $R_{BVV_i}^{'}$.

$$\begin{aligned} R_{BVV_i}^{'} &= E_{K_i^{'}}\Big(\sigma_{BVV}\Big(R_{DCV}\Big)\Big) \\ &= E_{K_{ij}^{'}}\left(\left\{\ \sigma_{BVV}\Big(R_{ij}^{'}\Big)\Big|1 \leq j \leq Z\ \right\}\right) \\ &= \left\{\ R_{ij}^{"}\big|1 \leq j \leq Z\ \right\} \end{aligned}$$

Encryption and shuffling of the deck prevents the DCV to guess the sequence of the cards that each player gets. The

newly published raw deck of all the players is represented as $R^{'}_{BVV}$.

$$R^{'}_{BVV} = \left\{ R^{'}_{BVV_i} \middle| 1 \leq i \leq (N - \delta) \right\}$$

Once the shuffled encrypted deck is created by the BVV, it also digitally signs the blinding values.Since the permutation applied by BVV remains unchanged so $A^{'}_{BVV}$ vector remains unchanged and all the remaining authenticated vectors will be recomputed.

The digital fingerprint of newly constructed raw deck is

$$\begin{aligned} A_{BVV} &= \left\{ A_{BVV_i} | 1 \leq i \leq N \right\} \\ &= \left\{ sign\Big(R_{BVV^{'}_i}, b\Big) \middle| 1 \leq i \leq N \right\} \\ &= \left\{ sign\Big(R^{"}_{ij}, b\Big) \middle| 1 \leq i \leq N, 1 \leq j \leq Z \right\} \end{aligned}$$

Combined authenticator vector over a newly generated blinding values is

$$A^{"}_{BVV} = sign\Big(H\Big(||_{i=1}^{N}||_{j=1}^{Z} b^{'}_{ij}\Big), b\Big)$$

Since the blinding values are needed to reveal decrypt the card, BVV should provide a mechanism to check the authenticity and correctness of the blinding values. The $Hash$ of the blinding values signed by BVV is as follows:

$$\begin{aligned} A_{BV} &= \left\{ A_{BV^{'}_i} \middle| 1 \leq i \leq N \right\} \\ &= \left\{ sign\Big(H\big(BV^{'}_i\big), b\Big) \middle| 1 \leq i \leq N \right\} \\ &= \left\{ sign\Big(H\big(b^{'}_{ij}\big), b\Big) \middle| 1 \leq i \leq N, 1 \leq j \leq Z \right\} \end{aligned}$$

The newly reconstructed raw deck along with newly generated authenticator vectors is as follows:

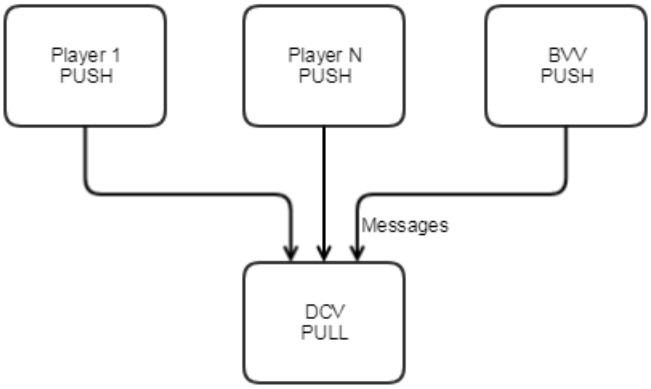
$$\Big(R^{'}_{BVV}, A_{BVV}, A^{'}_{BVV}, A^{"}_{BVV}, A_{BV}\Big)$$

Since the newly generated deck preserves the sequence of the cards in the original sequence, once it get distributed to the players the play resumes from the point where it is halted.

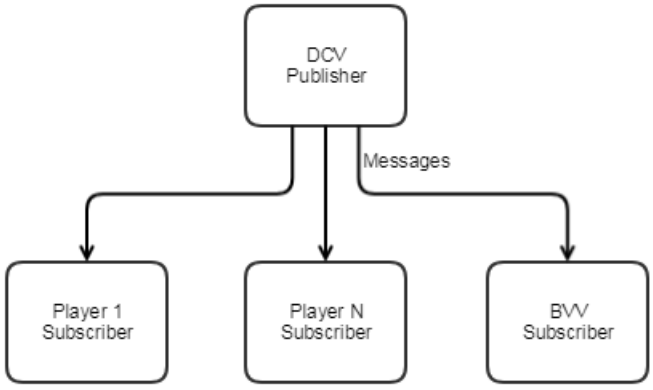
III. Implementation

A. Message Communication

All the communications in the game must happen through DCV, PANGEA protocol doesn't allow any direct communication between the Players and the BVV. Players and BVV connect to DCV via NN_PUSH/NN_PULL socket, if any entity in the game is willing to send a message, it sends it to DCV via NN_PUSH, and DCV receives it via NN_PULL.



Once the DCV receives the messages it publishes it via NN_PUB and since Players and BVV are subscribed to DCV via NN_SUB so whenever the DCV publishes the messages the Players and BVV receives it.



B. Coding Implementation

The whole project is written in C and the source code is available at [8] and [9].

CHIPS is a bitcoin fork, use of lightening protocol to get the consensus of the block chain make the transactions in the game real time. CHIPS uses $curve25519$ for faster response. In order to understand the cryptomath PANGEA does, it is required to understand a bit of the curve25519 internals.

C code to generate a public point on the curve for a given random scalar value

```
bits256 curve25519(bits256 r, bits256 G)
{
    bits320 bp,x,z;
    r.bytes[0] &= 0xf8,
    r.bytes[31] &= 0x7f,
    r.bytes[31] |= 0x40;
    bp = fexpand(G);
    cmult(&x,&z,r,bp);
    return(fcontract(fmul(x,crecip(z))));
}
```

The above is the fundamental curve25519 operation that takes a 256 bit scalar and a compressed field element. Few bits of random scalar value is set and cleared to meet the $curve25519$ dynamics. Then the compressed field element is expanded, a curve multiplication is done to create X and Z values. Finally X/Z is calculated by doing a field multiplication between X and the reciprocal of Z and this field element is compressed.

The C code for a 32bit CPU compatible equivalent function, where the base point(G) is hard coded to the generator $\{9, 0, 0, 0, 0, 0, 0, 0\}$ is This creates the public key result out of the private key r.

```
bits256 xoverz_donna(bits256 r)
{
    limb x[10],zmone[10],z[10],bp[10],out[11];
    bits256 R,G;
    memset(G.bytes,0,sizeof(G));
    G.bytes[0] = 9;
    fexpand32(bp,G.bytes);
    cmult32(x,z,a.bytes,bp);
    crecip32(zmone,z);
    fmul32(out,x,zmone);
    fcontract32(R.bytes,out);
    return(R);
}
```

The key mathematical aspect that is utilized is that a $curve25519$ public key is the field division of the X and Z coordinates of the point on the curve, i.e X/Z. The assumption is that calculating the reverse is mathematically hard, i.e. going from the value of X/Z to find the individual X and Z elements requires a lot of brute force (or a Quantum Computer from the future). The above assumption is what the curve25519 encryption is based on, so it is a safe assumption that it is valid. PANGEA makes a further assumption that Z/X is equally mathematically hard.

A simple proof that this is true is as follows: If Z/X is not mathematically hard, we can calculate the reciprocal of X/Z to get Z/X and solve curve25519 public keys. Since the reciprocal that converts between X/Z and Z/X does not change the curve25519 from being mathematically hard, it follows that we could use Z/X form and have the same security level.

We also know that a curve25519 shared secret provides a secure way for two independent key pairs to communicate with each other.

$$curve25519(privA, pubB) == curve25519(privB, pubA)$$
$$X/Z.privA * pubB == X/Z.privB * pubA$$

We will designate specially selected key pairs as cards, we will require that the second byte of the private key is the card index 0 to 51. To create a deck, we need 52 private keys such that no two have the same second byte. Further, we will designate players by their public keys (which should be a session based key pair). This allows encoding each card for each player. Essentially each "card" is a vector of field elements with the special property that only the designated player can decode the private key to determine what the second byte is.

The C code for player and cards key pair generation is below:

```
bits256 card_init(int32_t
privkeyflag,int8_t index)
{
    bits256 randval;
    OS_randombytes(randval.bytes,
sizeof(randval));
    if ( privkeyflag != 0 )
        randval.bytes[0] &=
0xf8, randval.bytes[31] &= 0x7f,
randval.bytes[31] |= 0x40;

    randval.bytes[30] = index;
    return(randval);
}

struct pair256 deck_init(struct pair256
*cards,int32_t numcards)
{
    int32_t i; struct pair256
player_keypair,tmp;
    player_keypair.priv=
curve25519_keypair(&player_keypair.prod);
    for (i=0; i<numcards; i++) {
        tmp.priv = card_init(1,i);
        tmp.prod = curve25519(tmp.priv,
curve25519_basepoint9());
        cards[i] = tmp;
    }
    return(player_keypair);
}
```

The messages exchanged between the entities are encoded in JSON format, and each message is encoded/decoded based on the predefined messages format structure defined in the game. Below we provided the messages exchanges during the initialization of the deck, when the number of players is two. When the players join the table, the player deck initialization is done. It sends the $init\_p$ message to DCV.

Below are the deck initialization messages that are calculated as per derived in the equation 1

At Player1:

```
{
"messageid": "init_p",
"playerid": 0,
"range": 2,
"publickey":
"46285eaa84e641b1c4489df2fab5d72b
da1bdf84dd1503418aa6e572a3fbf738",
"playercards":
["738564a588c9df918f8cdf578c06d4f
3385c9f8a1950dc1d1c03cf25d870dc59",
"897bdfa5c667b0e7c19ffe2e40e7e91d
01614661cd85fe9caf28577e0edfd610"]
}
```

At Player2:

```
{
"messageid": "init_p",
"playerid": 1,
"range": 2,
"publickey":
"0f3573bcb687482c9e2da620615f0a46
a89fdd4b1ead0eb18b2718c284f81172",
"playercards":
["8defa7ba654c6cd611c4c214a2dbd45
b1d0c14aff1638bd49cf288b6ace8822f",
"02ff2dd130c0fe861f8d622c12cd1591
e820a37cf96277b2baf2dde4a87efd45"]
}
```


$init\_d$ is the messageid for DCV deck initialization message, like as derived in the equation 4 the outcome of the deck initialization by DCV consists of $\left(C, R_{DCV}\right)$ At DCV:


```
{
"messageid": "init_d",
"deckid":
"b0ccd2d5859ae1eebdf3549e2eb1b919
4eedd7e943d169dbe8ea736eef5775fd",
"C":
["1a3eaf4f043077eefe4f0b5746bf072
bd360a51f8376b7593893e34451162e6b",
"82428513a920fd68b93045f8b8dd3bd2
9d0d518056b886db8c068e279f96b649",
"0446251cb7baf6a1626507eeb55b049e
6159b7e6aac9e19d7cdf62d4ad031974",
"0cd729895b494fffe7485e3238ac92e1
31783954321f76200a311e253af2850f"],
"R_DCV":
["e4f6576f707dd283586e6f19234490b
661730e50db86e11c6e899eac1246f648",
"07aff04cef5432914b46d3ca8bd7c70
c66d5353b763464d0761f48244600cb37",
"189cb13c24e4ffa22e773314bc5e9cc
fc604e9379bb4aae51ac0c68201886106",
"916b847c4502bb5c2638082677a44f40
8514428844cd126a3f4d369d1f2e464e"],
}
```


$init\_b$ is the messageid for BVV deck initialization message, like as derived in the equation 6 the outcome of the deck initialization by BVV is the raw deck R_{BVV} At BVV:


```
{
"messageid": "init_b",
"publickey_b":
"5a422ec139da8305a59b20082b96840b
ef4fa8ae20de29b18b961d25f2df4c5e",
"R_BVV":
["bae4616651f8ef5be1290ac22d2ead3
9324492d434658973b767ce30f79e5a49",
"45bb6d13d6e385ec7fe79c43ed7ee679
dc5b802a3230ab44d3dc40af60de0e73",
"e61f04822391f73e66c87dc6753fa8c0
4bdf772795f48bf988c01a77077f910c",
"409a443ac82dd6146893b0890a50b53a
12f3cadfaae7f432a973ab13fddcd276"],
}
```


IV. Conclusion

We proposed a fully decentralized peer-to-peer gaming system which doesn't rely upon any third party for trust. The association of identities to the entities with a key pair and the use of digital signatures represents a true sense of ownership and establishes the trust. We have handled the collusion between any entities by involving all the entities to participate in shuffling and then further blinding them to maintain the secrecy, and then linking the process unblinding to $M\ of\ N$ trust using Shamir shards forces the agreement of majority of the players to make any valid move.

We addressed resolving disputes using by evaluating the actions of each player recorded over the block chain. Player dis-connectivity is addressed by adjusting the threshold value M by reconstructing and redistributing the deck whenever it is needed. The use of lightning network with CHIPS makes any transactions instant, the use of JUMBLR to transact CHIPS into the game provides the anonymity to the participating entities. The proposed system outweighs the law of the land and lets everyone play the game with common rules.

References

[1] Ruffing T., Moreno-Sanchez P., Kate A. (2014) CoinShuffle: Practical Decentralized Coin Mixing for Bitcoin. In: Kutyowski M., Vaidya J. (eds) Computer Security - ESORICS 2014. ESORICS 2014. Lecture Notes in Computer Science, vol 8713. Springer, Cham

[2] D. J. Bernstein, "A state-of-the-art Diffie-Hellman function." Available: https://cr.yp.to/ecdh.html. [Accessed: Jan. 18, 2018]

[3] Adi Shamir, "How to Share a Secret." Available: http://www.cs.tau.ac.il/~bchor/Shamir.html. [Accessed: Jan. 18, 2018]

[4] Decker C., Wattenhofer R. (2015) A Fast and Scalable Payment Network with Bitcoin Duplex Micropayment Channels. In: Pelc A., Schwarzmann A. (eds) Stabilization, Safety, and Security of Distributed Systems. SSS 2015. Lecture Notes in Computer Science, vol 9212. Springer, Cham

[5] Shamir A., Rivest R.L., Adleman L.M. (1981) Mental Poker. In: Klarner D.A. (eds) The Mathematical Gardner. Springer, Boston, MA

[6] P. Golle, "Dealing cards in poker games," International Conference on Information Technology: Coding and Computing (ITCC'05) - Volume II, 2005, pp. 506-511 Vol. 1.

[7] Greg Maxwell, "Confidential Transactions." Available: https://people.xiph.org/~greg/confidential_values.txt. [Accessed: Jan. 18, 2018]

[8] jl777, "Source code for PANGEA." Available: https://github.com/jl777/lightning. [Accessed: Jan. 18, 2018]

[9] sg777, "Source code for PANGEA." Available: https://github.com/sg777/lightning. [Accessed: Jan. 18, 2018]