multisignature wallets tech spike

Zhao Loong Ng

270121

Problem statement:

- very important document that requires 2 or more parties to authorise
- parties will sign document
- document is sent to bank

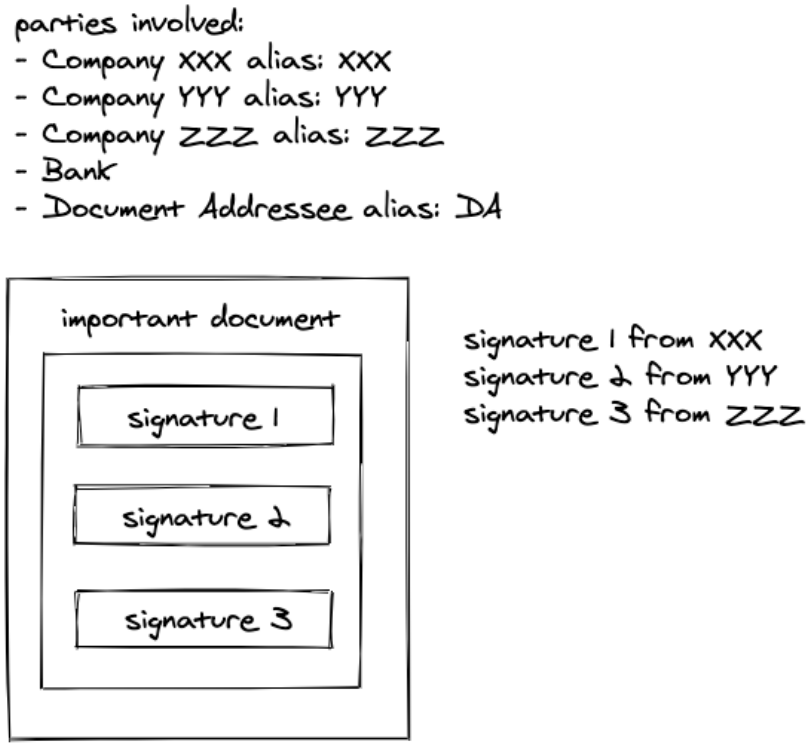

Figure 1: stage

Happy flow:

- bank verifies that all signatures from the participating parties are valid
- proceeds on with the transaction
- party at the end of the transaction benefits and is happy.
- All parties are happy

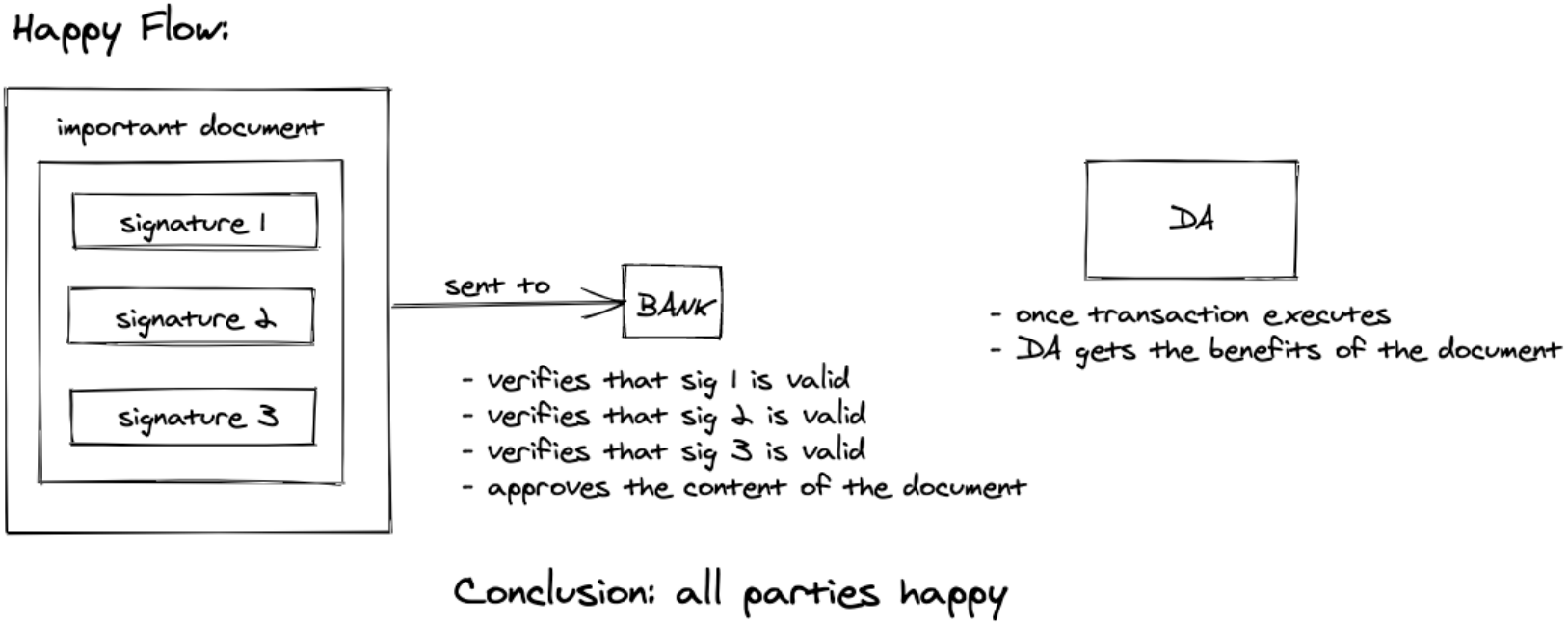

Figure 2: happy-flow

Unhappy flow 1(pls correct me on this):

- somehow bank does not do its due diligence and still proceeds on with transaction
- transaction has caused a problem and the party who received the bad circumstances due to the transaction wants to sue the bank
- since bank did not do their due diligence, thus bank is responsible

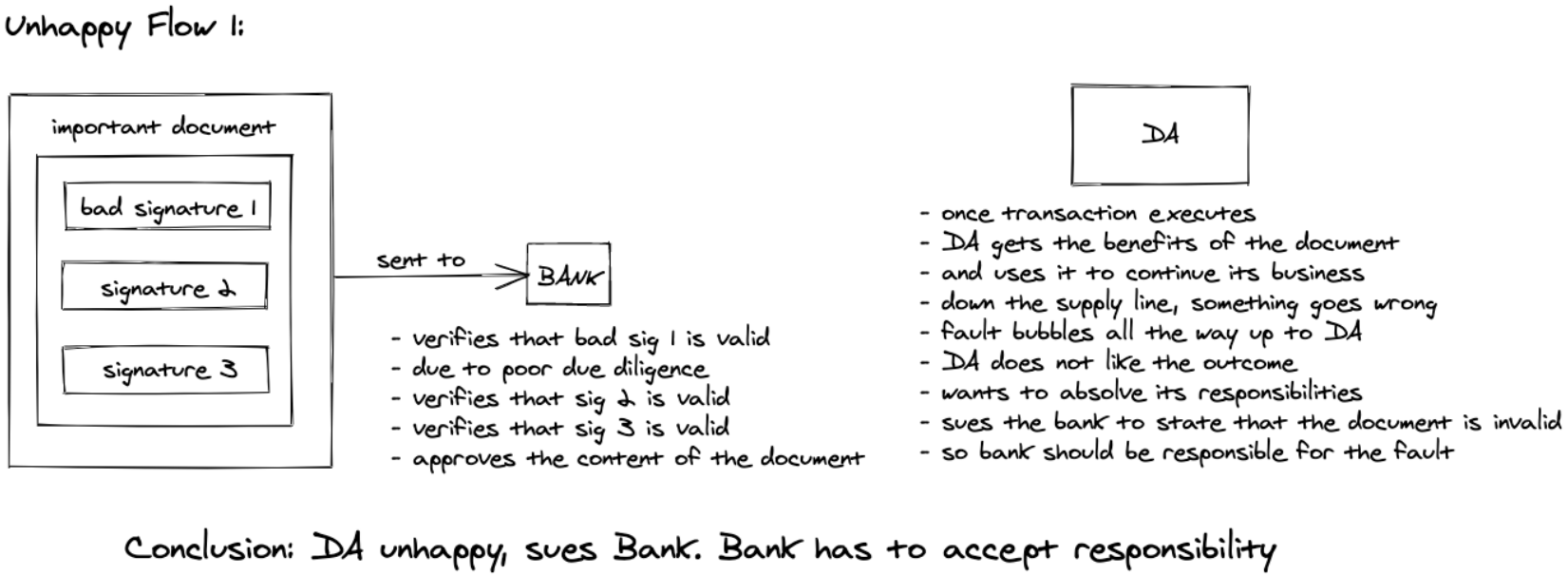

Figure 3: unhappy-flow1

Unhappy flow 2:

- bank notes that one of the parties's signatures is suspicious
- does not proceed with transaction
- transaction is discarded

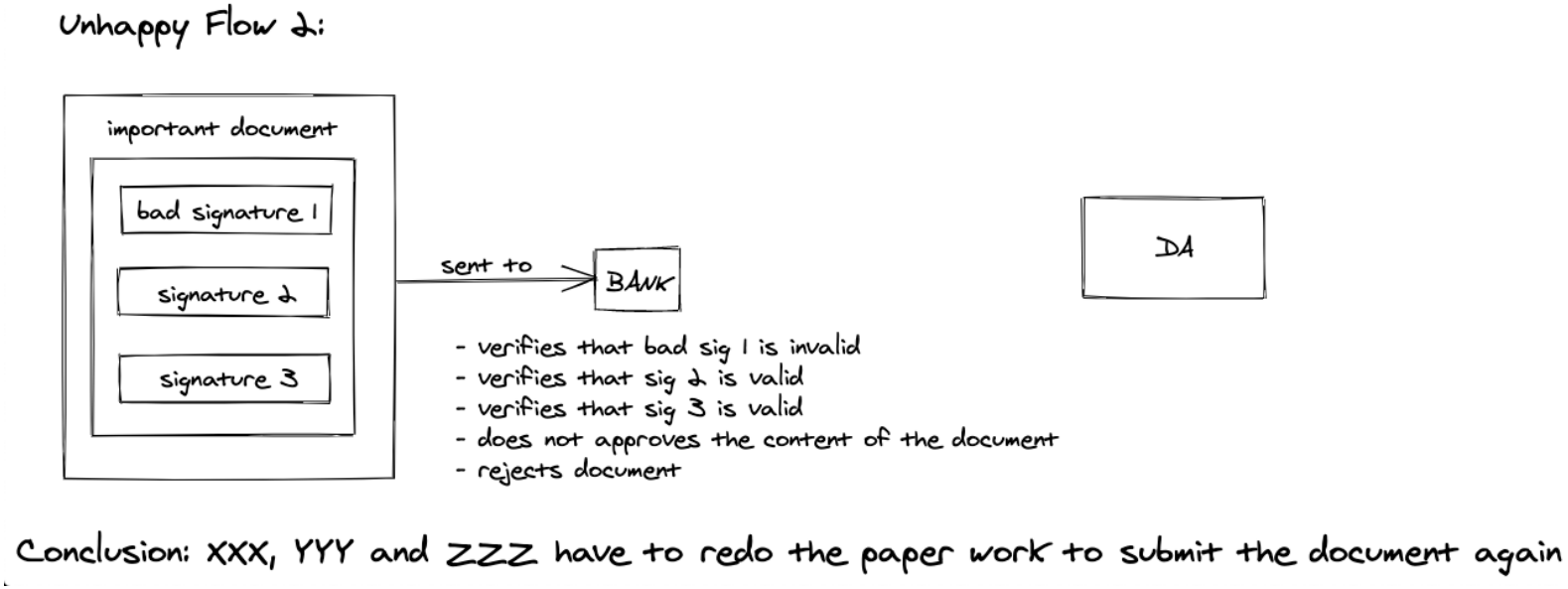

Figure 4: unhappy-flow2

- how can we leverage on etheruem and multi-sig wallets such that we can discard unhappy flow 1 and 2?
- put in a another way, could we prevent unhappy flow 1 and 2 from ever happening?

State of current verification of a verifiable document:

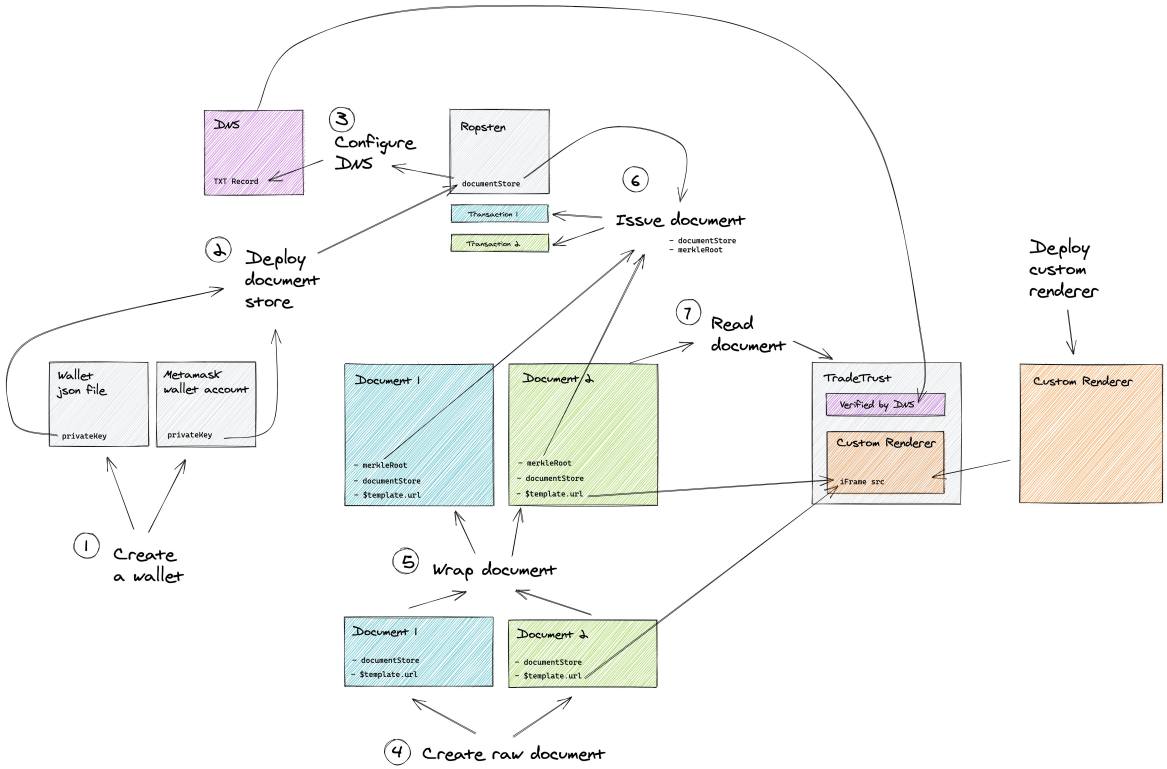

Figure 5: picture of document store flow

Multisig wallets:

- misnomer, actually not a wallet but a **smart contract** in the blockchain (non-custodian wallet)
- essentially a contract that acts as a **staging area** to the other smart contract (document store)
- allows more than 1 party to authorise a transaction
- multiple implementations, each with pros and cons:
  - simple-multi-sig
  - GnosisSafe

simple-multi-sig:

- https://github.com/christianlundkvist/simple-multisig
- minimalistic implementation of a multi-sig wallet
- idea here is to keep surface of attack of the smart contract small
- so features such as spending limits / updating who can sign are not present
- live implementation of a simple-multi-sig factory

Pros:

- only function it exposes is an execute function
- execute function only invokes after it gathers all signatures from all parties
  - how the signatures are gathered is not the concern of this contract
- no features so the cost of deploying should be cheap
- formally verified and audited

Cons:

- `pragma solidity >=0.4.24;` (probably a major dealbreaker)
- does not seem to be maintained for quite some time
- no features :(
- quite a bit of work is needed to be done to make it user-friendly

GnosisSafe:

- https://gnosis-safe.io/
- fully fledged solution for multi-sig (batteries included)

Pros:

- quite feature-full (can go to the gnosis safe website to look at the features)
- everyone pays a bit of the transaction
- good ecosystem for developers to deploy confidently
- formally verified and audited
- excellent user interface and user experience

Cons:

- too many features so cost of deploying is high
- too many features so surface of attack is large
- want a banana, but get the gorilla and tropical island that comes with it (paying for things that one does not need)

how multi-sig will solve the problem defined:

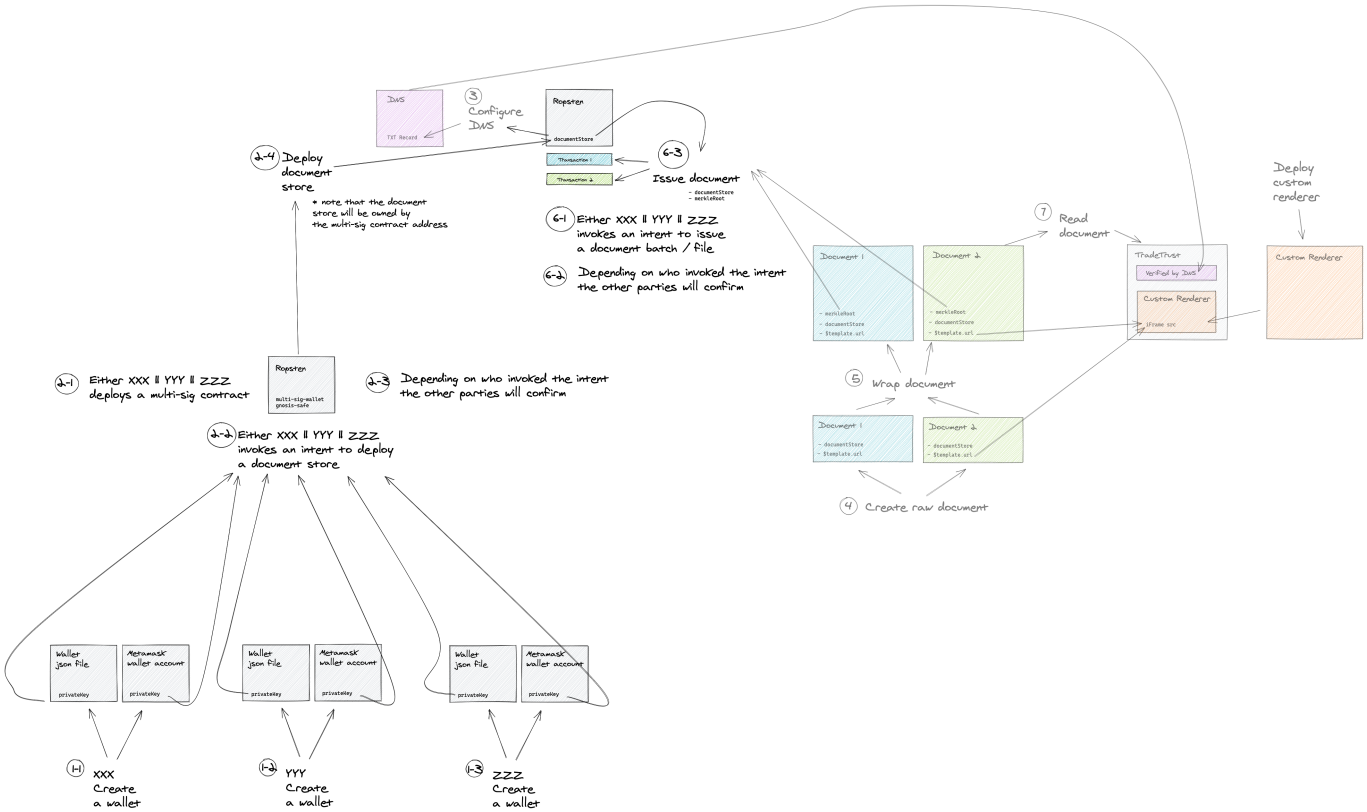

Figure 6: picture of new flow

Conclusion:

- who creates the multi-sig contract?

Conclusion:

- who creates the multi-sig contract?
  - 1 party who will do the initial setup, adding allowed owners etc

Conclusion:

- who creates the multi-sig contract?
  - 1 party who will do the initial setup, adding allowed owners etc
- who maintains the multi-sig contract?

Conclusion:

- who creates the multi-sig contract?
  - 1 party who will do the initial setup, adding allowed owners etc
- who maintains the multi-sig contract?
  - parties involved in the authorisation

Conclusion:

- who creates the multi-sig contract?
  - 1 party who will do the initial setup, adding allowed owners etc
- who maintains the multi-sig contract?
  - parties involved in the authorisation
- who pays for a multi-sig transaction?

Conclusion:

- who creates the multi-sig contract?
  - 1 party who will do the initial setup, adding allowed owners etc
- who maintains the multi-sig contract?
  - parties involved in the authorisation
- who pays for a multi-sig transaction?
  - depending on implementation

Conclusion:

- who creates the multi-sig contract?
  - 1 party who will do the initial setup, adding allowed owners etc
- who maintains the multi-sig contract?
  - parties involved in the authorisation
- who pays for a multi-sig transaction?
  - depending on implementation
  - if simple-multi-sig then the last party pays for the entire transaction (the signatures are gathered off-chain)

Conclusion:

- who creates the multi-sig contract?
  - 1 party who will do the initial setup, adding allowed owners etc
- who maintains the multi-sig contract?
  - parties involved in the authorisation
- who pays for a multi-sig transaction?
  - depending on implementation
  - if simple-multi-sig then the last party pays for the entire transaction (the signatures are gathered off-chain)
  - if gnosis-safe then each party pays for the gas for the approval of the transaction then the party who initiates the transaction will pay for the execution

- can i change parties who own the multi-sig contract?

- can i change parties who own the multi-sig contract?
  - yes if using the gnosis-safe implementation

- can i change parties who own the multi-sig contract?
  - yes if using the gnosis-safe implementation
  - no if using simple-multi-sig

- can i change parties who own the multi-sig contract?
  - yes if using the gnosis-safe implementation
  - no if using simple-multi-sig
- who owns the DNS for this multi-owned document store?

- can i change parties who own the multi-sig contract?
  - yes if using the gnosis-safe implementation
  - no if using simple-multi-sig
- who owns the DNS for this multi-owned document store?
  - who configures it?

- can i change parties who own the multi-sig contract?
  - yes if using the gnosis-safe implementation
  - no if using simple-multi-sig
- who owns the DNS for this multi-owned document store?
  - who configures it?
- should our team provide a quick way to deploy and set up a multi-sig contract (most probably no)

- can i change parties who own the multi-sig contract?
  - yes if using the gnosis-safe implementation
  - no if using simple-multi-sig
- who owns the DNS for this multi-owned document store?
  - who configures it?
- should our team provide a quick way to deploy and set up a multi-sig contract (most probably no)
  - seems like the hype for multi-sig contracts have been impacted by the hacking of Parity multi-sig wallets

- can i change parties who own the multi-sig contract?
  - yes if using the gnosis-safe implementation
  - no if using simple-multi-sig
- who owns the DNS for this multi-owned document store?
  - who configures it?
- should our team provide a quick way to deploy and set up a multi-sig contract (most probably no)
  - seems like the hype for multi-sig contracts have been impacted by the hacking of Parity multi-sig wallets
- how should our team advice companies who want to implement a multi-sig wallet?

- can i change parties who own the multi-sig contract?
  - yes if using the gnosis-safe implementation
  - no if using simple-multi-sig
- who owns the DNS for this multi-owned document store?
  - who configures it?
- should our team provide a quick way to deploy and set up a multi-sig contract (most probably no)
  - seems like the hype for multi-sig contracts have been impacted by the hacking of Parity multi-sig wallets
- how should our team advice companies who want to implement a multi-sig wallet?
  - look through this write up

- can i change parties who own the multi-sig contract?
  - yes if using the gnosis-safe implementation
  - no if using simple-multi-sig
- who owns the DNS for this multi-owned document store?
  - who configures it?
- should our team provide a quick way to deploy and set up a multi-sig contract (most probably no)
  - seems like the hype for multi-sig contracts have been impacted by the hacking of Parity multi-sig wallets
- how should our team advice companies who want to implement a multi-sig wallet?
  - look through this write up
  - engage our team to be deployed to setup a GnosisSafe (some kind of secondment)

- can i change parties who own the multi-sig contract?
  - yes if using the gnosis-safe implementation
  - no if using simple-multi-sig
- who owns the DNS for this multi-owned document store?
  - who configures it?
- should our team provide a quick way to deploy and set up a multi-sig contract (most probably no)
  - seems like the hype for multi-sig contracts have been impacted by the hacking of Parity multi-sig wallets
- how should our team advice companies who want to implement a multi-sig wallet?
  - look through this write up
  - engage our team to be deployed to setup a GnosisSafe (some kind of secondment)
- opportunities for reuse for multi-sig contracts?

- can i change parties who own the multi-sig contract?
  - yes if using the gnosis-safe implementation
  - no if using simple-multi-sig
- who owns the DNS for this multi-owned document store?
  - who configures it?
- should our team provide a quick way to deploy and set up a multi-sig contract (most probably no)
  - seems like the hype for multi-sig contracts have been impacted by the hacking of Parity multi-sig wallets
- how should our team advice companies who want to implement a multi-sig wallet?
  - look through this write up
  - engage our team to be deployed to setup a GnosisSafe (some kind of secondment)
- opportunities for reuse for multi-sig contracts?
  - yes if using the gnosis-safe implementation

- can i change parties who own the multi-sig contract?
  - yes if using the gnosis-safe implementation
  - no if using simple-multi-sig
- who owns the DNS for this multi-owned document store?
  - who configures it?
- should our team provide a quick way to deploy and set up a multi-sig contract (most probably no)
  - seems like the hype for multi-sig contracts have been impacted by the hacking of Parity multi-sig wallets
- how should our team advice companies who want to implement a multi-sig wallet?
  - look through this write up
  - engage our team to be deployed to setup a GnosisSafe (some kind of secondment)
- opportunities for reuse for multi-sig contracts?
  - yes if using the gnosis-safe implementation
  - no if using simple-multi-sig

Further actions:

1) do a poc for a simple-multi-sig contract
2) do a poc for GnosisSafe
3) explore other multi-sig contract options

References:

- https://channel9.msdn.com/Shows/Blocktalk/Introduction-to-Multi-Signature-Wallets
- https://medium.com/@ChrisLundkvist/exploring-simpler-ethereum-multisig-contracts-b71020c19037
- https://docs.opencerts.io/v1/issuing_multisig_certificate.html
- https://www.reddit.com/r/ethdev/comments/8le4tn/executing_funct