Signatures

When a user submits a signed transaction, a serialized signature and a serialized transaction data is submitted. The serialized transaction data is the BCS serialized bytes of the struct TransactionData and the serialized signature is defined as a concatenation of bytes of flag || sig || pk .

The flag is a 1-byte representation corresponding to the signature scheme that the signer chooses. The following table lists each signing scheme and its corresponding flag:

The sig bytes are the compressed bytes representation of the signature instead of DER encoding. The following table lists the expected size of each format:

The pk bytes are the bytes representation of the public key corresponding to the signature.

The signature must commit to the hash of the intent message of the transaction data, which you can construct by appending the 3-byte intent before the BCS serialized transaction data. To learn more on what an intent is and how to construct an intent message, see Sui Intent Signing .

When invoking the signing API, you must first hash the intent message of the transaction data to 32 bytes using Blake2b. This external hashing is distinct from the hashing performed inside the signing API. To be compatible with existing standards and hardware secure modules (HSMs), the signing algorithms perform additional hashing internally. For ECDSA Secp256k1 and Secp256r1, you must use SHA-2 SHA256 as the internal hash function. For pure Ed25519, you must use SHA-512.

An accepted ECDSA secp256k1 and secp256r1 signature must follow:

An accepted pure Ed25519 signature must follow:

See a concrete example for offline signing using CLI in the Offline Signing topic.

For more on zkLogin signature, see zkLogin .

For more on passkey signature, see SIP-8 .

The Authority on Sui (collection of validators) holds three distinctive keypairs:

The protocol key pair provides authority signatures on user-signed transactions if they are verified. When a stake of the authorities that provide signatures on user transactions passes the required two-thirds threshold, Sui executes the transaction. Sui uses the BLS12381 scheme for its fast verification on aggregated signatures for a given number of authorities. In particular, Sui uses the minSig BLS mode, where each individual public key is 96 bytes, while the signature is 48 bytes. The latter is important as typically validators register their keys once at the beginning of each epoch and then they continuously sign transactions; thus, we optimize on minimum signature size.

As with the BLS scheme, you can aggregate independent signatures resulting in a single BLS signature payload. Sui also accompanies the aggregated signature with a bitmap to denote which of the validators signed. This effectively reduces the authorities' signature size from (2f + 1) × BLS_sig size to just one BLS_sig payload, which in turn has significant network cost benefits resulting in compressed transaction certificates independently on the validators set size.

To counter potential rogue key attacks on BLS12381 aggregated signatures, proof of knowledge of the secret key (KOSK) is used during authority registration. When an authority requests to be added to the validator set, a proof of possession is submitted and verified. See Intent Signing on how to create a proof of possession. Unlike most standards, the Sui proof of knowledge scheme commits to the address as well, which offers an extra protection against adversarial reuse of a validator's BLS key from another malicious validator.

The account that the authority uses to receive payments on staking rewards is secured by the account key pair. Sui uses pure Ed25519 as the signing scheme.

The private key is used to perform the TLS handshake required for consensus networking. The public key is used for validator identity. Pure Ed25519 is used as the scheme.

See more authority key toolings in Validator Tool .

Signature requirements

The signature must commit to the hash of the intent message of the transaction data, which you can construct by appending the 3-byte intent before the BCS serialized transaction data. To learn more on what an intent is and how to construct an intent message, see Sui Intent Signing .

When invoking the signing API, you must first hash the intent message of the transaction data to 32 bytes using Blake2b. This external hashing is distinct from the hashing performed inside the signing API. To be compatible with existing standards and hardware secure modules (HSMs), the signing algorithms perform additional hashing internally. For ECDSA Secp256k1 and Secp256r1, you must use SHA-2 SHA256 as the internal hash function. For pure Ed25519, you must use SHA-512.

An accepted ECDSA secp256k1 and secp256r1 signature must follow:

An accepted pure Ed25519 signature must follow:

See a concrete example for offline signing using CLI in the Offline Signing topic.

For more on zkLogin signature, see zkLogin .

For more on passkey signature, see SIP-8 .

The Authority on Sui (collection of validators) holds three distinctive keypairs:

The protocol key pair provides authority signatures on user-signed transactions if they are verified. When a stake of the authorities that provide signatures on user transactions passes the required two-thirds threshold, Sui executes the transaction. Sui uses the BLS12381 scheme for its fast verification on aggregated signatures for a given number of authorities. In particular, Sui uses the minSig BLS mode, where each individual public key is 96 bytes, while the signature is 48 bytes. The latter is important as typically validators register their keys once at the beginning of each epoch and then they continuously sign transactions; thus, we optimize on minimum signature size.

As with the BLS scheme, you can aggregate independent signatures resulting in a single BLS signature payload. Sui also accompanies the aggregated signature with a bitmap to denote which of the validators signed. This effectively reduces the authorities' signature size from $(2f + 1) \times$ BLS_sig size to just one BLS_sig payload, which in turn has significant network cost benefits resulting in compressed transaction certificates independently on the validators set size.

To counter potential rogue key attacks on BLS12381 aggregated signatures, proof of knowledge of the secret key (KOSK) is used during authority registration. When an authority requests to be added to the validator set, a proof of possession is submitted and verified. See Intent Signing on how to create a proof of possession. Unlike most standards, the Sui proof of knowledge scheme commits to the address as well, which offers an extra protection against adversarial reuse of a validator's BLS key from another malicious validator.

The account that the authority uses to receive payments on staking rewards is secured by the account key pair. Sui uses pure Ed25519 as the signing scheme.

The private key is used to perform the TLS handshake required for consensus networking. The public key is used for validator identity. Pure Ed25519 is used as the scheme.

See more authority key toolings in Validator Tool .

Authority signature

The Authority on Sui (collection of validators) holds three distinctive keypairs:

The protocol key pair provides authority signatures on user-signed transactions if they are verified. When a stake of the authorities that provide signatures on user transactions passes the required two-thirds threshold, Sui executes the transaction. Sui uses the BLS12381 scheme for its fast verification on aggregated signatures for a given number of authorities. In particular, Sui uses the minSig BLS mode, where each individual public key is 96 bytes, while the signature is 48 bytes. The latter is important as typically validators register their keys once at the beginning of each epoch and then they continuously sign transactions; thus, we optimize on minimum signature size.

As with the BLS scheme, you can aggregate independent signatures resulting in a single BLS signature payload. Sui also accompanies the aggregated signature with a bitmap to denote which of the validators signed. This effectively reduces the authorities' signature size from $(2f + 1) \times$ BLS_sig size to just one BLS_sig payload, which in turn has significant network cost benefits resulting in compressed transaction certificates independently on the validators set size.

To counter potential rogue key attacks on BLS12381 aggregated signatures, proof of knowledge of the secret key (KOSK) is used

during authority registration. When an authority requests to be added to the validator set, a proof of possession is submitted and verified. See Intent Signing on how to create a proof of possession. Unlike most standards, the Sui proof of knowledge scheme commits to the address as well, which offers an extra protection against adversarial reuse of a validator's BLS key from another malicious validator.

The account that the authority uses to receive payments on staking rewards is secured by the account key pair. Sui uses pure Ed25519 as the signing scheme.

The private key is used to perform the TLS handshake required for consensus networking. The public key is used for validator identity. Pure Ed25519 is used as the scheme.

See more authority key toolings in Validator Tool .