TRISA

Travel Rule Information Sharing Architecture for Virtual Asset Service Providers (TRISA)

Version 5

December 10, 2019. Version 5

Modifications since Version 4

- Added message formats for originator and beneficiary information
- Updated message flow to include message receipt including blockchain transaction ID
- Updated Fig 3 to show message flow to include blockchain transaction ID (TxID)

TRISA Overview

The Travel Rule Information Sharing Architecture was initiated in July 2019 as a response to the FATF and FinCEN emerging regulations around data transfer for cryptocurrency transactions between Virtual Asset Service Providers (VASPs). The goals of TRISA are:

- Enable compliance on a global level
- Protect user privacy
- Ensure fast and inexpensive transactions
- Open source
- Decentralized
- Open governance body
- Interoperability with other approaches

The goal of the Travel Rule Information Sharing Architecture (TRISA) is to enable compliance with the FATF and FinCEN Travel Rules for cryptocurrency transaction identity information without modifying the core blockchain protocols, and without incurring increased transaction costs or modifying virtual currency peer-to-peer transaction flows.

In June 2019 The Financial Action Task Force (FATF) proposed global rules regarding the sharing of beneficiary and originator information between Virtual Asset Service Providers (VASPs), inspired by regulation from the Financial Crimes Enforcement Network (FinCEN) in the US. Entities subject to these regulations are cryptocurrency exchanges, custodial wallets, DEX operators or others based on the interpretation of regulations in each jurisdiction.

This whitepaper proposes a peer-to-peer mechanism for complying with these regulations, with minimal cost impact to participants, and with consideration for preserving high performance transaction processing at cryptocurrency virtual asset service providers.

FATF Funds Travel Rule

In June 2019, the Financial Action Task Force (FATF)—an international money-laundering watchdog organization based in Paris—released clarification to its guidance to member nations regarding the regulation of VASPs and other crypto entities. In response to the increasing use of virtual assets for money laundering and terrorist financing, the updated guidance includes a "Travel Rule." Essentially, it requires VASPs to share sender (originator) and receiver (beneficiary) information in cryptocurrency transactions. This is similar to so-called Travel Rules that have for years required financial institutions to share this information when executing bank wire transfers and SWIFT electronic funds transfers.

At the close of a summit on June 29, finance ministers and central bankers of the G20 economic bloc formally announced their support for FATF's updated cryptocurrency guidelines, which include the Travel Rule. The G20 member countries have now rapidly begun the process of translating the Travel Rule into their respective local laws.

BSA Travel Rule – 31 CFR 103.33(g)

The Bank Secrecy Act (BSA) established a Funds Travel Rule for fiat currency transfers in the US in 1996 and amended it to include electronic funds transfers 2012. FinCEN is charged with enforcing BSA rules, and in May 2019 it released guidance that Treasury would classify many cryptocurrency exchanges as money service businesses (MSBs), meaning exchanges operating within the United States must now comply with the BSA Travel Rule. According to the rule, any time a transfer of funds is greater than or equal to $3,000, financial institutions must include the following in the transmittal order: the name, account details, and financial institution of the recipient and the transmitter. The regulation's text does not dictate exactly how financial institutions must collect, verify or transfer this information.

The Solution: Modify Blockchains or Add an Overlay Layer?

The goal of the Travel Rule Information Sharing Architecture (TRISA) is to enable compliance with the FATF and FinCEN Travel Rules for transaction identity information without modifying the core blockchain and cryptocurrency protocols. Trying to modify the protocols is bound to fail, as there are many different protocols, and forcing hard forks is simply not feasible. A better option involves creating a separate out-of-band mechanism to augment existing blockchains and cryptocurrencies for compliance purposes.

This whitepaper describes a peer-to-peer mechanism for VASPs to comply with the respective Funds Travel Rule for transaction identification exchange between originators and beneficiaries.

What the FATF Travel Rule Requires VASPs to Retain and Share

The FATF guidelines require both sending and receiving VASPs to exchange and store originator and beneficiary identification information in addition to the cryptocurrency addresses and transaction ID for each transaction. Regulators require the latter since cryptocurrency addresses can be used by multiple beneficiary customers. For example, some exchanges use a single address to send all transactions. Also, cryptocurrency addresses can be recycled and consequently may be used by multiple customers at a VASP.

Applying the Certificate Authority Model to Reliably Identify and Verify VASPs

The FATF rule creates a technical challenge for VASPs—how to comply with the requirement for exchanging information while still protecting user privacy. The solution requires the equivalent of a certificate authority (CA) that verifies the identity of VASPs and serves as a dictionary for their public key certificates so that they can be identified and establish secure communications between VASPs.

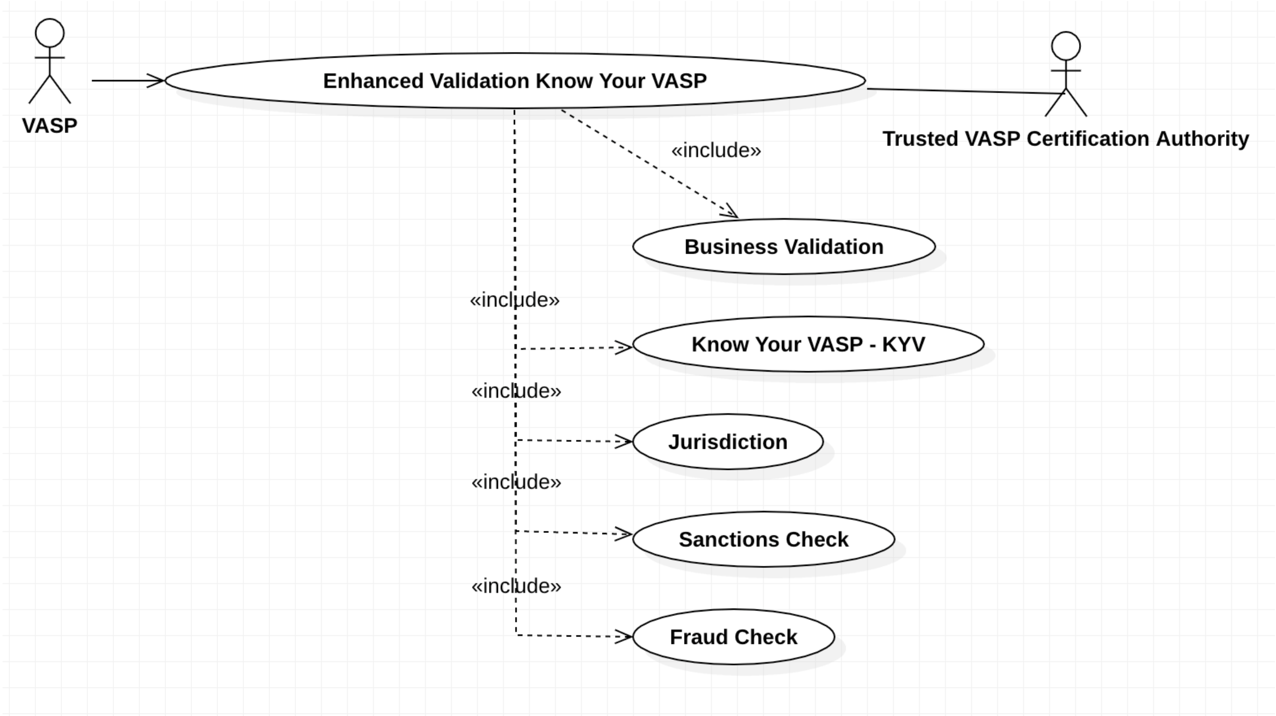


Figure 1: Implementing Enhanced Validation Know Your VASP (EV KYV)

Receiving VASPs should return receipts, ideally digitally signed, to sending VASPs to confirm that the transaction identity information has been received. It may be desirable to be able to reject a transaction in a receipt, for example if the sender's identity or purported beneficiary's identity data fails sanctions or other blocking tests by the receiving VASP. In such cases, the sending VASP should not proceed with the blockchain transaction and should notify the originator of a failed transaction. This enhanced validation standard is known as Extended Validation Know Your VASP (EV KYV). See Figure 1.

In a CA model, one or more third parties verify the identity of a VASP through a number of steps such as email identification, domain name ownership identification, phone call verification, and business paperwork verification. The CA can then issue a digital certificate signed by the CA and the VASP to serve as identification and a way to establish secure encrypted communications with the verified VASP (Figure 2). These certificates should have an expiration date. They should also be subject to revocation by the CA through an OCSP mechanism or revocation list.

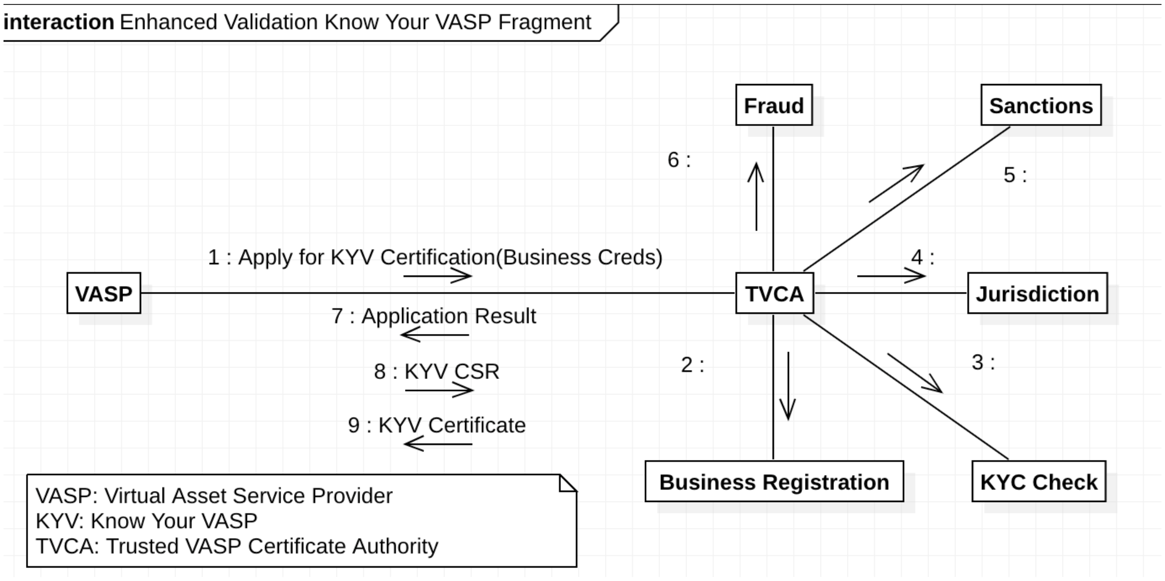

interaction Enhanced Validation Know Your VASP Fragment
Fraud
Sanctions
6 :
5 :
1 : Apply for KYV Certification(Business Creds)
4 :
VASP
TVCA
Jurisdiction
7 : Application Result
2 :
3 :
8 : KYV CSR
9 : KYV Certificate
Business Registration
KYC Check
VASP: Virtual Asset Service Provider
KYV: Know Your VASP
TVCA: Trusted VASP Certificate Authority

Figure 2: A validated certificate X.509 from a certificate authority (CA) protects communications between two VASPs by encrypting the connection between them. In this model, a VASP applies for certification through a registered VASP CA. The CA would then verify that all legal requirements have been met before the VASP can send a certificate signing request (CSR) and the TVASP CA (Trusted VASP Certificate Authority) can produce the signed certificate.

Another approach is to use a mechanism similar to Domain Keys Identified Mail (DKIM) whereby public keys and configuration information regarding where to connect to the transaction identity services are published in the DNS records of VASPs. A problem with this approach, however, is that many VASPs operate multiple domain names for different services, sometimes in different jurisdictions. The CA model provides more oversight and simplification; however, it does require one or more trusted third parties to operate the verification, issuance, and revocation of certificates.

In web connections, the CA is the cornerstone trust for public key infrastructure (PKI), by issuing trusted digital certificates and managing, distributing, and revoking these certificates. It works by using two different cryptographic keys: a public key and a private key. In an TLS interaction, for example, the public key is available to any user that connects with the website. The private key—a unique key generated when a connection is made—remains secret. When communicating, the client app or browser uses the public key to encrypt and decrypt data, while the server uses the private key. This match-up of keys enables the sending and receiving machines to establish a secure connection, which protects users' information from theft or tampering. In addition, since the CA issues digital certificates that associate an entity with a given pubic key, this approach ensures users interact with the intended party, not an imposter.

The CA model is one that has been developed and proven in production on the Internet for over 20 years, having been successfully used success to establish trust on untrusted networks, facilitating the secure electronic transfer of information for a range of network activities such as e-commerce, internet banking and government communications. It provides proven and well-established trust models, audit procedures, issuance and revocation mechanisms.

Establish Secure Communications Between VASPs

When a VASP wishes to send transaction originator and beneficiary information to another VASP in support of the Travel Rule requirements, they must establish secure communication with the other VASP. One way to do this is to identify the VASP, get its certificate and communications address and port, and then establish an SSL/TLS secure connection directly to the receiving VASP. This is much the same way that browsers connect securely to web servers.

Extended Validation KYV Certificates

Extended Verification SSL (EV SSL) certificates provide an additional layer of trust for web communications beyond that provided by standard SSL certificates. Similar to the way in which an EV SSL Certificate authenticates a website and the entity controlling the website, an EV KYV Certificate contains the following required fields, which are validated by a trusted third party:

Required VASP Identity Fields

1. **Subject Organization Name** – Must contain the full legal name of the entity
2. **Registration Number** – The unique Registration Number assigned by the Incorporating Agency in the Jurisdiction of Incorporation
3. **Address of Place of Business** – Must contain the address of the physical location for the Subject. City, state and country information are required
4. **Business Category** – Must contain one of the following strings: "Private Organization," "Government Entity," "Business Entity," or "Non-Commercial Entity"
5. **Subject Jurisdiction of Incorporation or Registration** – The Jurisdiction of Incorporation or Registration
6. **Domain name** – Must contain one or more host domain name(s) owned or controlled by the Subject for association with the Subject's publicly accessible server

Mutual authentication can be facilitated over the connection protocol by the originating VASP providing their identification certificate to the receiving VASP, that then verifies it during the secure communication session establishment and checks revocation status. In this way, both VASPs are certain the counterparty on the other end of the connection is who they say they are.

Reliable Communications Between VASPs

In the case where peer-to-peer communication of transaction identity information is used to satisfy Travel Rule requirements, it is essential that communication between VASPs is reliable. To that extent, sending VASPs must ensure that their systems have reliable retry and resend capabilities to retry sending the information in the case that the receiving party's servers are unreachable or they do not receive a transmission receipt notification.

The protocol to send transaction identity information must also include a receipt that is provided by the receiver to the sender to prove that the data has been received. The receiver should timestamp and digitally sign this receipt and include the hash of the identity information that was sent, so that the sender can store this for non-repudiation purposes in case of a future audit.

Similarly, receivers should not deposit received funds into the account of a beneficiary until the transaction identity information is received from the sending VASP. This brings up the challenge of the receiving VASP determining if an inbound transaction is from a VASP or from an individual private wallet, which is not required to provide transaction identification information under the FATF Travel Rule.

Protocol

The protocol for sending transaction identity information should be:

1. Establish a secure, mutually authenticated SSL/TLS connection between VASPs by the originator to assure privacy of data in transit
2. Originator posts a transaction identification message
3. Receiver posts a signed receipt
4. Originator posts transaction to a blockchain and receives a transaction ID
5. Originator posts the transaction ID to the receiver

Connection Optimization

Because establishing a new SSL/TLS-authenticated session between VASPs for every single transaction could prove to be overly costly in computation for key exchange and session establishment, it is acceptable to keep a connection open and exchange data for multiple transactions over a single connection. This is similar to how a browser keeps HTTPS web connections open for accessing multiple web pages in a single connection with a web server.

Mitigating the Risk of Sending Private Information to the Wrong Entity

The simple way to do this is to ask the sending user if they are sending to a VASP, and if so, which one. VASPs such as Coinbase have used this mechanism for several years. It does rely on trusting the user, but it is the simplest way to implement compliance by senders. To fully comply with the Travel Rule requirements, the user would also have to enter the beneficiary's information before the transaction can proceed, so that this can be stored by the originating VASP and also communicated to the receiving VASP.

This poses the danger that the user can claim the wrong VASP as the intended destination. For example, a user claims that they are sending to VASP CoinAA when they are really sending to VASP CoinBB. Assuming both VASPs are legitimate and are registered in the system, then the sending VASP would establish a connection to CoinAA and would send the user and beneficiary's information to CoinAA. This discloses this private data to the wrong VASP, and also puts the actual receiving VASP, CoinBB, in a compliance violation.

Mitigating this risk is done by verifying that the receiving address is actually controlled by the declared beneficiary VASP. This requires a high-speed lookup whereby the sending VASP can query the beneficiary VASP about the address and confirm that the receiving address actually belongs to that VASP (VASP Address Confirmation protocol).

1. Transactions should not be posted to the blockchain until the receiving VASP confirms the receiving address. This can delay transaction delivery if the lookup mechanism goes down for any reason.
2. Smaller VASPs would have to ensure 7x24 high availability of this mechanism, unlike today where VASPs can operate in batch mode.
3. It could allow mining of addresses to map them to VASPs.
4. It will create errors and delays for users if there is not a match, or the matching is delayed for several minutes. In such cases, the originating VASP would notify the user that there is no match. This could cause much customer support overhead if in fact it was due to the receiving VASP having a maintenance delay or other problem with the mechanism, when in fact it really is a match.

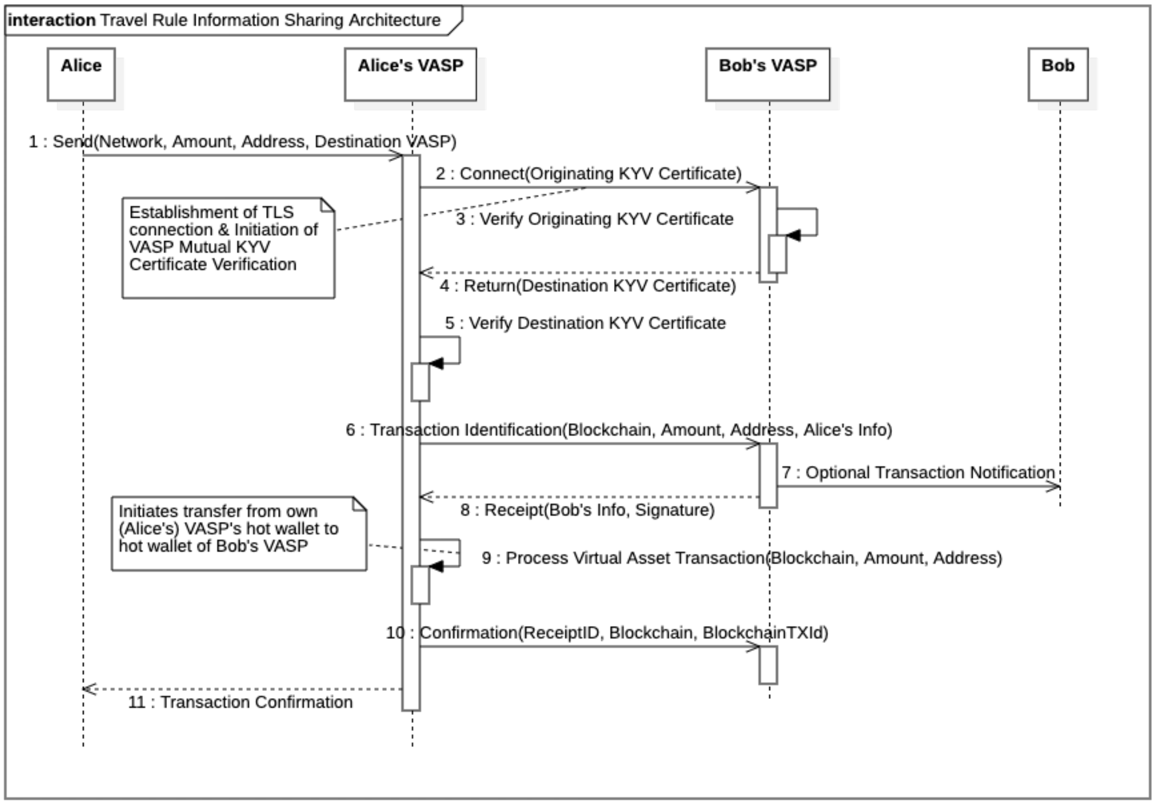

interaction Travel Rule Information Sharing Architecture
Alice
Alice's VASP
Bob's VASP
Bob
1 : Send(Network, Amount, Address, Destination VASP)
2 : Connect(Originating KYV Certificate)
Establishment of TLS connection & Initiation of VASP Mutual KYV Certificate Verification
3 : Verify Originating KYV Certificate
4 : Return(Destination KYV Certificate)
5 : Verify Destination KYV Certificate
6 : Transaction Identification(Blockchain, Amount, Address, Alice's Info)
7 : Optional Transaction Notification
8 : Receipt(Bob's Info, Signature)
Initiates transfer from own (Alice's) VASP's hot wallet to hot wallet of Bob's VASP
9 : Process Virtual Asset Transaction(Blockchain, Amount, Address)
10 : Confirmation(ReceiptID, Blockchain, BlockchainTXId)
11 : Transaction Confirmation

Figure 3: Communication between exchanges begins when the originating exchange establishes a secure connection through CA. The diagram above show how mutual authentication can be facilitated over the connection protocol as the Originator VASPs (here called "Alice's VASP" as a real-world example) also provides their identification certificate to the destination exchange (here Bob's VASP). Once a secure connection is established, the Originator VASP can then initiate the virtual asset transfer to the Destination VASP along with the information required under the Travel Rule. To prove it has received the data, the Destination VASP sends a timestamped and digitally signed receipt that includes the hash of the identity information that was sent. The Originator VASP must keep this information to meet its record-keeping obligations under FATF recommendations.

Determining by Receiver if Sender Is a VASP

A somewhat more complicated problem is how a receiving VASP, who gets an inbound transaction to one of their addresses, can determine if the inbound transaction is from a VASP or not. For full compliance if the inbound transaction is from a regulated VASP the receiving VASP should not make funds available to the beneficiary until the Travel Rule transaction identity information is received and recorded.

One approach would require the receiving VASP to wait for a period of some minutes, and then if no transaction identity information is received to assume that it is from a private individual and not another VASP. The VASP would record it as such, and only then make the funds available to the beneficiary. This seems perilous, as the delay could be due to problems with the transaction identity system by either the sender or receiver's systems, which operate separately from the underlying blockchain. Also, what if the receiver waits five minutes and does not receive transaction identity information and releases funds, but then after six minutes receives the transaction identity information which indicates that the originator is a sanctioned individual or entity or a person known to have criminal ties? The receiving VASP will have committed a violation of the Travel Rule.

If a receiving VASP's transaction identity system goes down for any reason, including external attack or denial of service, then the receiving VASP should not process any inbound transactions—including those from private individuals—as they have no way of knowing if the inbound transactions are from VASPs or not.

Sending VASPs should always ensure they receive a signed receipt of transaction identity information from a receiving VASP before transactions are placed onto a blockchain. This can complicate the processing systems at sending VASPs because multiple transfers of value are typically batched up into a single transaction in order to reduce blockchain processing fees. That workflow would have to be reworked to only queue transfers to happen once the travel rule processing has been confirmed.

Automatically Determining a VASP from a Blockchain Address

We propose that originators with cryptocurrency stored at a VASP declare that they are sending funds to another VASP rather than to an individually managed address. This mechanism has been used by VASPs such as Coinbase for several years. However, some parties have requested the ability to automatically look up if a receiving address is hosted by a VASP and not require users to self-declare this.

This capability can be provided in two ways. The first is to use the VASP Address Confirmation peer-to-peer protocol to test every VASP to see if a new beneficiary address belongs to that VASP. A simple implementation of this “data mining” approach is highly undesirable, as it tests every address against every VASP until a hit is found. The process can be optimized by storing address-to-VASP correlations for addresses that are reused by beneficiaries. The following section discusses optimization of the protocol and detection and revocation of data miners in more detail.

The second way is for VASPs to publish address-to-VASP mappings to a high-speed blockchain when those addresses are created. VASPs can post hashes of addresses rather than actual addresses, providing a modicum of privacy, however it does not protect against data mining of address-to-VASP relationships. This approach requires a very high speed blockchain with minimal confirmation times (seconds) in order to avoid delaying the sending of transactions. If performance of that system were slow, address-to-VASP mapping would become unreliable. Another challenge is that this blockchain would contain every beneficiary address and its associated VASP for all blockchains. It seems like a very heavyweight approach with the aforementioned reliability issues. A more efficient mechanism involves using a centralized service to provide this functionality. Such a service could expire address-to-VASP mappings over time and throttle data mining attempts, but would constitute a central point of failure.

The peer-to-peer discovery mechanism is the preferred approach for the reasons of privacy, decentralization, performance and reduction of impact on existing transaction workflows.

Optimization of the Network

If sending VASPs do not want to trust their customers to declare that funds are being sent to another VASP or a personal wallet, then they need to query other VASPs to confirm if they control the receiving address and associated account. If done randomly, then we can expect each sent transaction to require an API call to 50% of the VASPs. With over 300 active VASPs, this would be inefficient.

Transaction analysis of hundreds of millions of transactions shows that there are trading clusters where exchanges tend to send and receive transactions between a small group of other exchanges. In fact, as much as 60% of transactions from one exchange can be with two other exchanges.

Sending VASPs can employ caching to optimize automated discovery of the receiving VASP without the input of the end user customer. This cache is simply the ordered list of the most frequent VASPs that receive funds. More elaborate caching can be performed on a per-user basis as well. In such a case, queries required prior to sending a transaction can be reduced by well over 90% for a typical exchange or hosted wallet provider.

Certificate authorities can further optimize the network by delivering to VASPs a prioritized list of exchanges to query, based on transaction flows.

Security Concerns

This type of system would create several new security concerns for the VASP industry.

Since the transaction identity sending and receiving services must be online and highly available (7x24), these services are particularly vulnerable to security breaches and attack. A distributed denial of service (DDoS) attack could take a VASP's entire transaction capability offline, and a large-scale attack on the transaction identity services of major exchanges could take the entire industry offline. Today, it is only possible to take a service's user interface offline with a DDoS attack because back-end transaction processing that interfaces with blockchains is typically separated from the visible interface.

Once a VASP implements Travel Rule compliant data exchange and storage the VASP has massive amounts more personal data to protect from data breaches. Today, VASPs only store the personal data of their customers, and only in one location. That data can be stored offline and encrypted in extreme cases. However, with travel rule data requirements, every VASP will have originator and beneficiary data for every transaction. This means that VASPs will find themselves storing the personal information not only of their customers but also of everyone who ever sends them funds.

Revocation and Blacklisting

Certificate Authorities must provide a revocation service to remove VASPs from the trusted counterparty list. The reasons for revoking a VASP can include bankruptcy, fraud, criminal activities, or sanctions. Both Certificate Revocation Lists (CRLs) and real-time Online Certificate Status Protocol (OCSP) lookups should be supported. OCSP lookups can be performed on a transaction level.

OCSP lookups can also provide blacklisting of malicious VASPs that are trying to datamine counterparty address information. For example, if a VASP decides to try and find which exchanges hosts every address on a blockchain, this lookup data can be reported to the CAs, and then used to create revocation data and update a blacklist.

Encryption of Transmitted Data

In a solution where data is encrypted during transmission between VASPs, using SSL/TLS connections eliminates the further data encryption of originator or beneficiary information during transit. The onus for checking transaction identity data for sanctioned or suspicious persons or entities falls on both sending and receiving VASPs. Thus, they must access this information in plaintext. Naturally, once checked, VASPs should store the information in an encrypted database, but the data must remain accessible at any time for filing CTR, SAR, for audits, and financial investigations.

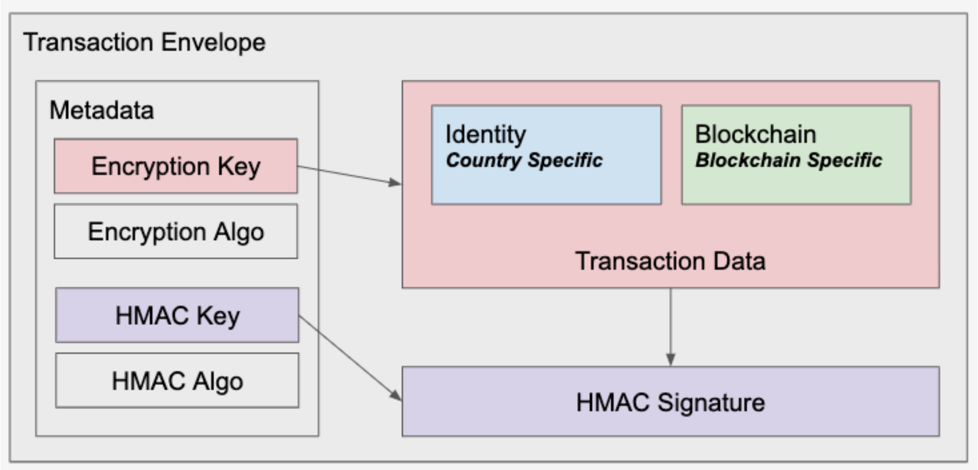
Message Formats

TRISA defines a trust architecture and peer-to-peer model for compliance with the Travel Rule. In Version 4, TRISA did not define message formats for sending the actual compliance data (account number, name, geographic address, beneficiary name and account number). TRISA V5 defines message formats as follows. These are subject to review and modification based on collaboration with GDF and Digital Chamber collaboration.

The peer-to-peer exchange protocol is defined in two different layers:

1. The wire protocol itself, which defines the API to exchange the Transaction Data envelop
2. The Transaction Data itself is composed of two distinct parts:
    a. Identity Information
    b. Blockchain Data

All APIs follow a clear versioning paradigm to allow for future expansion and integrations. When integrating with TRISA or maintaining additional libraries or SDK’s, you can verify on the TRISA documentation website which versions are current or deprecated. (***NOTE: to be added on our website mainpage)***



The peer-to-peer TRISA protocol is defined under proto/trisa/protocol. It defines the RPC exchange endpoint and the Transaction Data envelop. Next to the Identity and blockchain Data, the envelop applies additional encryption of the message and an HMAC digital signature.

Although all messages are exchanged over a secure channel, by adding this additional layer VASP's can securely store the full Transaction Envelope at rest in a backend of their choice while maintaining full repudiation of the exchange.

https://github.com/trisacrypto/trisa/blob/master/proto/trisa/protocol/v1alpha1/trisa.proto

```
// Peer-to-peer Exchange Service for the Transaction Envelope
service TrisaPeer2Peer {
    rpc TransactionStream(stream Transaction) returns (stream Transaction) {}
}

message Transaction {
    // The transaction identifier generated by the sender. Any response
    // to a transaction request needs to carry the same identifier.
    string id = 1;

    // Encrypted TransactionData
    bytes transaction = 2;

    // Encryption key used to encrypt the transaction blob. This key itself
    // is encrypted using the public key of the receiver.
    bytes encryption_key = 3;

    // The encryption algorithm used to encrypt the transaction blob.
    string encryption_algorithm = 4;

    // HMAC signature calculated from encrypted transaction blob.
    bytes hmac = 5;
    // The HMAC secret used to calculate the HMAC signature. This secret
    // itself is encrypted using the public key of the receiver.
    bytes hmac_secret = 6;

    // The algorithm used to calculate the HMAC signature.
    string hmac_algorithm = 7;
}

// Transaction Data composed of both Identity and blockchain Data
message TransactionData {
    // Identity contains any valid identity structure.
    google.protobuf.Any identity = 1;

    // Data contains the network specific data.
    google.protobuf.Any data = 2;
}
```

The Transaction Data is split into two section to be able to maintain and version them separately:

1. The different fields for the Identity section depend on the country/region of the subject being described in it. TRISA suggests using ISO 3166-1 alpha-2 country codes to maintain the different formats. If any subdivision is needed, this can be easily accomplished using dedicated region suffixes (i.e. “us-ca”).

2. The Blockchain data allows for specific properties defining the transaction based on the specific blockchain used. Although there might be some common ground among different blockchains, from a protocol perspective TRISA suggest maintaining clear and dedicated formats per blockchain type.

Note that next to the versioning of the wire protocol itself, both Identity and Blockchain Data sections maintain their own versioning. Once a message has been vetted by the community it will become stable and should be accepted by any production client.

At its current state, TRISA defines Identity examples for “us” (United States of America) and “be” (Belgium), see https://github.com/trisacrypto/trisa/tree/master/proto/trisa/identity.

proto/trisa/identity/us/v1alpha1

```
message Identity {
    string first_name = 1;
    string last_name = 2;
    string ssn = 3;
    string state = 4;
    string driver_license = 5;
}
```

proto/trisa/identity/be/v1alpha1

```
message Identity {
    string first_name = 1;
    string last_name = 2;
    string national_number = 3;
    string city_of_birth = 4;
}
```

For the Blockchain specific data there are two examples for “bitcoin” and “ethereum”, see https://github.com/trisacrypto/trisa/tree/master/proto/trisa/data.

proto/trisa/data/bitcoin/v1alpha1

```
message Data {
    string source = 1;
    string destination = 2;
    string amount = 3;
}
```

proto/trisa/data/ethereum/v1alpha1

```
message Data {
    string source = 1;
    string destination = 2;
    string amount = 3;
}
```

Based on community feedback, TRISA would like to expand on all different Blockchains and different country Identity definitions. Note that the above Identity and Blockchain data are examples and need to be formalized and vetted by the TRISA community.

Versioning paradigm

Each message starts at ***v1alpha1***. When the TRISA community vetted an API for prime time, it gets bumped to ***v1***. If needed any intermediate version like ***v1alph2***, ***v1alpha3*** or even ***v1beta1***, … can be applied. TRISA is using gRPC which allows for compatible schema evolution as not all clients will get updated at once.

By decoupling the wire protocol from the messages TRISA can evolve much easier, depending on the introduced changes.

Message formats are:

Does Cryptocurrency Address belong to this VASP?
    Cryptocurrency type (chain)
    Cryptocurrency address

Sending Transaction Identity Information
    Originator Name
    Originator Address
    Originator State
    Originator Account Number
    Beneficiary Name
    Beneficiary Account Number

Receipt
    Receipt ID tied to Transaction Identity Information Send

Transaction Confirmation
    Receipt ID
    Blockchain Transaction Number

Here are the initial message formats for exchanging identity information.

```
message Identity {
        string first_name = 1;
        string last_name = 2;
        string ssn = 3;
        string state = 4;
        string driver_license = 5;
}
```

```
message
Identity
{
               string first_name = 1;
               string last_name = 2;
               string national_number = 3;
               string city_of_birth = 4;
           }
```

The TRISA transaction ID which is in the Transaction message ("id" field) needs to be present in each message exchange. This is how both parties can correlate messages:

1. Originator --> sends transaction identification - the "TRISA transaction id" is generated by the originator
2. Receiver --> validates the wallet, sends back receipt - this message uses the same "id" as received from #1
3. Originator --> correlates using this TRISA transaction "id" to which the response belongs
4. After putting the TX on the blockchain, originator send the confirmation message containd both TXID + TRISA transaction id
5. Receiver uses again the TRISA transaction id to know to which message exchange this confirmation belongs

Collaboration on Message Formats

The TRISA project will be collaborating with other efforts to standardize the data interoperability through message formats.

TRISA participants are working with GDF and Digital Chamber on message format standardization.

GDF AML Working Group
https://www.gdf.io/mem_wgroup/kyc-aml-ctf/

Digital Chamber

https://digitalchamber.org

TRISA participants are working with Shyft on interoperability of VASP identity certificates.

https://www.shyft.network

TRISA participants are working with NetKi on how BIP 75 might be adapted to support Travel Rule requirements, and interoperability with the TRISA framework.

https://github.com/bitcoin/bips/blob/master/bip-0075.mediawiki

Open Source Project

The TRISA project is open source. The latest code can be found here:

https://github.com/trisacrypto/trisa

For more information or to submit comments
Please contact:
David Jevans, CipherTrace
trisa@ciphertrace.com