Sui Validator Node Configuration

Validators on the Sui network run special nodes and have additional tasks and responsibilities beyond those of Full node operators.

To run a Sui validator, you must set up and configure a Sui Validator node. After you have a running node, you must have a minimum of 30 million SUI in your staking pool to join the validator set on the Sui network.

To learn how to set up and configure a Sui Validator node, see Sui for Node Operators on GitHub. The guide includes all of the information you need to configure your Validator node. It also provides guidance on the tasks you must perform after you join the validator set.

Specific steps you must take include:

There are minimum staking requirements a validator must satisfy to become active and to stay in the active validator set.

More precisely:

Suggested minimum hardware specifications to run a Sui Validator node:

The total voting power on Sui is always 10,000, regardless of the amount staked. Therefore, the quorum threshold is 6,667. There is no limit to the amount of SUI users can stake with a validator. Each validator has consensus voting power proportional to SUI in its staking pool, with one exception: the voting power of an individual validator is capped at 1,000 (10% of the total). If a validator accumulates more than 10% of total stake, the validator's voting power remains fixed at 10%, and the remaining voting power is spread across the rest of the validator set.

When users stake SUI tokens, these SUI objects are wrapped into StakedSUI objects. The calculation to determine each user's relative ownership of the staking pool is done directly with the timestamp of the StakedSUI object (which determines the moment at which the deposit took place) and the change in the exchange rates between the deposit epoch and the withdrawal epoch. Each staking pool's data structure contains a time series with that pool's exchange rates. These exchange rates can be used to determine the withdrawals of any of the pool's stakers.

Stake withdrawals are processed immediately with the exchange rate prevailing at the previous epoch's exchange rate. Withdrawals do not have to wait for the current epoch to close. Withdrawals include both the original stake the user deposited and all the stake rewards accumulated up to the previous epoch. Stakers do not earn the rewards accruing to their stake during the epoch at which they withdraw. Since there is no way to know how many stake rewards will be accumulated during the current epoch until the epoch closes, these cannot be included in the withdrawal. Hence, any user can withdraw their stake immediately and receive:

SUI withdrawn at E' = (SUI deposited at E) * (Exchange Rate at E'-1 / Exchange Rate at E)

Each epoch change emits a 0x2::validator_set::ValidatorEpochInfo event per validator with the exchange rate information. You can use the Events API to query events.

Within a given validator staking pool, all stakers receive the same proportion of rewards through the pool's exchange rate appreciation. In addition, since validators earn commissions over the stake they manage, validators receive additional StakedSUI objects at the end of each epoch in proportion to the amount of commissions their staking pool earns.

Staking rewards are funded by transaction gas fees collected during the current epoch and by stake subsidies released at the end of the epoch.

StakeRewards = StakeSubsidies + GasFees

Stake subsidies are intended to subsidize the network during its early phases and are funded by a 10% allocation of SUI tokens. After this allocation depletes, the entirety of stake rewards will be made up of gas fees collected through regular network operations.

Stake rewards are made up of gas fees and stake subsidies. The total amount distributed throughout each epoch is determined as follows:

The total amount of gas fees collected corresponds to the sum of gas fees across all transactions processed in the epoch. During regular market conditions, the vast majority of transactions should have a GasPrice equal to the ReferenceGasPrice .

A stake deposit request goes into a pending state immediately in the staking pool as soon as it is made. Sui Wallet reflects any pending stake deposit requests for the user's account. However, pending stake deposit requests do not take effect until the end of the epoch during which the request is made.

A withdrawal (un-stake) request is processed immediately as soon as it is received. The staker obtains the originally deposited SUI together with all accrued stake rewards up to the previous epoch boundary – in other words, they do not include stake rewards for the current epoch.

Users can't withdraw a portion of their active stake. They must withdraw all staked SUI at the same time. Users can, however, stake using multiple StakedSui objects by splitting their SUI into multiple coins. They can then perform a partial withdrawal from a validator by un-staking only some of the StakedSUI objects.

Sui is designed such that end-users can expect the gas price to be stable and predictable during regular network operations. This is achieved by having validators set the network's reference gas price at the beginning of each epoch.

Operationally this is achieved through a gas price survey that occurs as follows:

For example, assume that there are seven validators with equal stake, and the price quotes they submit are {15, 1, 4, 2, 8, 3, 23} . The protocol sets the reference gas price at 8.

In practice, the process for submitting a gas price quote for the Gas Price Survey is a straightforward one. Each validator owns an object that contains their quote for the reference gas price. To change their response, they must update the value in that object.

For example, to set the price quote for the next epoch to 42, run:

Beginning with the Sui v1.24.1 release , the --gas-budget option is no longer required for CLI commands.

Importantly, the gas object's value persists across epochs so that a validator who does not update and submit a new quote uses the same quote from the previous epoch. Hence, a validator seeking to optimize its own operations should update its quote every epoch in response to changes in network operations and market conditions.

Sui is designed to encourage and enforce community monitoring of the validator set. This is done through the Tallying Rule by which each validator monitors and scores every other validator in order to ensure that everyone is operating efficiently and in the network's best interest. Validators that receive a low score can be penalized with slashed stake rewards.

The protocol only computes the global Tallying Rule score at the epoch boundary and so relies on validators monitoring actively and changing their individual scores whenever they detect changes in other validator behavior. In general, the Tallying Rule default option should always be a score of one for all validators and only be changed to zero upon determining bad operations. In practice, the Tallying Rule consists of a set of objects each validator owns that default to scores of one and thus a validator will generally be passive and only update the object corresponding to another validator's score whenever needed.

For example, to report a validator whose Sui address is 0x44840a79dd5cf1f5efeff1379f5eece04c72db13512a2e31e8750f5176285446 as bad or non-performant, run:

The Tallying Rule should be implemented through a social equilibrium. The validator set should actively monitor itself and if one validator is clearly non-performant, then the other validators should score that validator with a 0 and slash its rewards. Community members can launch public dashboards tracking validator performance and that can be used as further signal into a validator's operations. There is no limit on the number of validators that can receive a 0 tallying score in an epoch.

Requirements to run a validator on Sui

To run a Sui validator, you must set up and configure a Sui Validator node. After you have a running node, you must have a minimum of 30 million SUI in your staking pool to join the validator set on the Sui network.

To learn how to set up and configure a Sui Validator node, see Sui for Node Operators on GitHub. The guide includes all of the information you need to configure your Validator node. It also provides guidance on the tasks you must perform after you join the validator set.

Specific steps you must take include:

There are minimum staking requirements a validator must satisfy to become active and to stay in the active validator set.

More precisely:

Suggested minimum hardware specifications to run a Sui Validator node:

The total voting power on Sui is always 10,000, regardless of the amount staked. Therefore, the quorum threshold is 6,667. There is no limit to the amount of SUI users can stake with a validator. Each validator has consensus voting power proportional to SUI in its staking pool, with one exception: the voting power of an individual validator is capped at 1,000 (10% of the total). If a validator accumulates more than 10% of total stake, the validator's voting power remains fixed at 10%, and the remaining voting power is spread across the rest of the validator set.

When users stake SUI tokens, these SUI objects are wrapped into StakedSUI objects. The calculation to determine each user's relative ownership of the staking pool is done directly with the timestamp of the StakedSUI object (which determines the moment at which the deposit took place) and the change in the exchange rates between the deposit epoch and the withdrawal epoch. Each staking pool's data structure contains a time series with that pool's exchange rates. These exchange rates can be used to determine the withdrawals of any of the pool's stakers.

Stake withdrawals are processed immediately with the exchange rate prevailing at the previous epoch's exchange rate. Withdrawals do not have to wait for the current epoch to close. Withdrawals include both the original stake the user deposited and all the stake rewards accumulated up to the previous epoch. Stakers do not earn the rewards accruing to their stake during the epoch at which they withdraw. Since there is no way to know how many stake rewards will be accumulated during the current epoch until the epoch closes, these cannot be included in the withdrawal. Hence, any user can withdraw their stake immediately and receive:

SUI withdrawn at E' = (SUI deposited at E) * (Exchange Rate at E'-1 / Exchange Rate at E)

Each epoch change emits a 0x2::validator_set::ValidatorEpochInfo event per validator with the exchange rate information. You can use the Events API to query events.

Within a given validator staking pool, all stakers receive the same proportion of rewards through the pool's exchange rate appreciation. In addition, since validators earn commissions over the stake they manage, validators receive additional StakedSUI objects at the end of each epoch in proportion to the amount of commissions their staking pool earns.

Staking rewards are funded by transaction gas fees collected during the current epoch and by stake subsidies released at the end of the epoch.

StakeRewards = StakeSubsidies + GasFees

Stake subsidies are intended to subsidize the network during its early phases and are funded by a 10% allocation of SUI tokens. After this allocation depletes, the entirety of stake rewards will be made up of gas fees collected through regular network operations.

Stake rewards are made up of gas fees and stake subsidies. The total amount distributed throughout each epoch is determined as follows:

The total amount of gas fees collected corresponds to the sum of gas fees across all transactions processed in the epoch. During regular market conditions, the vast majority of transactions should have a GasPrice equal to the ReferenceGasPrice .

A stake deposit request goes into a pending state immediately in the staking pool as soon as it is made. Sui Wallet reflects any pending stake deposit requests for the user's account. However, pending stake deposit requests do not take effect until the end of the epoch during which the request is made.

A withdrawal (un-stake) request is processed immediately as soon as it is received. The staker obtains the originally deposited SUI together with all accrued stake rewards up to the previous epoch boundary – in other words, they do not include stake rewards for the current epoch.

Users can't withdraw a portion of their active stake. They must withdraw all staked SUI at the same time. Users can, however, stake using multiple StakedSui objects by splitting their SUI into multiple coins. They can then perform a partial withdrawal from a validator by un-staking only some of the StakedSUI objects.

Sui is designed such that end-users can expect the gas price to be stable and predictable during regular network operations. This is achieved by having validators set the network's reference gas price at the beginning of each epoch.

Operationally this is achieved through a gas price survey that occurs as follows:

For example, assume that there are seven validators with equal stake, and the price quotes they submit are {15, 1, 4, 2, 8, 3, 23} . The protocol sets the reference gas price at 8.

In practice, the process for submitting a gas price quote for the Gas Price Survey is a straightforward one. Each validator owns an object that contains their quote for the reference gas price. To change their response, they must update the value in that object.

For example, to set the price quote for the next epoch to 42, run:

Beginning with the Sui v1.24.1 release , the --gas-budget option is no longer required for CLI commands.

Importantly, the gas object's value persists across epochs so that a validator who does not update and submit a new quote uses the same quote from the previous epoch. Hence, a validator seeking to optimize its own operations should update its quote every epoch in response to changes in network operations and market conditions.

Sui is designed to encourage and enforce community monitoring of the validator set. This is done through the Tallying Rule by which each validator monitors and scores every other validator in order to ensure that everyone is operating efficiently and in the network's best interest. Validators that receive a low score can be penalized with slashed stake rewards.

The protocol only computes the global Tallying Rule score at the epoch boundary and so relies on validators monitoring actively and changing their individual scores whenever they detect changes in other validator behavior. In general, the Tallying Rule default option should always be a score of one for all validators and only be changed to zero upon determining bad operations. In practice, the Tallying Rule consists of a set of objects each validator owns that default to scores of one and thus a validator will generally be passive and only update the object corresponding to another validator's score whenever needed.

For example, to report a validator whose Sui address is 0x44840a79dd5cf1f5efeff1379f5eece04c72db13512a2e31e8750f5176285446 as bad or non-performant, run:

The Tallying Rule should be implemented through a social equilibrium. The validator set should actively monitor itself and if one validator is clearly non-performant, then the other validators should score that validator with a 0 and slash its rewards. Community members can launch public dashboards tracking validator performance and that can be used as further signal into a validator's operations. There is no limit on the number of validators that can receive a 0 tallying score in an epoch.

Hardware requirements to run a Validator node

Suggested minimum hardware specifications to run a Sui Validator node:

The total voting power on Sui is always 10,000, regardless of the amount staked. Therefore, the quorum threshold is 6,667. There is no limit to the amount of SUI users can stake with a validator. Each validator has consensus voting power proportional to SUI in its staking pool, with one exception: the voting power of an individual validator is capped at 1,000 (10% of the total). If a validator accumulates more than 10% of total stake, the validator's voting power remains fixed at 10%, and the remaining voting power is spread across the rest of the validator set.

When users stake SUI tokens, these SUI objects are wrapped into StakedSUI objects. The calculation to determine each user's relative ownership of the staking pool is done directly with the timestamp of the StakedSUI object (which determines the moment at which the deposit took place) and the change in the exchange rates between the deposit epoch and the withdrawal epoch. Each staking pool's data structure contains a time series with that pool's exchange rates. These exchange rates can be used to determine the withdrawals of any of the pool's stakers.

Stake withdrawals are processed immediately with the exchange rate prevailing at the previous epoch's exchange rate. Withdrawals do not have to wait for the current epoch to close. Withdrawals include both the original stake the user deposited and all the stake rewards accumulated up to the previous epoch. Stakers do not earn the rewards accruing to their stake during the epoch at which they withdraw. Since there is no way to know how many stake rewards will be accumulated during the current epoch until the epoch closes, these cannot be included in the withdrawal. Hence, any user can withdraw their stake immediately and receive:

SUI withdrawn at E' = (SUI deposited at E) * (Exchange Rate at E'-1 / Exchange Rate at E)

Each epoch change emits a 0x2::validator_set::ValidatorEpochInfo event per validator with the exchange rate information. You can use the Events API to query events.

Within a given validator staking pool, all stakers receive the same proportion of rewards through the pool's exchange rate appreciation. In addition, since validators earn commissions over the stake they manage, validators receive additional StakedSUI objects at the end of each epoch in proportion to the amount of commissions their staking pool earns.

Staking rewards are funded by transaction gas fees collected during the current epoch and by stake subsidies released at the end of the epoch.

StakeRewards = StakeSubsidies + GasFees

Stake subsidies are intended to subsidize the network during its early phases and are funded by a 10% allocation of SUI tokens. After this allocation depletes, the entirety of stake rewards will be made up of gas fees collected through regular network operations.

Stake rewards are made up of gas fees and stake subsidies. The total amount distributed throughout each epoch is determined as follows:

The total amount of gas fees collected corresponds to the sum of gas fees across all transactions processed in the epoch. During regular market conditions, the vast majority of transactions should have a GasPrice equal to the ReferenceGasPrice .

A stake deposit request goes into a pending state immediately in the staking pool as soon as it is made. Sui Wallet reflects any pending stake deposit requests for the user's account. However, pending stake deposit requests do not take effect until the end of the epoch during which the request is made.

A withdrawal (un-stake) request is processed immediately as soon as it is received. The staker obtains the originally deposited SUI together with all accrued stake rewards up to the previous epoch boundary – in other words, they do not include stake rewards for the current epoch.

Users can't withdraw a portion of their active stake. They must withdraw all staked SUI at the same time. Users can, however, stake using multiple StakedSui objects by splitting their SUI into multiple coins. They can then perform a partial withdrawal from a validator by un-staking only some of the StakedSUI objects.

Sui is designed such that end-users can expect the gas price to be stable and predictable during regular network operations. This is achieved by having validators set the network's reference gas price at the beginning of each epoch.

Operationally this is achieved through a gas price survey that occurs as follows:

For example, assume that there are seven validators with equal stake, and the price quotes they submit are {15, 1, 4, 2, 8, 3, 23} . The protocol sets the reference gas price at 8.

In practice, the process for submitting a gas price quote for the Gas Price Survey is a straightforward one. Each validator owns an object that contains their quote for the reference gas price. To change their response, they must update the value in that object.

For example, to set the price quote for the next epoch to 42, run:

Beginning with the Sui v1.24.1 release , the --gas-budget option is no longer required for CLI commands.

Importantly, the gas object's value persists across epochs so that a validator who does not update and submit a new quote uses the same quote from the previous epoch. Hence, a validator seeking to optimize its own operations should update its quote every epoch in response to changes in network operations and market conditions.

Sui is designed to encourage and enforce community monitoring of the validator set. This is done through the Tallying Rule by which each validator monitors and scores every other validator in order to ensure that everyone is operating efficiently and in the network's best interest. Validators that receive a low score can be penalized with slashed stake rewards.

The protocol only computes the global Tallying Rule score at the epoch boundary and so relies on validators monitoring actively and changing their individual scores whenever they detect changes in other validator behavior. In general, the Tallying Rule default option should always be a score of one for all validators and only be changed to zero upon determining bad operations. In practice, the Tallying Rule consists of a set of objects each validator owns that default to scores of one and thus a validator will generally be passive and only update the object corresponding to another validator's score whenever needed.

For example, to report a validator whose Sui address is 0x44840a79dd5cf1f5efeff1379f5eece04c72db13512a2e31e8750f5176285446 as bad or non-performant, run:

The Tallying Rule should be implemented through a social equilibrium. The validator set should actively monitor itself and if one validator is clearly non-performant, then the other validators should score that validator with a 0 and slash its rewards. Community members can launch public dashboards tracking validator performance and that can be used as further signal into a validator's operations. There is no limit on the number of validators that can receive a 0 tallying score in an epoch.

Validator consensus and voting power

The total voting power on Sui is always 10,000, regardless of the amount staked. Therefore, the quorum threshold is 6,667. There is no limit to the amount of SUI users can stake with a validator. Each validator has consensus voting power proportional to SUI in its staking pool, with one exception: the voting power of an individual validator is capped at 1,000 (10% of the total). If a validator accumulates more than 10% of total stake, the validator's voting power remains fixed at 10%, and the remaining voting power is spread across the rest of the validator set.

When users stake SUI tokens, these SUI objects are wrapped into StakedSUI objects. The calculation to determine each user's relative ownership of the staking pool is done directly with the timestamp of the StakedSUI object (which determines the moment at which the deposit took place) and the change in the exchange rates between the deposit epoch and the withdrawal epoch. Each staking pool's data structure contains a time series with that pool's exchange rates. These exchange rates can be used to determine the withdrawals of any of the pool's stakers.

Stake withdrawals are processed immediately with the exchange rate prevailing at the previous epoch's exchange rate. Withdrawals do not have to wait for the current epoch to close. Withdrawals include both the original stake the user deposited and all the stake rewards accumulated up to the previous epoch. Stakers do not earn the rewards accruing to their stake during the epoch at which they withdraw. Since there is no way to know how many stake rewards will be accumulated during the current epoch until the epoch closes, these cannot be included in the withdrawal. Hence, any user can withdraw their stake immediately and receive:

SUI withdrawn at E' = (SUI deposited at E) * (Exchange Rate at E'-1 / Exchange Rate at E)

Each epoch change emits a 0x2::validator_set::ValidatorEpochInfo event per validator with the exchange rate information. You can use the Events API to query events.

Within a given validator staking pool, all stakers receive the same proportion of rewards through the pool's exchange rate appreciation. In addition, since validators earn commissions over the stake they manage, validators receive additional StakedSUI objects at the end of each epoch in proportion to the amount of commissions their staking pool earns.

Staking rewards are funded by transaction gas fees collected during the current epoch and by stake subsidies released at the end of the epoch.

StakeRewards = StakeSubsidies + GasFees

Stake subsidies are intended to subsidize the network during its early phases and are funded by a 10% allocation of SUI tokens. After this allocation depletes, the entirety of stake rewards will be made up of gas fees collected through regular network operations.

Stake rewards are made up of gas fees and stake subsidies. The total amount distributed throughout each epoch is determined as follows:

The total amount of gas fees collected corresponds to the sum of gas fees across all transactions processed in the epoch. During regular market conditions, the vast majority of transactions should have a GasPrice equal to the ReferenceGasPrice .

A stake deposit request goes into a pending state immediately in the staking pool as soon as it is made. Sui Wallet reflects any pending stake deposit requests for the user's account. However, pending stake deposit requests do not take effect until the end of the epoch during which the request is made.

A withdrawal (un-stake) request is processed immediately as soon as it is received. The staker obtains the originally deposited SUI together with all accrued stake rewards up to the previous epoch boundary – in other words, they do not include stake rewards for the current epoch.

Users can't withdraw a portion of their active stake. They must withdraw all staked SUI at the same time. Users can, however, stake using multiple StakedSui objects by splitting their SUI into multiple coins. They can then perform a partial withdrawal from a validator by un-staking only some of the StakedSUI objects.

Sui is designed such that end-users can expect the gas price to be stable and predictable during regular network operations. This is achieved by having validators set the network's reference gas price at the beginning of each epoch.

Operationally this is achieved through a gas price survey that occurs as follows:

For example, assume that there are seven validators with equal stake, and the price quotes they submit are {15, 1, 4, 2, 8, 3, 23} . The protocol sets the reference gas price at 8.

In practice, the process for submitting a gas price quote for the Gas Price Survey is a straightforward one. Each validator owns an object that contains their quote for the reference gas price. To change their response, they must update the value in that object.

For example, to set the price quote for the next epoch to 42, run:

Beginning with the Sui v1.24.1 release , the --gas-budget option is no longer required for CLI commands.

Importantly, the gas object's value persists across epochs so that a validator who does not update and submit a new quote uses the same quote from the previous epoch. Hence, a validator seeking to optimize its own operations should update its quote every epoch in response to changes in network operations and market conditions.

Sui is designed to encourage and enforce community monitoring of the validator set. This is done through the Tallying Rule by which each validator monitors and scores every other validator in order to ensure that everyone is operating efficiently and in the network's best interest. Validators that receive a low score can be penalized with slashed stake rewards.

The protocol only computes the global Tallying Rule score at the epoch boundary and so relies on validators monitoring actively and changing their individual scores whenever they detect changes in other validator behavior. In general, the Tallying Rule default option should always be a score of one for all validators and only be changed to zero upon determining bad operations. In practice, the Tallying Rule consists of a set of objects each validator owns that default to scores of one and thus a validator will generally be passive and only update the object corresponding to another validator's score whenever needed.

For example, to report a validator whose Sui address is 0x44840a79dd5cf1f5efeff1379f5eece04c72db13512a2e31e8750f5176285446 as bad or non-performant, run:

The Tallying Rule should be implemented through a social equilibrium. The validator set should actively monitor itself and if one validator is clearly non-performant, then the other validators should score that validator with a 0 and slash its rewards. Community members can launch public dashboards tracking validator performance and that can be used as further signal into a validator's operations. There is no limit on the number of validators that can receive a 0 tallying score in an epoch.

Staking rewards

Within a given validator staking pool, all stakers receive the same proportion of rewards through the pool's exchange rate appreciation. In addition, since validators earn commissions over the stake they manage, validators receive additional StakedSUI objects at the end of each epoch in proportion to the amount of commissions their staking pool earns.

Staking rewards are funded by transaction gas fees collected during the current epoch and by stake subsidies released at the end of the epoch.

StakeRewards = StakeSubsidies + GasFees

Stake subsidies are intended to subsidize the network during its early phases and are funded by a 10% allocation of SUI tokens. After this allocation depletes, the entirety of stake rewards will be made up of gas fees collected through regular network operations.

Stake rewards are made up of gas fees and stake subsidies. The total amount distributed throughout each epoch is determined as follows:

The total amount of gas fees collected corresponds to the sum of gas fees across all transactions processed in the epoch. During regular market conditions, the vast majority of transactions should have a GasPrice equal to the ReferenceGasPrice .

A stake deposit request goes into a pending state immediately in the staking pool as soon as it is made. Sui Wallet reflects any pending stake deposit requests for the user's account. However, pending stake deposit requests do not take effect until the end of the epoch during which the request is made.

A withdrawal (un-stake) request is processed immediately as soon as it is received. The staker obtains the originally deposited SUI together with all accrued stake rewards up to the previous epoch boundary – in other words, they do not include stake rewards for the current epoch.

Users can't withdraw a portion of their active stake. They must withdraw all staked SUI at the same time. Users can, however, stake using multiple StakedSui objects by splitting their SUI into multiple coins. They can then perform a partial withdrawal from a validator by un-staking only some of the StakedSUI objects.

Sui is designed such that end-users can expect the gas price to be stable and predictable during regular network operations. This is achieved by having validators set the network's reference gas price at the beginning of each epoch.

Operationally this is achieved through a gas price survey that occurs as follows:

For example, assume that there are seven validators with equal stake, and the price quotes they submit are {15, 1, 4, 2, 8, 3, 23} . The protocol sets the reference gas price at 8.

In practice, the process for submitting a gas price quote for the Gas Price Survey is a straightforward one. Each validator owns an object that contains their quote for the reference gas price. To change their response, they must update the value in that object.

For example, to set the price quote for the next epoch to 42, run:

Beginning with the Sui v1.24.1 release , the --gas-budget option is no longer required for CLI commands.

Importantly, the gas object's value persists across epochs so that a validator who does not update and submit a new quote uses the same quote from the previous epoch. Hence, a validator seeking to optimize its own operations should update its quote every epoch in response to changes in network operations and market conditions.

Sui is designed to encourage and enforce community monitoring of the validator set. This is done through the Tallying Rule by which each validator monitors and scores every other validator in order to ensure that everyone is operating efficiently and in the network's best interest. Validators that receive a low score can be penalized with slashed stake rewards.

The protocol only computes the global Tallying Rule score at the epoch boundary and so relies on validators monitoring actively and changing their individual scores whenever they detect changes in other validator behavior. In general, the Tallying Rule default option should always be a score of one for all validators and only be changed to zero upon determining bad operations. In practice, the Tallying Rule consists of a set of objects each validator owns that default to scores of one and thus a validator will generally be passive and only update the object corresponding to another validator's score whenever needed.

For example, to report a validator whose Sui address is 0x44840a79dd5cf1f5efeff1379f5eece04c72db13512a2e31e8750f5176285446 as bad or non-performant, run:

The Tallying Rule should be implemented through a social equilibrium. The validator set should actively monitor itself and if one validator is clearly non-performant, then the other validators should score that validator with a 0 and slash its rewards. Community members can launch public dashboards tracking validator performance and that can be used as further signal into a validator's operations. There is no limit on the number of validators that can receive a 0 tallying score in an epoch.

Reference gas price

Sui is designed such that end-users can expect the gas price to be stable and predictable during regular network operations. This is achieved by having validators set the network's reference gas price at the beginning of each epoch.

Operationally this is achieved through a gas price survey that occurs as follows:

For example, assume that there are seven validators with equal stake, and the price quotes they submit are {15, 1, 4, 2, 8, 3, 23} . The protocol sets the reference gas price at 8.

In practice, the process for submitting a gas price quote for the Gas Price Survey is a straightforward one. Each validator owns an object that contains their quote for the reference gas price. To change their response, they must update the value in that object.

For example, to set the price quote for the next epoch to 42, run:

Beginning with the Sui v1.24.1 release , the --gas-budget option is no longer required for CLI commands.

Importantly, the gas object's value persists across epochs so that a validator who does not update and submit a new quote uses the same quote from the previous epoch. Hence, a validator seeking to optimize its own operations should update its quote every epoch in response to changes in network operations and market conditions.

Sui is designed to encourage and enforce community monitoring of the validator set. This is done through the Tallying Rule by which each validator monitors and scores every other validator in order to ensure that everyone is operating efficiently and in the network's best interest. Validators that receive a low score can be penalized with slashed stake rewards.

The protocol only computes the global Tallying Rule score at the epoch boundary and so relies on validators monitoring actively and changing their individual scores whenever they detect changes in other validator behavior. In general, the Tallying Rule default option should always be a score of one for all validators and only be changed to zero upon determining bad operations. In practice, the Tallying Rule consists of a set of objects each validator owns that default to scores of one and thus a validator will generally be passive and only update the object corresponding to another validator's score whenever needed.

For example, to report a validator whose Sui address is 0x44840a79dd5cf1f5efeff1379f5eece04c72db13512a2e31e8750f5176285446 as bad or non-performant, run:

The Tallying Rule should be implemented through a social equilibrium. The validator set should actively monitor itself and if one validator is clearly non-performant, then the other validators should score that validator with a 0 and slash its rewards. Community members can launch public dashboards tracking validator performance and that can be used as further signal into a validator's operations. There is no limit on the number of validators that can receive a 0 tallying score in an epoch.

Validator slashing and tallying rule

Sui is designed to encourage and enforce community monitoring of the validator set. This is done through the Tallying Rule by which each validator monitors and scores every other validator in order to ensure that everyone is operating efficiently and in the network's best interest. Validators that receive a low score can be penalized with slashed stake rewards.

The protocol only computes the global Tallying Rule score at the epoch boundary and so relies on validators monitoring actively and changing their individual scores whenever they detect changes in other validator behavior. In general, the Tallying Rule default option should always be a score of one for all validators and only be changed to zero upon determining bad operations. In practice, the Tallying Rule consists of a set of objects each validator owns that default to scores of one and thus a validator will generally be passive and only update the object corresponding to another validator's score whenever needed.

For example, to report a validator whose Sui address is 0x44840a79dd5cf1f5efeff1379f5eece04c72db13512a2e31e8750f5176285446 as bad or non-performant, run:

The Tallying Rule should be implemented through a social equilibrium. The validator set should actively monitor itself and if one validator is clearly non-performant, then the other validators should score that validator with a 0 and slash its rewards. Community members can launch public dashboards tracking validator performance and that can be used as further signal into a validator's operations. There is no limit on the number of validators that can receive a 0 tallying score in an epoch.