CryptoEscrow: An approach for safer P2P trading

LiberDApps
liberdapps@protonmail.com

Abstract. One issue that plagues peer-to-peer markets is the counterparty risk. If buyers and sellers can feel safer when trading, having assurances that not only minimize the risk of frauds but also incentivize honest behavior, peer-to-peer markets have a much higher chance to unlock their full potential. In this paper, we propose a solution to create such an environment to allow trading in a decentralized, permissionless and safer way by leveraging smart contracts and incentive mechanisms.

The issue of trust in a peer-to-peer trade

Cryptocurrencies allow borderless transactions without relying on banks or on centralized financial institutions. Although this opens up huge possibilities for peer-to-peer markets, there are issues that prevent that from happening at full potential. One of the biggest issues is counterparty risk: how can we assure that both players in a trade will honor their part in the deal? Traditionally, one of the players must give in and rely on the reputation of the other party, being exposed to risks like one simply running away with the funds or not fully meeting the terms of the deal.

Taking crypto to fiat transactions as an example, buyers are usually the ones who give in by first sending the money and then waiting for sellers to transfer the coins. On one hand, we have potential buyers that feel extremely uncomfortable with that approach and don't engage in transactions because of that; on the other side, sellers are always afraid of getting scammed and, hoping to avoid that, end up asking for a lot of personal information from buyers, making the whole negotiation cumbersome.

Smart contracts to minimize the need for trust

One of the greatest things about smart contracts is that they can minimize or even eliminate the need to trust people. By having a transparent, immutable and incorruptible set of rules, the trust is shifted towards logic, mathematics and code, which are 100% predictable and auditable, unlike human behavior. With smart contracts, we can not only provide a mechanism where dealers are incentivized to act honestly, but we can also establish ways to punish bad actors with financial losses, creating a scenario where they can't simply run away with nothing to lose.

Of course, defining what a bad actor is can still be very subjective, and a smart contract alone won't be able to spot bad behavior. However, it can still provide mechanisms, written in code, that allow other players – acting as mediators – to step in and help solve any disputes, being financially incentivized to do so. In the section below we'll dig into the details for implementing such a system.

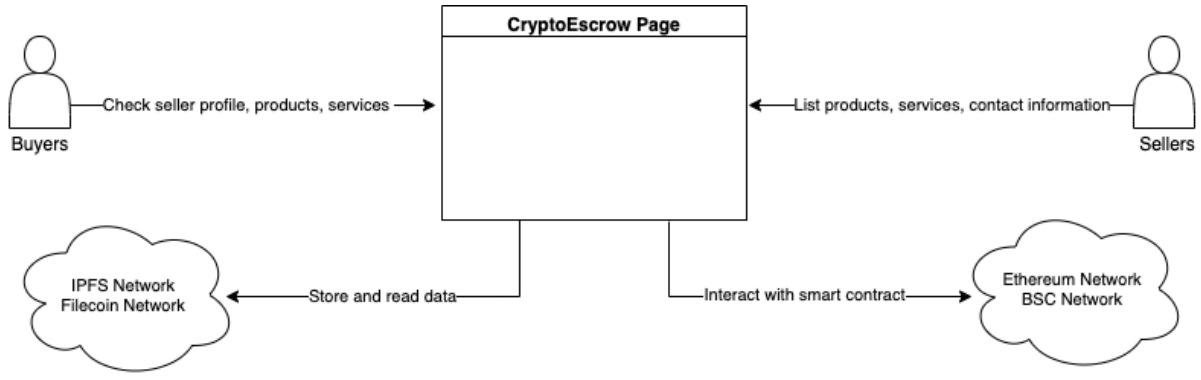
Implementing a CryptoEscrow platform

Sellers should be able to advertise their products and services, include contact information and any other relevant details in a web page. That web page should be built as a client-side app, meaning that all it takes to run it is a browser. That's needed to ensure censorship resistance: people may simply download the page and open it locally in their browser; they don't need to rely on any centralized server for accessing the app. With that said, in order to provide a friendlier experience, the page can still be hosted in a convenient and more centralized place, although not strictly necessary.

Again, to provide censorship resistance, all application data (like the seller profile and information about trades) should be stored in decentralized and distributed networks:

1. Information about products and services, contact information and any other relevant details provided by the seller should be stored in the *IPFS*[1] and *Filecoin*[2] networks. These details will be linked to the seller's wallet address, so in order to enhance privacy, sellers can set aside a particular wallet only for that purpose.
2. Transaction data, funds and trading statistics should be managed by smart contracts deployed to the *Ethereum*[3], *Binance Smart Chain (BSC)*[4] and/or any other compatible networks.

Buyers should be able to access the app, navigate through the list of products and services, check contact information and statistics (like the total number of trades, cancelled trades, etc) about sellers. The figure below summarizes that process:

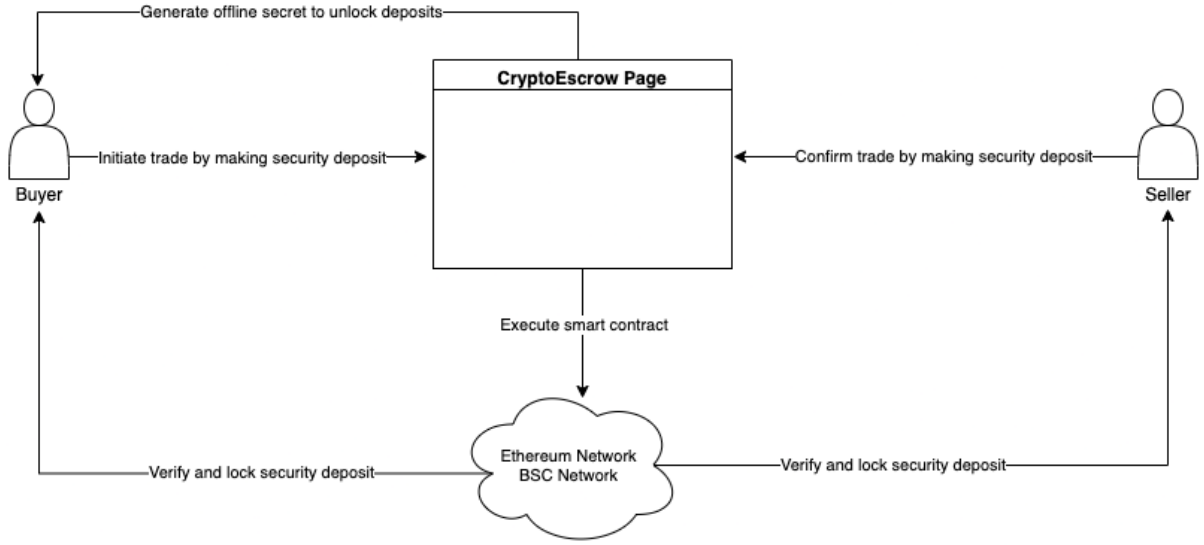


Both buyer and seller should agree on a security deposit. A security deposit is an amount in *ETH* (*Ethereum* network) or in *BNB* (*Binance Smart Chain* network) that both players must lock in a smart contract before engaging in a peer-to-peer trade. That

guarantees that both parties will have skin in the game and will be willing to trade honestly, at the risk of losing their security deposit to the other party in case of disputes. Buyer and seller will be able to freely negotiate the exact amount for the security deposit and, once that amount is agreed on, the trading process should be started like this:

1. Buyer should click on an option in the app to create a new trade with the seller, specifying the amount for the security deposit.
2. A secret code will be generated for the buyer. With that secret code, both parties will be able to unlock their security deposits after the trade is completed.
3. A smart contract will safely collect the buyer's security deposit.
4. Seller will be able to find the transaction on the web page and will have 30 minutes to make a security deposit with the exact same amount as enforced by the smart contract.
5. After the buyer and seller make their security deposits, the smart contract will effectively lock them, allowing both parties to proceed with the trade. At this point, the buyer could send money to the seller, for example.

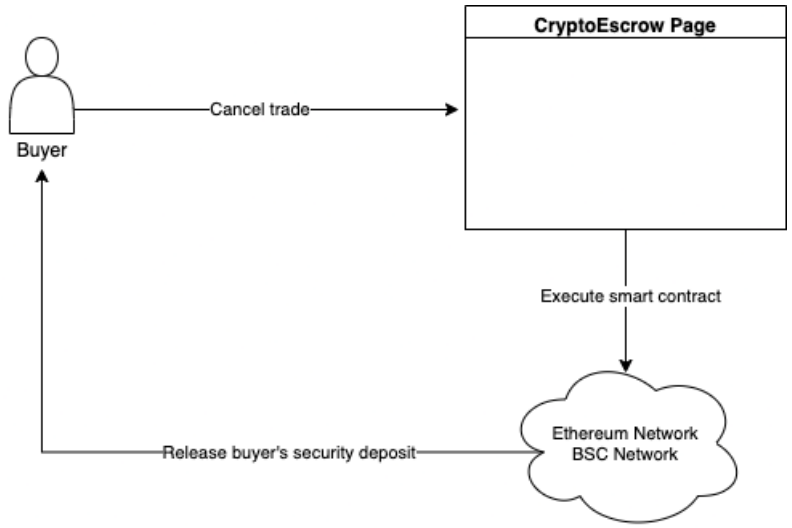
The figure below summarizes that process:



The scenario above guarantees that security deposits cannot be stolen and that both parties must provide insurance before trading. That's equivalent to building a vault for which only the buyer has the key (secret code) to open it. The buyer puts a security deposit inside the vault and waits for the seller to do the same thing. An automated agent (smart contract) will ensure that both deposits are inside the vault before effectively locking the vault and storing it in a safe place (the blockchain). From that point on, the vault can only be accessed under certain restricted rules.

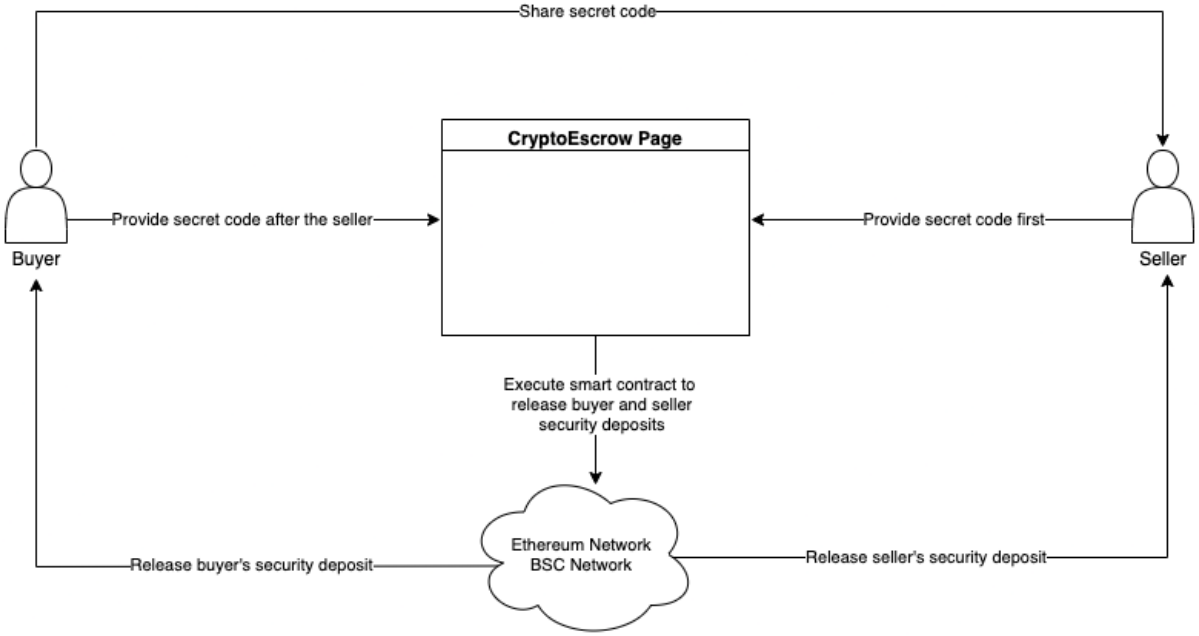
The 30-minutes window is meant to cover the case where sellers back out of deals and never make their security deposit: buyers should have a way for getting their deposits back in this scenario. In order to do so, buyers will have to cancel trades on the app page and the smart contract will be in charge of validating the time window and releasing

security deposits back to them. Since the vault hasn't been locked yet, no secret code will be needed for that operation:



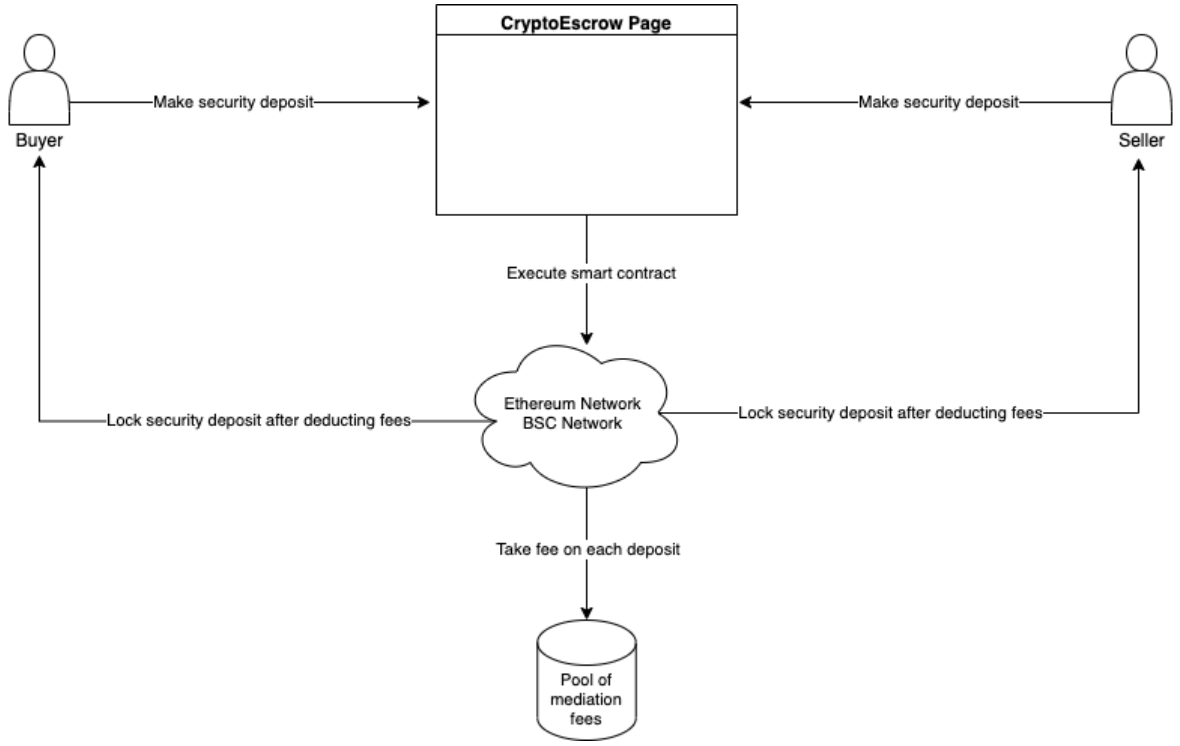
It's also worth mentioning that, when a trade is cancelled, that's accounted for in the seller's profile which might negatively affect the seller's reputation: a lot of cancelled trades might indicate that a seller is usually inactive or often backs out of deals. On the other hand, cancelling a trade requires the buyer to pay for network fees, and that's a natural anti-spam barrier: if someone is willing to create a big number of fake cancelled trades for a seller, that bad actor must dispose of money and time for that.

If trading goes well behind the scenes, the buyer should share the secret code with the seller in order to release their security deposits. The smart contract will require that the seller be the first one to withdraw the security deposit. In practice, that should only happen if the buyer gets the desired outcome in the trade and willingly shares the secret code with the seller. Once used by the seller, the smart contract will then allow the buyer to use that same code in order to also get the security deposit back. That way, we can make sure that both parties can recover their security deposits if they act honestly:



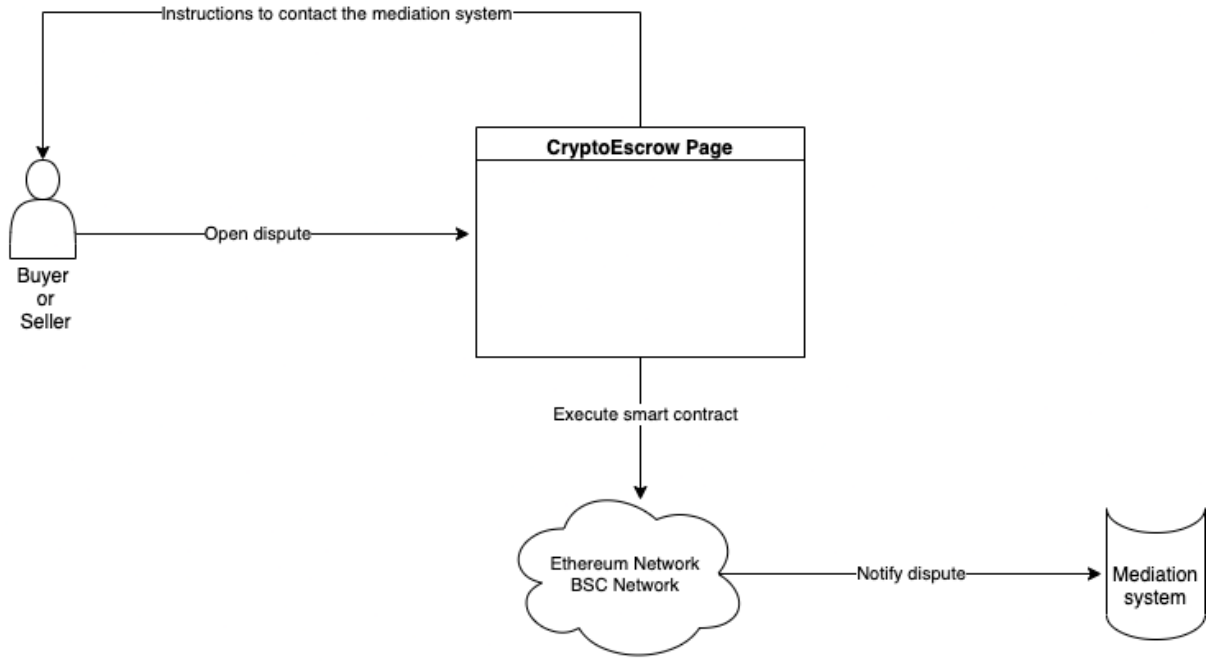
Releasing deposits using the secret code will be only possible if both parties cooperate. In the ideal scenario, that should always be the case. But what if cooperation is not possible? What if the buyer refuses to go on with the trade or if the seller, after getting buyer's resources, decides to step out of the deal? For such cases, any party can open a dispute so that mediators will step in and try to solve the issue between the parties.

To be sustainable, the system needs to financially encourage mediators to perform their job and, to minimize trust, that mechanism must be written in the smart contract. One reasonable approach is to allow for a small fee to be taken from the security deposits whenever a trade is started, maintaining those fees in a pool:



The system above automatically keeps a pool of fees that can be used to pay mediators for playing their role. The smart contract will also take those fees into account when releasing security deposits. For example, if the fee is 0.25% and both parties deposit 1 ETH each, 0.0025 ETH will be taken from the buyer's security deposit and 0.0025 ETH will be taken from the seller's deposit. When releasing security deposits, the buyer and the seller should get 0.995 ETH each – the other 0.005 ETH will be retained in the pool of fees. The smart contract should also make it so that only its owner should be able to withdraw those fees. Partnering with mediators that can contribute to the platform is in the best interest of the contract owner, and the pool of fees can be used to fund the partnerships.

When a buyer or a seller opens a dispute, the smart contract should update the transaction state and emit a notification in the blockchain so that the mediation system can know about it. The app page should also reflect that, instructing both parties to contact the mediation system by email, telegram, discord, or any other communication protocol. The figure below illustrates that:



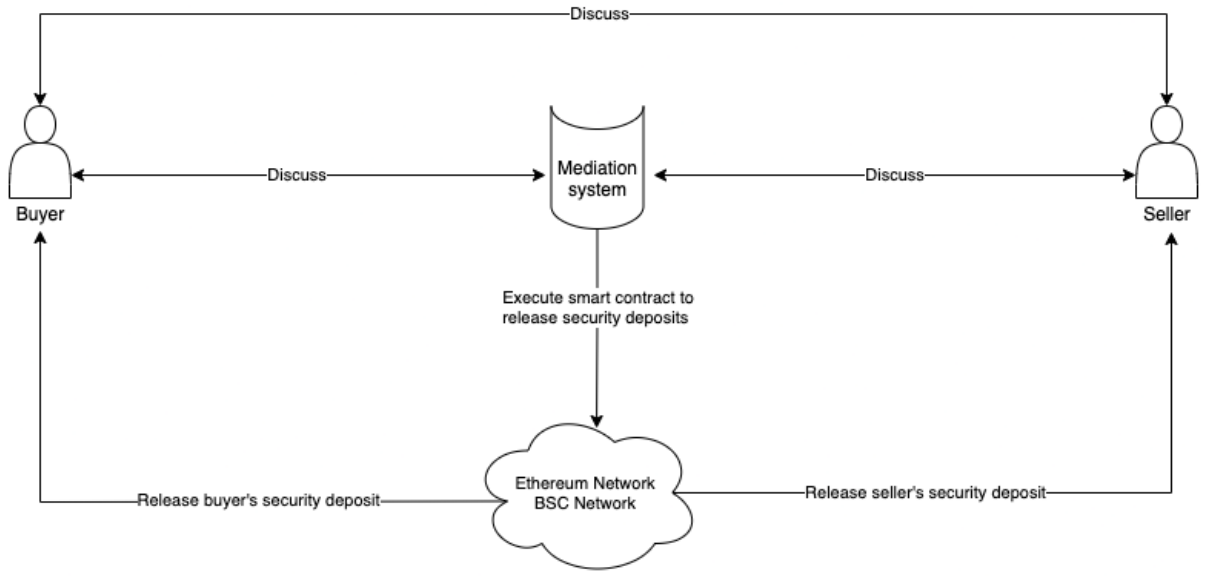
From this point on, the mediation system will work with the buyer and seller to solve the dispute in the best way possible. Mediators may require additional information like verifying the parties involved in the trade, any receipts, transaction IDs or any additional information that may assist and sort out the situation. Refusing to communicate with the mediation system may cause the offending party to lose the case and suffer penalties (losing the security deposit).

Even under a dispute, the trade can still be resolved in a friendly way if buyer and seller agree on an outcome: if that's the case, the buyer can still share the secret code with the seller to unlock their funds. In the worst case, though, mediators will have to take an action in order to resolve the dispute, and the smart contract should allow the following cases:

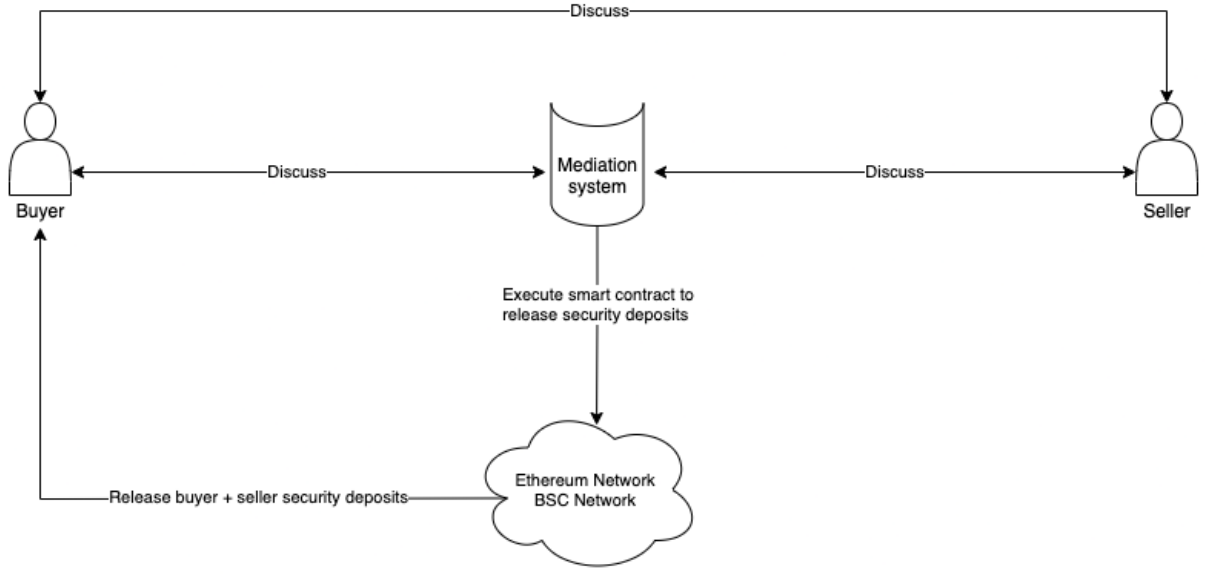
1. The buyer and the seller get their deposits back.
2. The seller was found to be a bad actor and gets penalized by losing the security deposit to the buyer.
3. The buyer was found to be a bad actor and gets penalized by losing the security deposit to the seller.

The first option is meant to cover cases where both parties ended up getting their expected outcome. It could be that the buyer just forgot the secret code and parties were unable to unlock their security deposits, requiring mediation to release them, for example. Another case is when parties had some disagreement during the trade but then decided to cooperate by fulfilling their part in the deal.

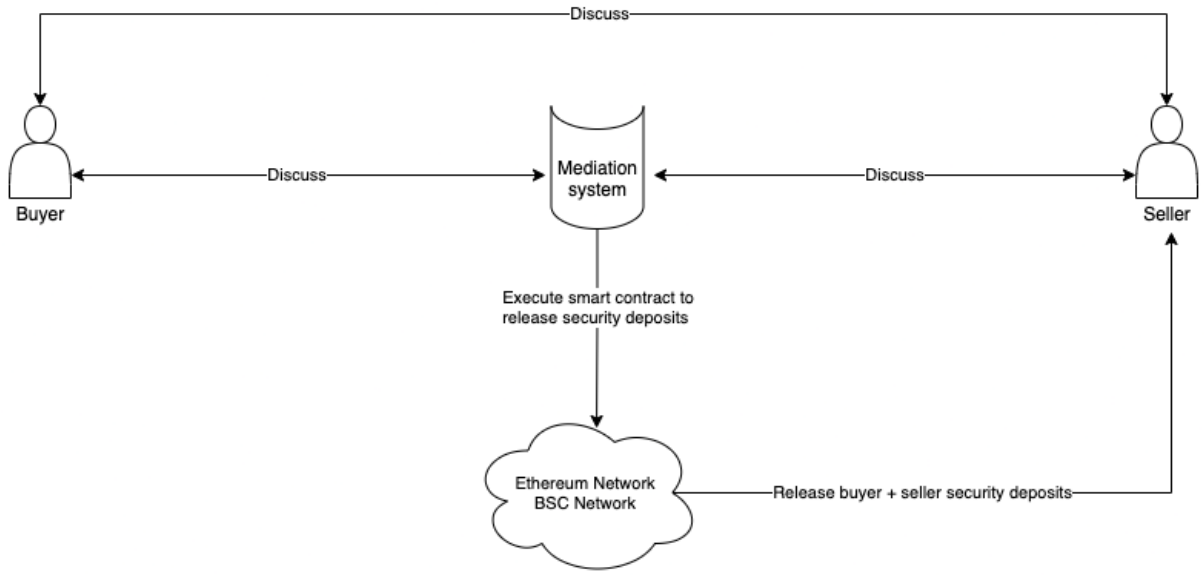
When mediators choose option #1, both buyer and seller will be able to withdraw their particular deposits (with fees deducted):



If mediators have strong evidence that the seller had the intention to harm the buyer, like refusing the transfer of assets even after being paid by the buyer, they can trigger option #2 in the smart contract. In such scenarios, mediation will allow the buyer to also withdraw the seller's security deposit (with fees deducted), as a punishment to the seller.



We can also have the opposite scenario: the buyer intentionally acted to harm the seller, like never sending the money or presenting fake receipts, for example. Similarly, mediators can trigger option #3 in the smart contract, which will allow the seller to also withdraw the buyer's security deposit, as a punishment to the buyer:



One thing to keep in mind is that, even with security deposits, bad actors might still have the incentive to act maliciously. That's particularly true if the security deposit is small enough so that even losing it to the other party outweighs the benefits that a bad actor might have. It's up to both parties to assess that risk and agree on security deposits that mitigate that risk: it is expected that, the higher the risk, the bigger the security deposits will tend to be. The platform should also establish a minimum value for security deposits so that both parties are incentivized to put skin in the game and that mediation fees are also reasonable.

Final considerations

We presented a platform that, even though doesn't completely eliminate the need for trust during a peer-to-peer transaction, greatly minimizes that need. Players are required to put skin in the game and risk losing resources if they act badly, which imposes a first barrier against scammers. By counting on mediators that are financially incentivized to perform their job, buyers and sellers also have an extra layer of security and support in case their negotiation goes wrong. Besides the right incentives, the security model is entirely governed by smart contracts, which are open, self-executable and auditable, making the whole process more predictable and deterministic.

Another big advantage of that system is that, unlike other centralized escrow solutions, it doesn't have a single point of failure. It doesn't rely on a centralized infrastructure that can be attacked or even seized by regulations and state organizations, for example. All funds are managed by smart contracts deployed to decentralized and distributed blockchains, which makes the platform more robust. That all combined allows for a decentralized, permissionless and censorship resistant solution for safer peer-to-peer trading.

Finally, this protocol can be also extended to other blockchain networks with support for smart contracts, potentially expanding the user base that can benefit from it and

providing enhancements like cheaper network fees and/or faster transaction speeds, for example.

References

[1] *IPFS* - https://ipfs.io
[2] *Filecoin* - https://filecoin.io
[3] *Ethereum* - *https://ethereum.org*
[4] *Binance Smart Chain* - https://www.binance.org/en/smartChain