Kuwait Koin White Paper

First Email Service Based On Blochchain Technology

Denying the impact of Blockchain technology is a difficult case to sell these days. Almost every industry is beginning to comprehend Blockchain's impact and understand how it will revolutionize the future. After all the Blockchain bridges a very important gap, that of trust.

The proposals put forward are so far revolutionary by all means, overhauling currency, revamping organizations as we know them, ridding the world from intermediaries, and even changing the way we communicate with each other.

If we take one-step back, as impressive as these initiatives sound, they all seem too difficult to implement, in fact, they all want to replace the current order with something entirely new. While this will certainly be true in the end, the path may be somewhat more of an evolution than revolution in many cases.

An alternative view for the application of Blockchain technology is of a multi-step approach. First, the Blockchain acts as an enhancer to the current state of affairs, running in a hybrid model where the current way of doing things is significantly enhanced by the Blockchain. Second, the world moves into full blockchain adoption. Last, Blockchains integrate and coordinate with each other. This journey needs to progress in incremental steps mainly because:

Blockchain technologies are not yet ready for mass adoption in many fields. The limitations on transaction throughput have proven to be a showstopper especially for micro transaction applications such as IoT, and the alternatives or upgrades are still in their early phases.

The work required to overhaul some of the legacy platforms is turning out to be a daunting one.

Regulatory frameworks have yet to adopt to what the Blockchain brings to the world, and without development on that front is hard to imagine the world completely switching to Blockchain-based versions of whatever it is that we are revolutionizing.

Blockchain as a technology is too good to wait for; therefore, incremental implementation of the technology as it matures will propagate across many industries as it matures.

Within this current state of affairs, the case for leveraging the Blockchain for an in transition platform has many merits. With this view, we will have time to bridge the gaps standing in the path of Blockchain mass adoption while retaining what works today.

We illustrate this idea by taking the proposals put forward to revolutionize the email system as an example. Current initiatives propose replacing SMTP with a Blockchain based protocol, changing the way emails are stored into Blockchain based repositories. All these activities require major changes in infrastructure across organizations, the introduction of new protocols, and even changing the addressing scheme we use.

What if we find an 80/20 approach to utilize the benefits of the Blockchain without overhauling the entire email world? This was the question, which we set forward while exploring the idea. To start, we first looked at the main benefits of turning emails into Blockchains, which are as follows:

End to end security and encryption of transferred data
An independent and agreed upon transaction ledger (proof of email transaction)
Anonymity
Decentralization

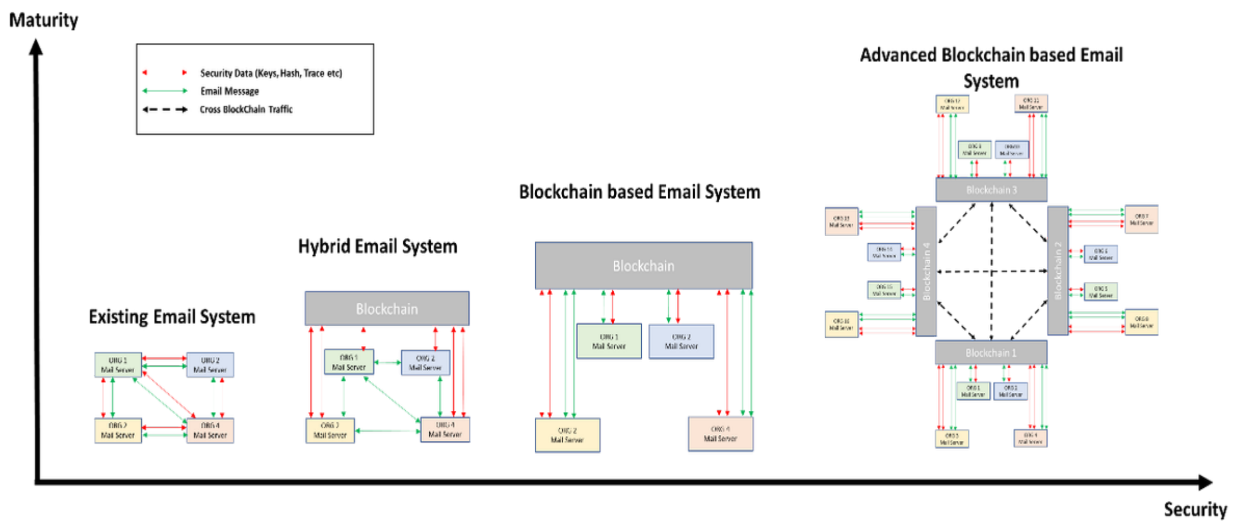
The above benefits largely will be responsible to significant reduction of Spam email and also Phishing attacks through email as traceability, authenticity and integrity will be built-in features of the platform.

We began to explore the possibility of achieving these goals without changing how emails work from the ground up. By running a Blockchain in parallel to SMTP, we can achieve the first two goals and partially the third with minimal interference. We call this approach Hybrid Blockchain Mode, the case for this approach is as follows:

Its simpler to improve an existing system than replacing it
Implementation and adoption is generally faster
Cost is minimal
Running the Blockchain enabled features in parallel provides users with choice (if we take the hybrid automobile analogy, then this is equivalent to the user having the choice between hybrid Vs all electric modes)

We will detail here an approach to secure Emails using encryption while utilizing the Blockchain as a medium of key exchange and transaction ledger. The transaction ledger provides overall accountability and traceability for email exchanges while the encryption is really just an easy add-on to the whole idea. This system runs in parallel to SMTP, not as a replacement to it, providing users with the option of either running on pure SMTP or Blockchain enhanced mode (we call this mode the registered email mode in the example we provide)

We have to emphasize that we view this hybrid approach only a stepping stone towards end-to-end Blockchain based email infrastructure, however the benefits of having a non-disruptive integration with Blockchain at this point in time merits consideration, below is our view on how we foresee Email platforms evolve on the long run using this intermediary step.

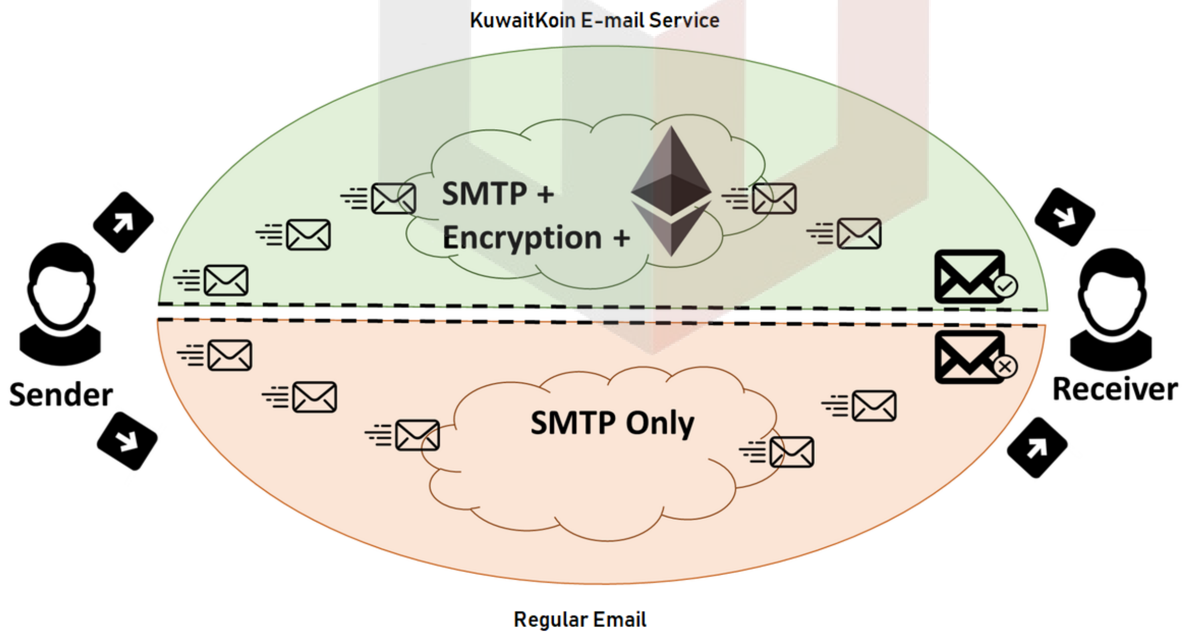


Why is it needed

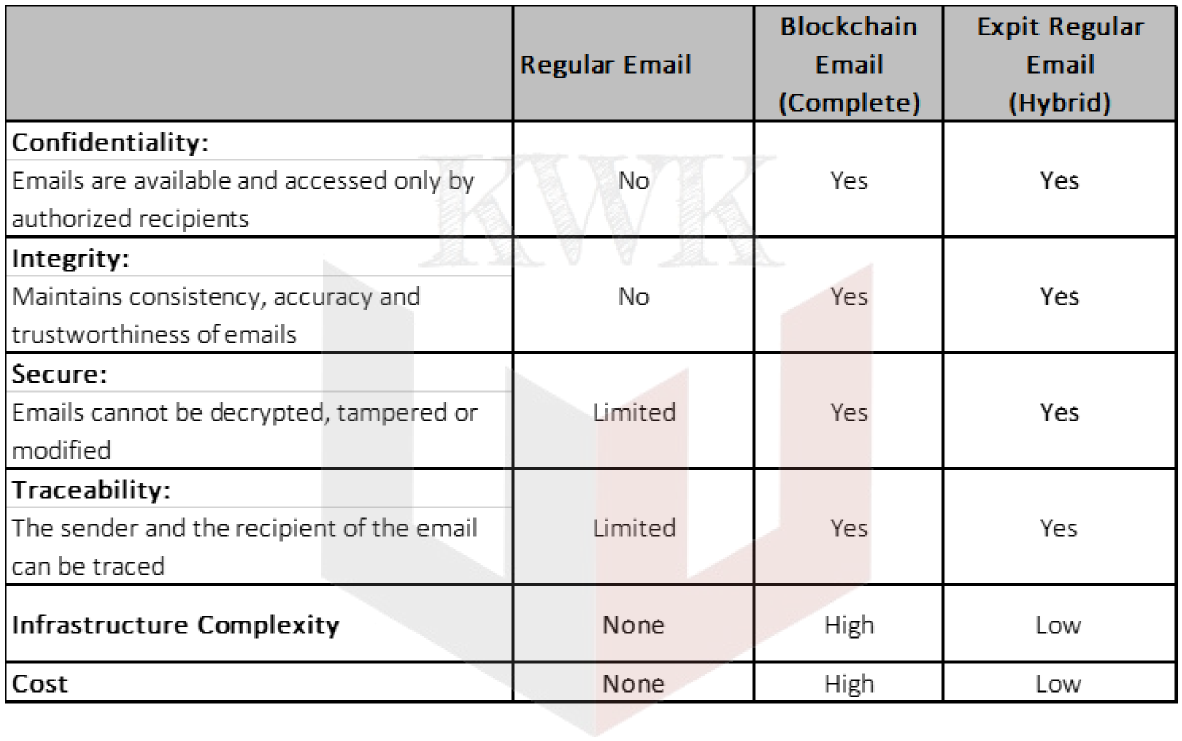
Email encryption alone is no longer enough, the ability to track email flow as transactions is becoming a necessity, especially in cases where business is conducted and real time validation of email flow is required. Think billing, invoicing, and contract exchange. We also need a stronger authentication mechanism for user identities. Securing email accounts as we do in cryptocurrency wallets for critical transactions is a necessity today.

In practice, email encryption today is implemented on the email provider level in most cases, GMAIL to YAHOO, etc. This approach provides in transit security for emails only; it does not take into account many other exposure levels. Additionally, tracking and investigating email transactions is also a complex task today, often requiring access to many internal email platforms and reviewing large amounts of logs for traces of email exchange evidence. Utilizing email as a tamper proof platform for legal exchange is held back by this limitation.

Our approach addresses these challenges of with a simple modification to how existing emails can work when we augment SMTP with BlockChain as an additional security layer. Expits Registered Email solution gives the extra option to the user to send an email through a secured or a regular channel on per email message basis.



A quick comparison between the three different states of email security will help illustrate the value of having a hybrid version of email systems that use the Blockchain for security applications.



	Regular Email	Blockchain Email (Complete)	Expit Regular Email (Hybrid)
Confidentiality: Emails are available and accessed only by authorized recipients	No	Yes	Yes
Integrity: Maintains consistency, accuracy and trustworthiness of emails	No	Yes	Yes
Secure: Emails cannot be decrypted, tampered or modified	Limited	Yes	Yes
Traceability: The sender and the recipient of the email can be traced	Limited	Yes	Yes
Infrastructure Complexity	None	High	Low
Cost	None	High	Low

How it works

We developed two distinct components to augment email security

- An outlook plugin that connects to a Blockchain (in this case the Ethereum Ropsten Testnet)
- A smart contract on Ethereum

Outlook plugin

The Outlook plugin is responsible for handling all activities taking place between the Blockchain and the email client including sending/receiving mechanisms. We chose outlook due to its popularity and ability to be enhanced via simple plug-ins.

We built the plugin using .NET (C#), Ethereum supports this approach via Nethereum, a.NET integration library for Ethereum, this approach also allowed us to interact with Ethereum clients like geth, eth or parity using RPC.

While designing the plug-in we took into consideration not to alter any behavior Outlook users are used to while sending or receiving emails. So, the main entry points for our plugin are embedded within the new-email and read-email screens.

The plugin is designed to ensure the below features:

- Security:
  - “Rijndael Symmetric Encryption Algorithm” is used as our email content encryption algorithm.
  - “RSA Asymmetric Encryption Algorithm” is used to encrypt the shared key. This allows us to protect the shared key during transit via the Blockchain.
- Integrity:
  - After the original email with its attachments (if any) are encrypted. We then use the “SHA256 Algorithm” to hash the encrypted email.
  - Hashing acts as an insurance policy for both the sender and recipient to detect any alteration attempts while emails are in transit
- Traceability:
  - All validations, key exchanges, and audit trails take place via the Blockchain, creating a proof of transaction per email
  - The current email status is updated by referencing the Blockchain contract on each side of the transaction (sender/recipient).

Smart Contract

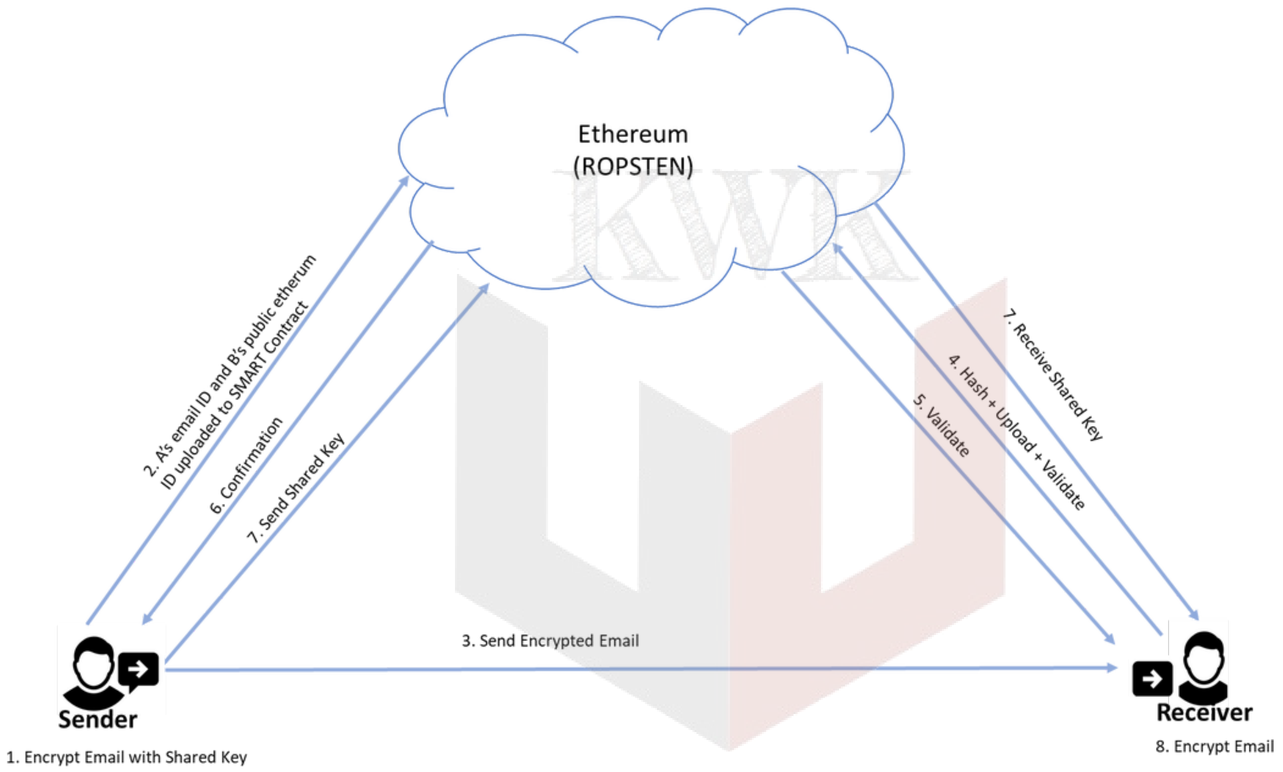
The contract is written in “Solidity” which is a contract-oriented, high-level language for implementing smart contracts. It was influenced by C++, Python and JavaScript and is designed to target the Ethereum Virtual Machine (EVM).

Our smart contract is designed to follow the sending and receiving steps. Each step is represented by a method responsible for either checking a value for verification or saving a value for later retrieval. The contract also includes some helper methods for validation. Below are our main smart contract methods:

- uploadEmailId:
  - Description:
    - Responsible for saving the recipient Ethereum public address along with the email Id which will be used as a reference for all communications between the sender and the receiver.
  - Used by: Sender.
- confirmReceivalAndUploadEmailHash:
  - Description:
    - Responsible for saving the email hash provided by the recipient for sender’s verification.
    - Responsible for saving the RSA public key provided by the recipient for the sender to use it in encrypting the shared key.
  - Used by: Recipient
- getEmailHash/getEmailEncryptionPublicKey
  - Description:
    - Responsible for getting the previously saved values of the email hash and the RSA public encryption key by the recipient.
  - Used by: Sender.
- confirmEmailHash
  - Description:
    - Responsible for confirming the hash provided earlier by the recipient. Also, it saves the encrypted shared key by the RSA public key provided earlier.
  - Used by: Sender.
- getEmailKey
  - Description:
    - Responsible for getting the encrypted shared key that will be used by the recipient to decrypt the original email.
  - Used by: Recipient.

Process flow

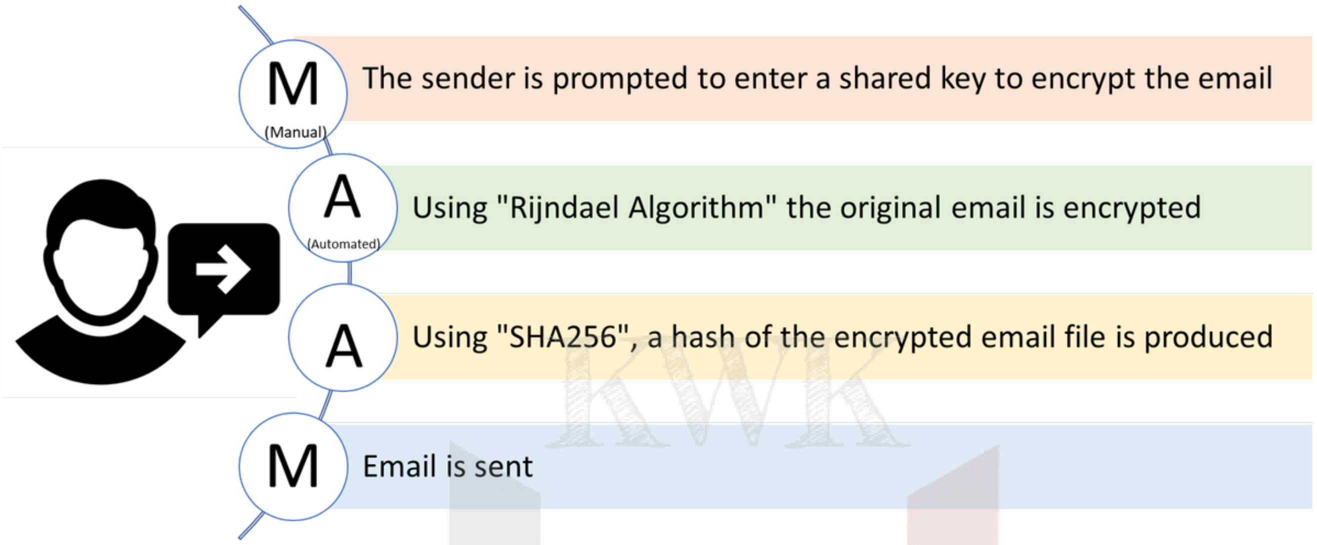
Below is the main flow of both the Outlook plugin working along with the Ethereum smart contract. The process Is performed locally by both parties with most tasks flowing through the etherum contract except for the email transmission step which is carried out via SMTP



The sender encrypts the email using a shared key.
The email Id of the encrypted email is uploaded to the contract along with the recipient Ethereum public address.
Sender A sends the encrypted email to Receiver B
The recipient hashs the recieved email and hash value to the ethereum contract. The recipient also uploads the encryption public key used for shared key exchange.
The sender validates the hash on the ethereum contract for validation.
The sender encrypts the shared key and uploads it to the contract.
The recipient retrieves the encrypted shared key via the Blockchain contract and decrypts the email.

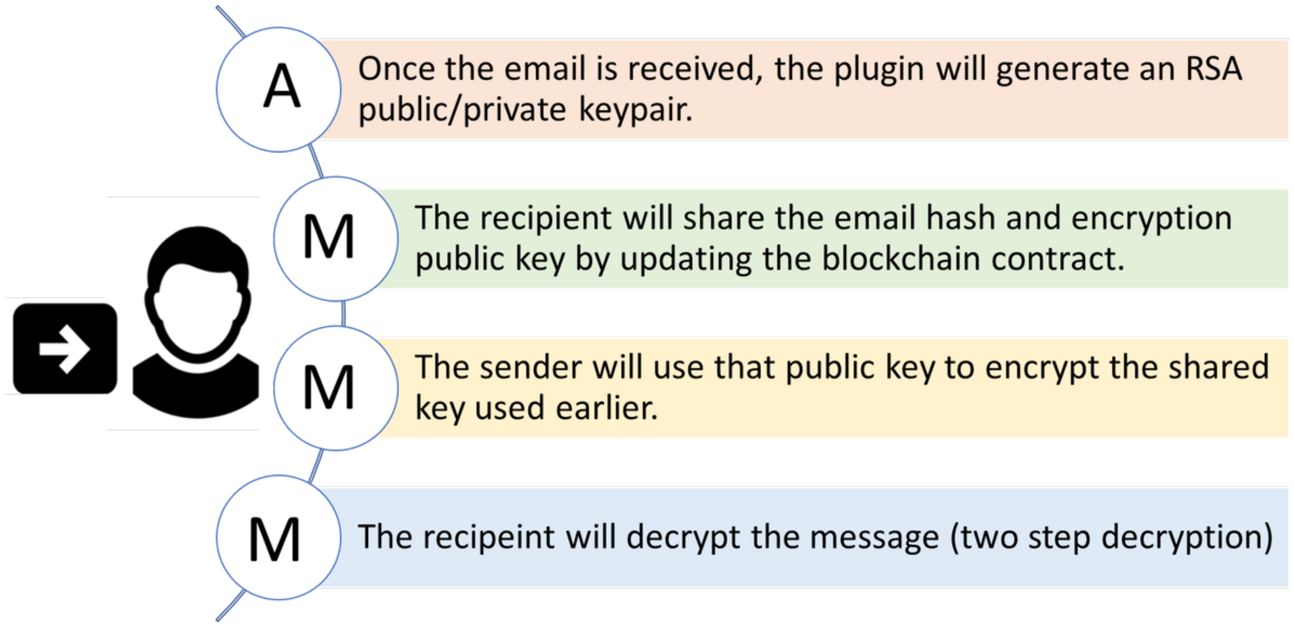
Sending email

The following steps illustrate the send actions (manual and automated) to complete the email exchange


M (Manual)
The sender is prompted to enter a shared key to encrypt the email
A (Automated)
Using "Rijndael Algorithm" the original email is encrypted
A
Using "SHA256", a hash of the encrypted email file is produced
M
Email is sent

Receiving the email

The following steps illustrate the send actions (manual and automated) to complete the email exchange


A
Once the email is received, the plugin will generate an RSA public/private keypair.
M
The recipient will share the email hash and encryption public key by updating the blockchain contract.
M
The sender will use that public key to encrypt the shared key used earlier.
M
The recipeint will decrypt the message (two step decryption)

We designed the process not to interfere with how SMTP works. So, the first step is carried-out inside the new email window which all users are familiar with. Our plugin user will encrypt the email to be registered with a shared key. The encryption process includes generating an email Id to be used as an email reference and the email hash which acts as an invisible seal to the email. The output of this step is an encrypted email as an attachment to the original email. The user then will place a transaction that includes the email Id along with the recipient Ethereum public address on Ethereum using our contract as the first entry point on the blockchain for that email. The user will then send the email normally using Outlook.

After the email is received, the recipient then will confirm the email receival. In that step, the user will hash the received email and generate a public/private encryption keypair to be used by the recipient for encrypting/decrypting the shared key later on. Using our plugin, the user will place a transaction on Ethereum including the email hash and the public encryption key. This part of the process acts as the signage of the recipient on receiving a registered email.

The sender then will verify that the sent email was not altered by validating the hash provided by the recipient. Once confirmed, the sender will then place a transaction on Ethereum containing the shared key used to encrypt the email. The sender will use the public encryption key provided by the recipient to encrypt the shared key to ensure the security of the email content.

As a final step, a transaction is placed on Ethereum by the recipient confirming the retrieval of the encrypted shared key. The recipient will then decrypt the shared key using the private key of the public/private keypair generated earlier. Using the decrypted shared key, the recipient will be able to decrypt the registered email and then reveal its contents.

As you can see, we used the Ethereum Blockchain as our mean of transferring all necessary information needed to complete the email registering process benefiting from the main feature of the blockchain which is its immutable transactions ledger.

Currently known limitations

While this project is aimed towards validating the case for hybrid Blockchain applications, it is by no means ready for production or wide adoption use yet, the limitations we have identified so far are as follows:

Transaction steps required is high:

Users are used to sending emails with one click. introducing any additional clicks is a drawback to this current proof of concept implementation.

Limits on Blockchain TPS rates:

Current network performances are by no means ready to handle global email traffic. Alternative ledger approaches such as Tangle and lighting transactions soon to be introduced into Blockchain networks will be the answer in the future, just not today.

Slowness:

The current average block time of ~15 seconds is slow for people used to sending emails in a 1 second click. Especially given that every step of registering the email is carried out as a transaction on the Blockchain, this means users will have to wait for some time to finish all the steps and have their final email decrypted and registered.

Smart contracts complexity:

Having a Gas limit can be an obstacle for running complex tasks especially on the recipient side who may elect not to spend the required Gas for receiving emails. We can introduce an approach in which the sender bares all the costs, this is an approach we are currently looking into

Anonymity

We have not yet decided on identity protection approach on the long run for such a system, the challenge we are facing has to do with stickiness of participants public address, which will be on the long run associated with that participants email ID.

Summary and the road ahead

The approach we introduced here is one for the in-transition case from a pre to post Blockchain world. We argue that the hybrid implementation of Blockchains, in which existing systems run in parallel to the Blockchain, is in many cases the most realistic way forward. The example we use to illustrate this concept is based on email security. With simple modifications, an email system can benefit from many capabilities the Blockchain brings forward without changing any of the underlying email components. Our aim from this effort is to highlight the benefits of running Blockchains in hybrid mode. We are just at the beginning of the exploration task. Use cases similar to this one will help bridge the transition gap to Blockchain based systems.

We see a future for this approach evolving in two fold, for email client developers and email providers, incorporating this approach into their mail clients is straightforward. Providers can easily add these as options in their current clients while giving users the choice for which Blockchain/contract to join.

For Blockchain platform developers, adding specific features within their platforms to handle email may be a key differentiator around platform dominance as this technology evolves. One candidate feature we are currently exploring is Ethereums precompiled zkSNARK contracts use to extend anonymity of mail traffic through Ethereum.