TODAQ

Exchange Guide

The Toda-as-a-Service (TaaS) API supports special functionality to provide a different set of security options for customers such as exchanges. In its role as a trusted service provider for enterprises and financial institutions, TODAQ manages critical assets and keys, with each API request signed by a second set of keys. Some TaaS users have regulatory or policy requirements to manage all keys themselves, and for those clients, TODAQ offers the ability to do fully external key management. This renders certain TaaS features inaccessible, like access to TODAQ's instance payment verifications, and introduces risk. Specifically, the loss of a key will lead to the inability to ever transact assets belonging to an address, and the theft of a key would permit the acquirer to transact those assets to their benefit.

Risks of self-managed keys

- Loss of key: assets can never be transacted again
- Theft of key: assets are no longer under your control

Benefits of self-managed keys

- Complete control over key management
- TODAQ is not a party to any transaction
- Fine-grained control over the details of every transaction before assent

Using External Keys

A TODA address is, in fact, the `sha256` hash of a public key. So, in order to use a supplied public key for an address, that key must be provided at the time the address is established. When creating an account in TaaS, supplying the hex-encoded X.509 public key as the `address-public-key` attribute will permanently link the resulting address to that key. Consequently, all activity undertaken by the TODA address will require a signature associated with its public key.

TODAQ requires that keys are `secp256r1` keypairs. We rely on ASN.1 series X.690 DER encoding (X.509) throughout, which is standard in several frameworks, including `java.security`. Signatures are similarly standard, using `SHA256withECDSA`.

Since the TODA protocol operates on timed cycles which are out-of-band from the time a transaction request is made on TaaS, the opportunity to sign a transaction must similarly be communicated asynchronously. At the time of the next cycle following one or more transaction requests, TaaS will communicate the fully-assembled transaction, in the form of a *transaction packet* and associated *proofs*, to the webhook configured in the *Settings* pane of the associated enterprise account. In order for TODA to actually submit the transaction for processing, those using self-managed, external keys, are required to scrutinize, sign, and return this transaction packet in a short window of time (approximately 15 seconds).

Signature Requests

The TaaS API documentation contains a detailed section entitled *Address Public Key*, which specifies how to interpret the supplied transaction packet, how to create a valid signed response, and the endpoint to which the assembled, signed packet must be sent. TaaS will respond synchronously with whether the signature attempt was successful, and if not, with an explanation of the failure. If a given cycle was not adequately signed, TaaS will subsequently report the associated transactions with the status of `MISSED-SIGNATURE`.

The API documentation contains a byte-by-byte description of the frames within both the transaction and signature packets, as well as the construction and `sha256` hashing required of the signature block. At this time, the signature block header is a reserved set of 32 bytes and should remain nulled. Exchanges which have a business need to validate the details of the proofs supplied with the transaction packet prior to signing are recommended to use the `toda.js` client library for parsing and inspecting TODA proof structures.

For more information or assistance, please reach out to your Account Manager, or one of the following technical points of contact:

Chris Kellendonk
Key Management Systems Engineer
chris.kellendonk@todaqfinance.com

Adam Gravitis
Chief Technology Officer
adam.gravitis@todaqfinance.com