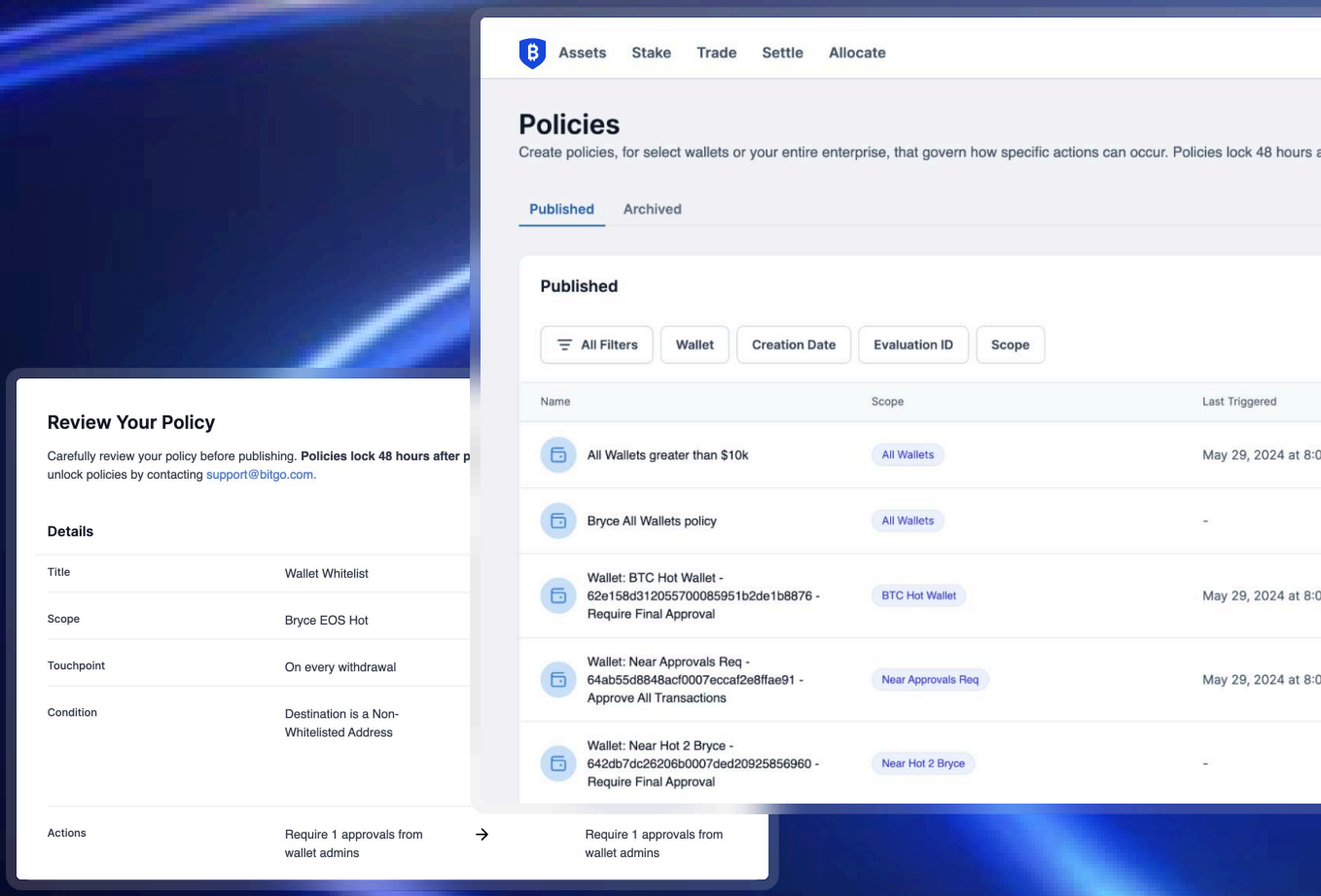
BitGo

User Guide

Policy Engine



Last updated: 2025 February

Table of Contents

Table of Contents

Table of Contents

Policy Engine User Guide

What is the Policy Engine?

The Policy Engine is a tool to build and manage policies that protect your funds and streamline operations. The policies are customizable to fit your needs. You can ensure standard and consistent policies across all of your wallets. There's also a full audit log history of every policy created and updated.

Why is good Policy Engine hygiene important?

A well tuned policy engine empowers companies to define, update, and apply policies consistently, ensuring security, compliance, and operational efficiency.

With precise access controls on your financial infrastructure, a policy engine acts as an automated decision-making hub, greatly reducing both human error and malicious actions.

Whether for security or compliance, BitGo's Policy Engine gives organizations control over their assets and employee actions with customizable policies and workflows.

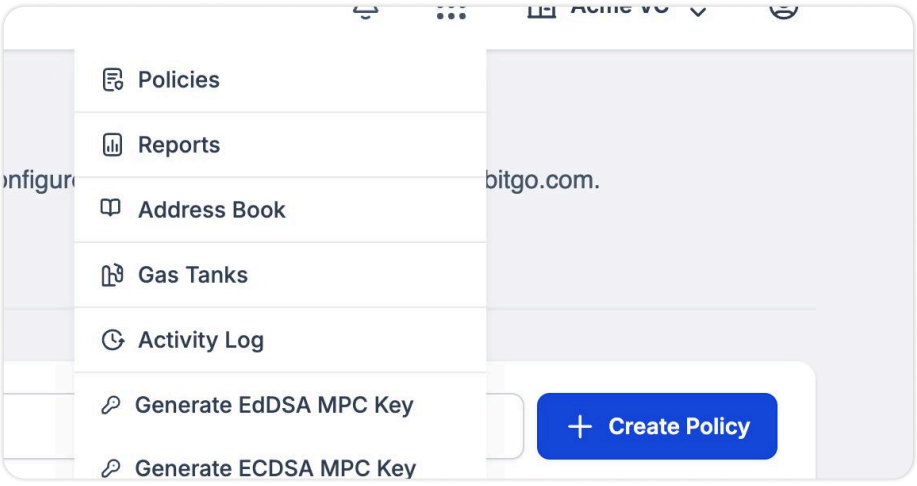
Quickstart Guide: Create Your First Policy

Get Started

To start, let's walk through an example of how to create a policy that will apply to all wallets in your enterprise.

STEP 1

Once logged into app.bitgo.com click on the 9 dot icon in top right corner of the dashboard and select "Policies"

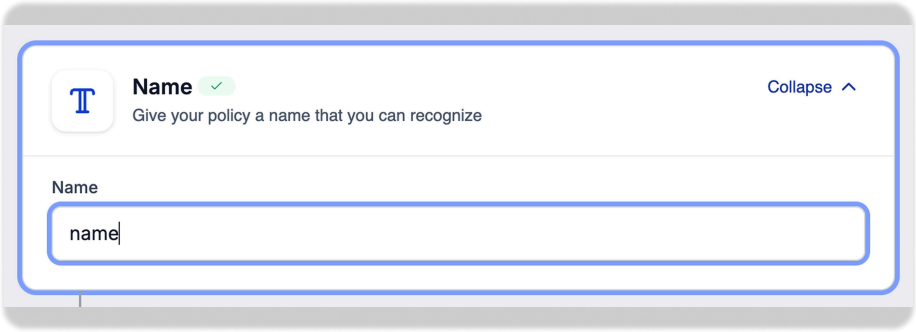


STEP 2

Click "Create Policy

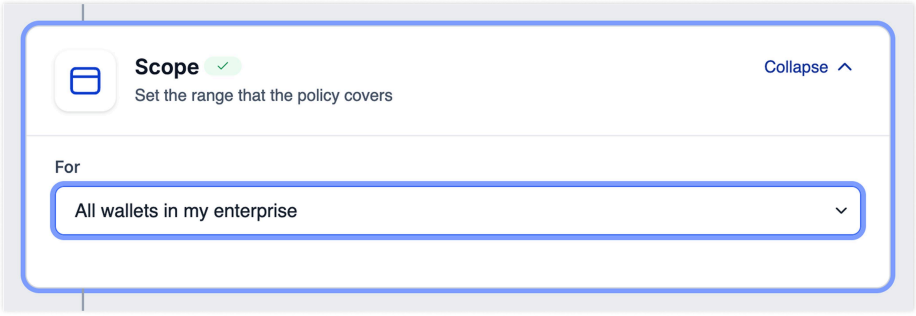
STEP 3

For the sake of this quickstart guide, let's create a policy that will prevent sending to a non-whitelisted address from any wallet in your enterprise. Start by naming your policy.



STEP 4

In the Scope field, choose that this policy will affect "All wallets in my enterprise." *Note: If you want to create policies that apply to specific wallets as opposed to all of them, you must have created at least one wallet.

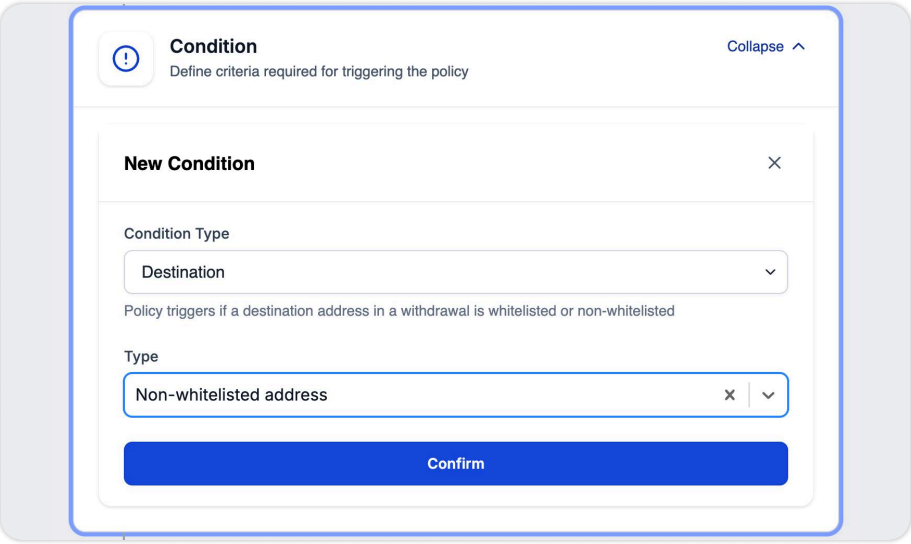


STEP 5

In the Touchpoint field, select that this policy will trigger on every attempted "Withdrawal."

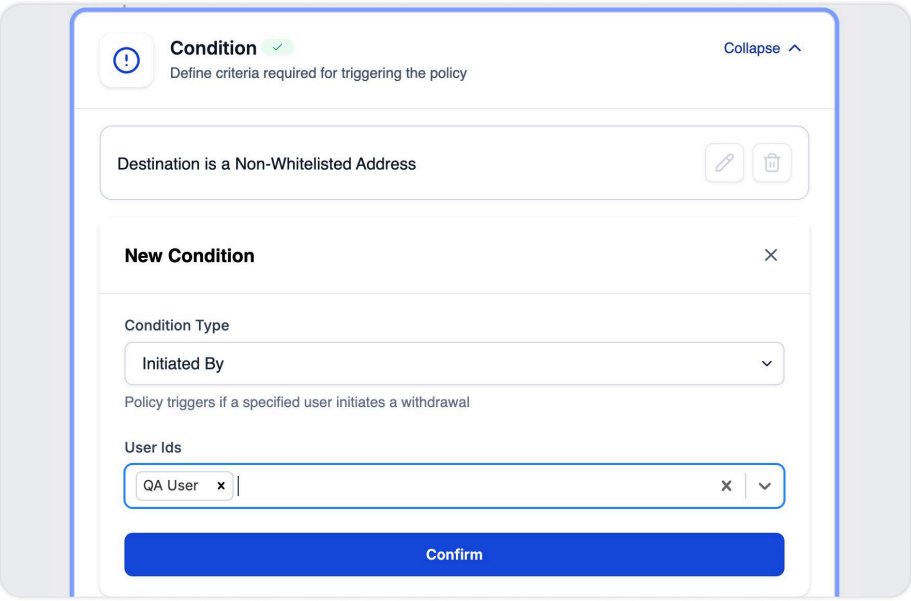
STEP 6

Click “New Condition” to define the criteria that will trigger the policy. For this example, we will create a rule that will prevent a specific user from sending funds to any non-whitelisted address. Because we are going to limit who is receiving transfers from this wallet, we select “Destination.” Then, select “Non-whitelisted address” from the “Type” dropdown, and click “Confirm.”

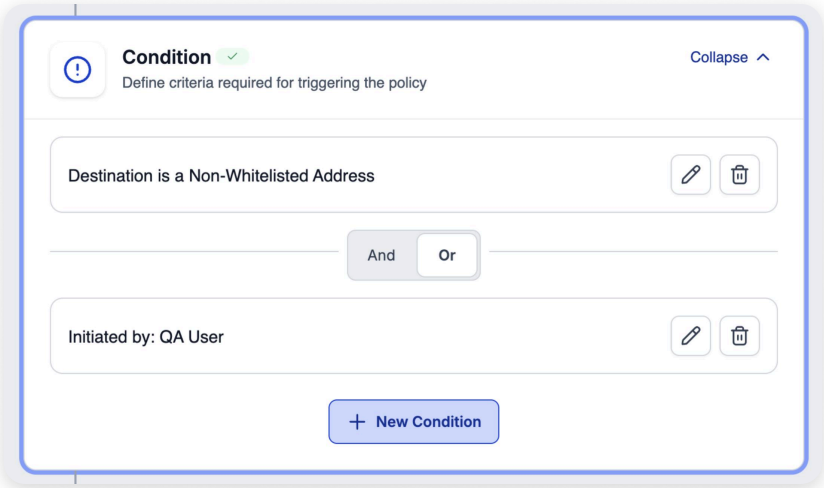


STEP 6B

You can build a policy with as many conditions as you need. For this example, we want to add an additional condition that blocks a specific user (QA User) from sending any funds from this wallet. Start in the same way - by clicking “New Condition.” Select "Initiated By” and the user’s name (here “QA User”) for the Condition Type and User Ids fields, respectively.

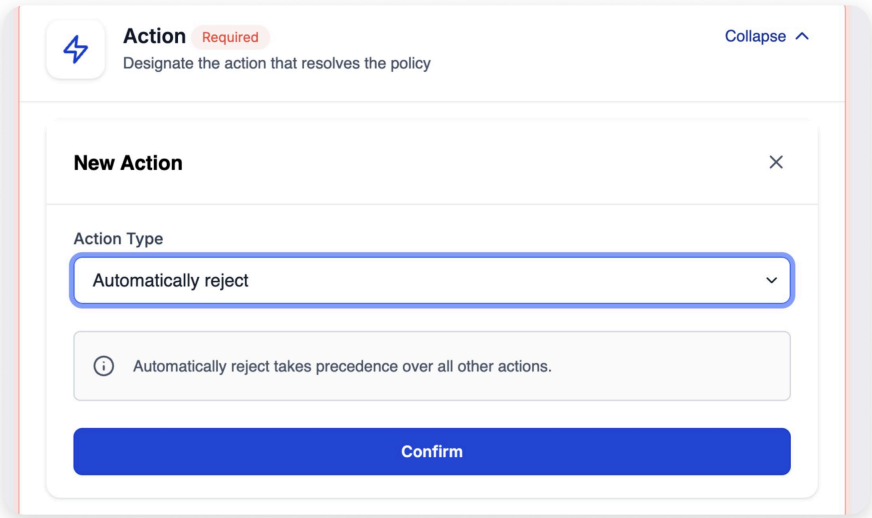


Clicking “Confirm” will create the condition and opens the “And/Or” option. Selecting AND means all conditions must be met. Selecting OR means at least 1 condition must be met. Here we will select “or” because we want the condition to be applied if either of those conditions are met.



STEP 7

Next, we decide what the action is if the conditions are met. Click “New Action.” Select the “Automatically reject” Action Type from the drop down menu and click “Confirm.” To move to the final review, click “Continue.”



STEP 8

Now that all of the conditions have been set, click “Continue” at the bottom of the page to see a summary of the policy.

If everything looks good, click “Publish Policy.”

And that’s it - congratulations on creating your first policy! Be sure to read through the rest of this guide to find the most efficient way to automate your policies and protect your assets.

How Does It Work?

It's important to understand the individual components that make up a policy. These are the building blocks that enable you to create the exact policy you need to protect your assets.

Components of a Policy:

- **Scope** - the coverage of the policy.
- **Touchpoint** - the operation that triggers the policy.
- **Condition** - required criteria that leads to specified actions.
- **Action** - something that must occur in order for the operation to proceed.

Examples of Each Component

Scope

- All wallets in your enterprise
- Wallets by type
    i. Hot
    ii. Self-managed cold
    iii. Custody
- Single wallet
- Multiple wallets
- Go Account

Note: We do not support V1 BTC wallets

Touchpoint

- Withdrawal
- Settlement initiated from Go Account
- Settlement approved from Go Account

Condition

- **Spending limit** - e.g., withdrawals greater than $10k
- **Velocity limit** - e.g., greater than $100k in withdrawals within the last 24 hours
- **Destination** - e.g., withdrawals to non-whitelisted addresses
- **Initiated by** - e.g., withdrawals initiated by specific users in the enterprise
- **Asset** - e.g., withdrawals of BTC, ETH, or USDT.
- **Webhook** - e.g., webhook response was 202, resulting in actions required

Action

Eligible Actions

Wallet admin approvals

- Eligible approvers: any admin on the wallet
- Dynamic or static: dynamic as the list of admins on the wallet changes based on adding or removing admins from the wallet
- Number of approvals: specified during policy creation

Final approver

- Eligible approvers: any user on the wallet
- Dynamic or static: static as the list is provided during policy creation
- Number of approvals: has to be 1

Note: this respects sequences and will be the final user approval action

Set of users' approval

- Eligible approvers: any user in the enterprise
- Dynamic or static: static as the list is provided during policy creation
- Number of approvals: has to be 1
- Number of approvals: specified during policy creation

Note: enterprise must have the View All Wallets option enabled from the enterprise settings

Automatically reject

AND/OR

- You can string together actions with an operator in between
- **Example:**
    i. Require 3 wallet admin approvals AND 1 final approval
    ii. Require 3 wallet admin approvals OR 2 approvals from a set of users

Ordering of Actions

- When a single policy or multiple policies trigger because of a transaction, BitGo aggregates and deduplicates all actions. At that point, all but the final approval can be done simultaneously.
- The one action type that does have a sequence is the final approval. The wallet admins must first approve the transaction before a final approver is able to provide their approval

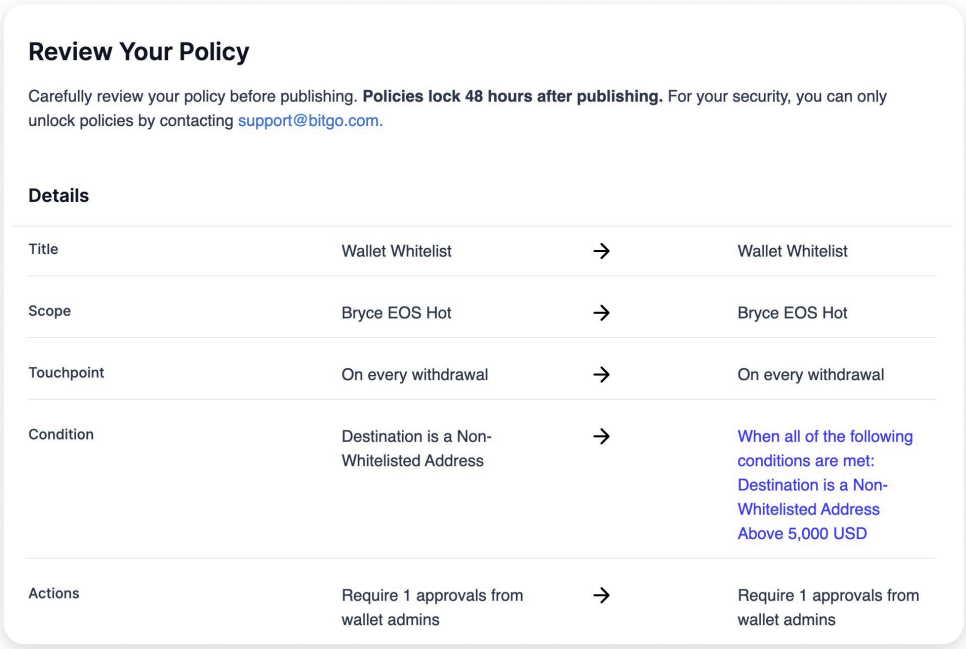
Whitelists

A whitelist helps ensure that withdrawals only go to specific trusted addresses.

- We recommend that you manage the whitelist within each wallet
- In most cases, the enterprise level whitelist is overly restrictive for the desired behavior. As such, we recommend avoiding the enterprise whitelist unless you absolutely need to.
- Whitelists have slightly different behavior per wallet type:
    i. Custody wallets and Go Account - BitGo requires that all withdrawals from Custody wallets go to whitelisted addresses. As such, there is no need for you to create a specific policy for this and all you need to do is add/remove the addresses.
    ii. Self-managed hot / cold wallets - When you add an address to the whitelist of one of these wallets, a corresponding enforcement policy is automatically created for you. You can then customize this policy to your liking (example below)

Making changes to the enforcement policy:

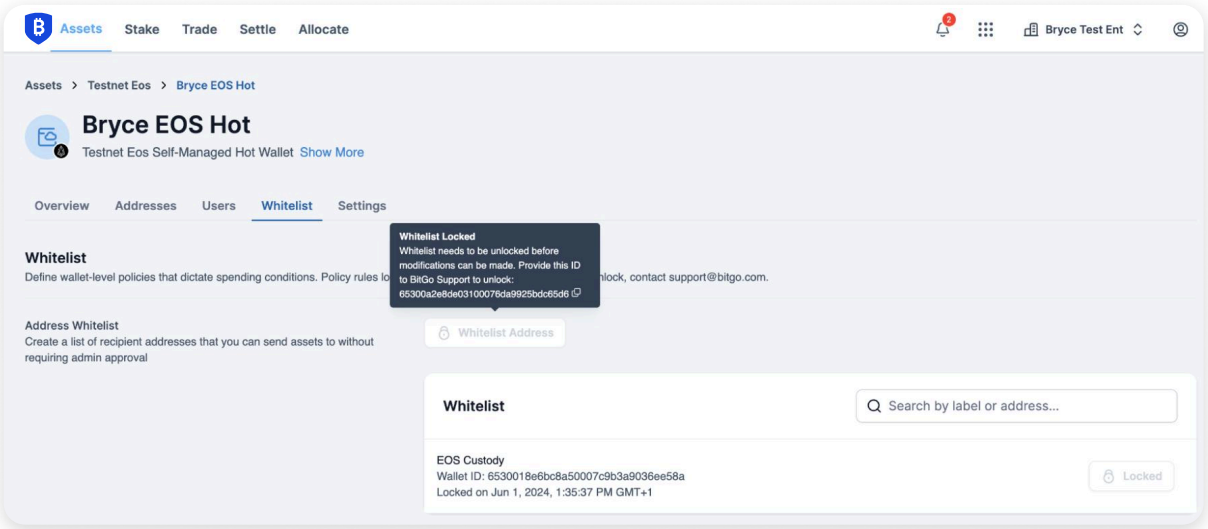
- This works like any other policy described in this guide, you will need to reach out to BitGo support to unlock the policy before making any changes
- In the example below, I am modifying the whitelist policy that was automatically created for a hot wallet to only apply if both of these are true:
    i. Destination is a non-whitelisted address
    ii. Withdrawal is above $5,000



Making changes to the whitelist (i.e., actual list of addresses):

- Custody wallets & Go Account - The whitelist on these wallets does not lock, meaning addresses can be added/removed any time. If there is at least one other wallet admin, that admin must approve these changes.
- Self-managed hot / cold - The whitelists on these wallets do lock, meaning you must contact BitGo support to unlock the whitelist before adding/removing any addresses. If there is more than one admin, the other admin must approve these changes.

Example of a hot wallet whitelist that is currently locked and requires unlock before any addresses:



Evaluating Policies

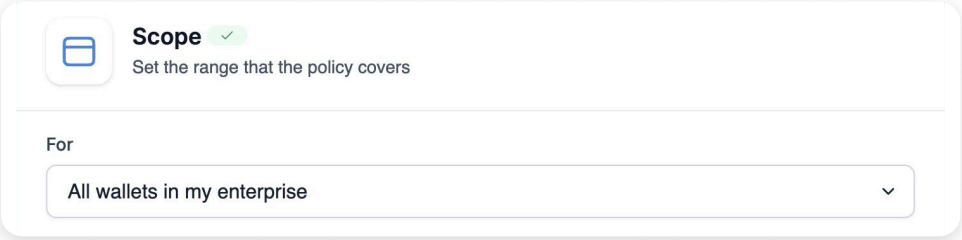
Every time there's a transaction initiated from a wallet, BitGo evaluates all of the policies within the enterprise:

- The policies that don't apply (i.e., the scope does not match), are disregarded.
    i. Eg.., a withdrawal from a hot wallet would not match a policy covering custodial wallets.
    ii. E.g., a withdrawal from WalletX wouldn't match a policy covering WalletY.
- Once the policies are filtered down based on scope, BitGo evaluates each condition from every policy.
    i. E.g., has a spending limit been exceeded?
    ii. E.g., was the withdrawal initiated by a specific user?
    iii. If there are ANDs/ORs in the conditions, BitGo evaluates these.
- Once the policies are filtered down a second time based on the conditions, the associated actions from each policy trigger. If there are redundant actions, BitGo deduplicates them.
    i. E.g., Require 1 wallet admin approval
    ii. E.g., Require 1 wallet admin approval or 2 approvals from userA, userB, or userC
- If any of the triggered policies contain the action Automatically Reject, all other actions will be trumped and the entire withdrawal will be rejected

Scopes

All Wallets

Description: Policies with this scope automatically apply to all existing and new wallets in the enterprise, with the exception of the Go Account.

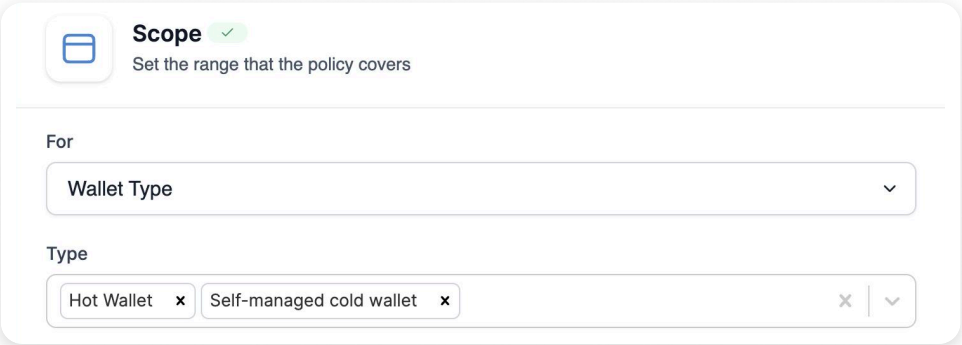


Wallet Type

Description: Policies with this scope automatically apply to all existing and new wallets in the enterprise that match the type, with the exception of the Go Account.

- Self-managed hot
- Self-managed cold
- Custodial

Note: this respects sequences and will be the final user approval action



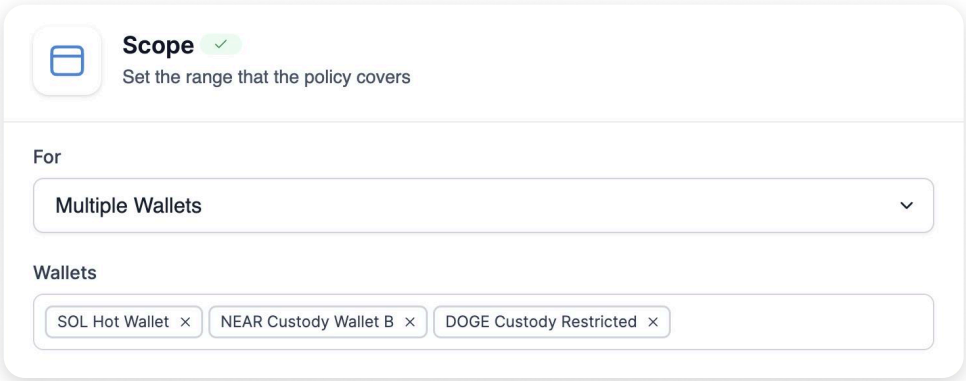
Single Wallet

Decription: Policies with this scope only apply to the selected wallet.

Multiple Wallets

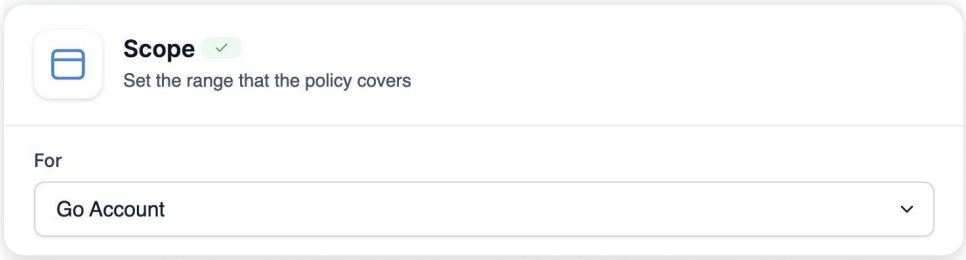
Description: Policies with this scope apply to any of the selected wallets

Example: Many clients want to “group” their wallets within a policy to ensure the policy is applied in a consistent approach across all wallets specified. For example, setting up a single policy that covers all standby wallets - which are technically self-managed hot wallets.



Go Account

Description: Policies with this scope only apply to the Go Account

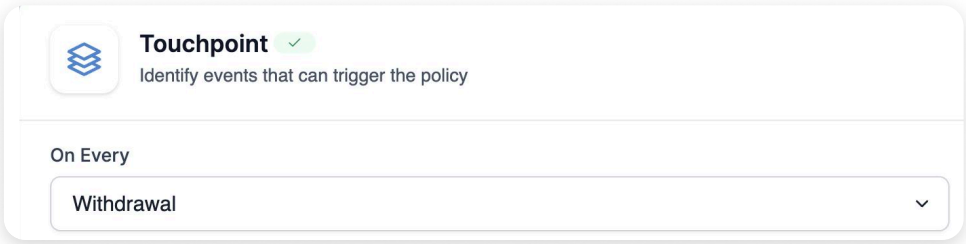


Touchpoints

Withdrawal

Description: This touchpoint applies to all withdrawals from the selected scope.

Note: Some staking transactions apply to withdrawal policies. BitGo plans to separate withdrawal transactions from staking transactions into different touchpoints.

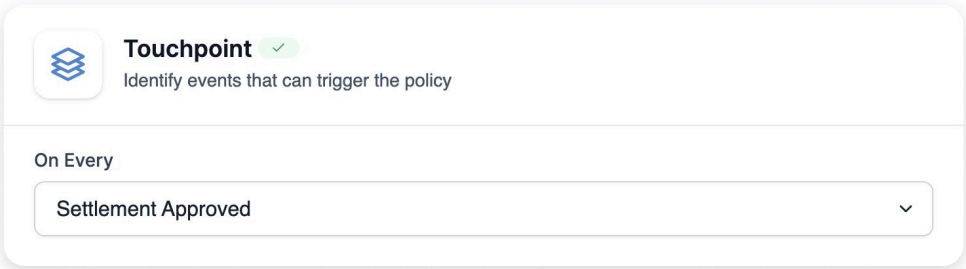


Settlement Initiated

Description: This touchpoint applies to all settlements initiated by a user in the enterprise containing this policy.

Settlement Approved

Description: This touchpoint applies to all settlements initiated by another enterprise or counterparty and require your approval



Conditions

Spending Limit

Description: Verifies the transaction against a specified threshold on a per-transaction basis.

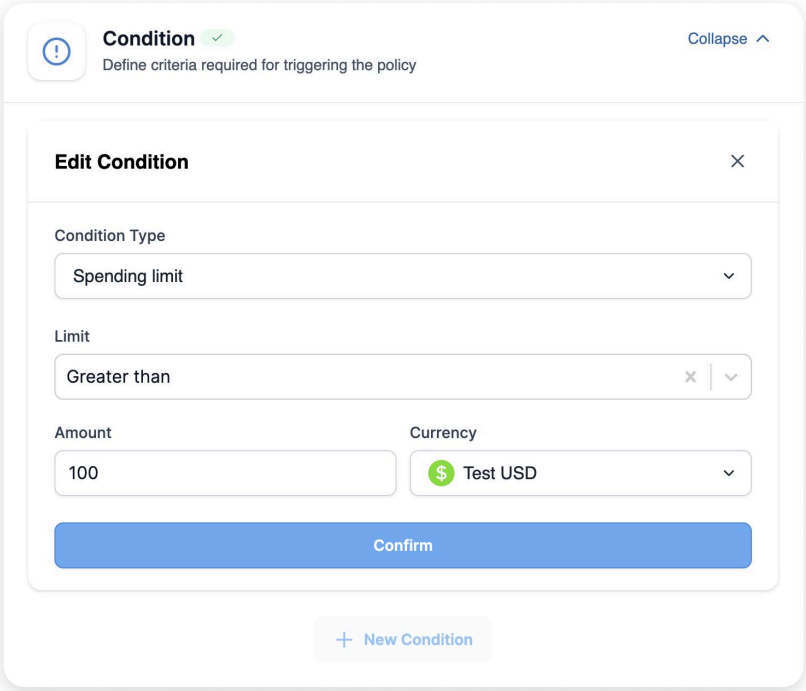
Note:

Limit can be any of the following:

- Greater than
- Greater than or equal to
- Less than
- Less than or equal to

Available assets depends on the scope of the policy

- All Wallets = can only specify in USD
- Wallets by type = can only specify in USD
- Single wallet = can specify in any asset that is supported by that wallet



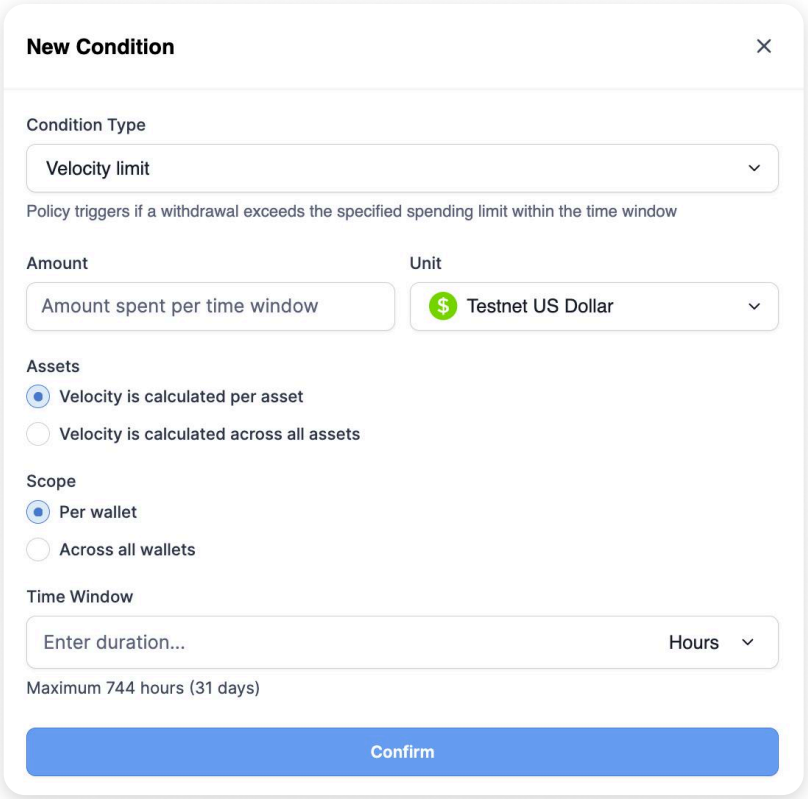
Velocity Limit
Description: Verifies how much can be withdrawn from a wallet or number of wallets within a specified period of time.

Example: Require approval for all withdrawals exceeding the velocity limit of $100 within 24 hours.

Notes:

- The calculation is done on a rolling basis from the point of transaction initiation.
- Pending transfers are included in the calculation. Failed, rejected, and confirmed transactions aren't included in the calculation.
- If the calculation can't be performed for any reason, the associated actions of the policy are required, ensuring that BitGo takes the most cautious approach.

- Toggle for per asset / across all assets
  - When you set a velocity limit denominated in USD, BitGo needs to know which assets to include in the spending calculation.
  - Per asset = only include transactions of the transacted asset in the spend calculation (e.g., on LINK tx's, only consider other LINK tx's)
  - Across all assets = include transactions of any asset in the spend calculation
- Toggle for per wallet / across all wallets (e.g., on LINK tx's, consider LINK and other assets like ETH, USDC, etc.)
  - This is only available if the scope includes more than 1 wallet - 'All Wallets', 'Wallets by type', 'Multiple wallets'
  - Per wallet = calculates the spending for that specific wallet
  - Across all wallets = calculates the spending across all wallets in the enterprise
- Available assets depends on the scope of the policy
  - All Wallets = can only specify in USD
  - Wallets by type = can only specify in USD
  - Single wallet = can specify in any asset that is supported by that wallet



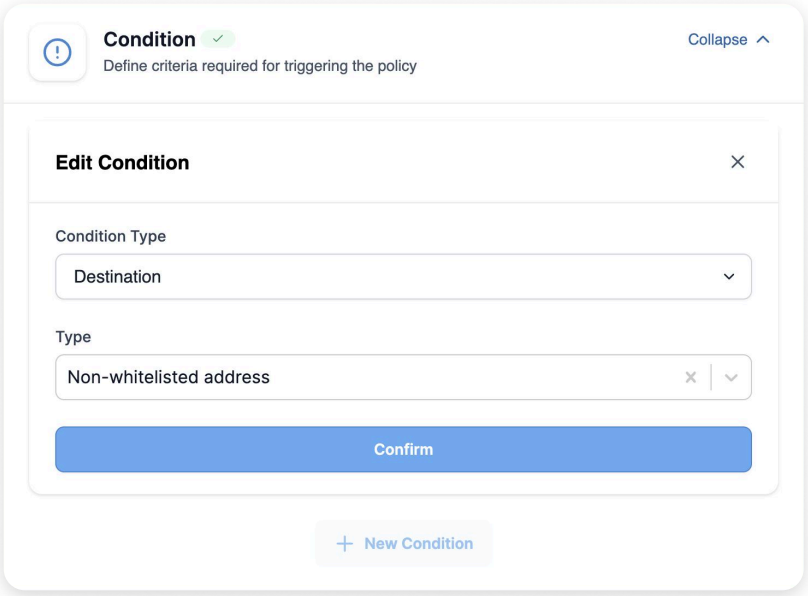
Destination

Description: Verifies whether transactions are to a whitelisted or non-whitelisted addresses.

Notes:

- It is most common to manage the whitelist at the individual wallet level.
- If an enterprise whitelist is in use for the asset being transferred, the address must be whitelisted on both the enterprise level and wallet level.
- The Address Book is not related to the whitelisted addresses
- Every time a new whitelist is created for a self-custody wallet, a corresponding rule is created that dictates the behavior. For example:
  - Scope: [Newly created wallet]
  - Touchpoint: Withdrawals
  - Condition: To a non-whitelisted address
  - Action: requires approval
- In order to change the behavior (e.g., change the associated action to ”deny”), you have 2 options:
  - Unlock the policy that was automatically created and change the action to “requires approval”.
  - Unlock the policy to archive, and then create a more global policy:

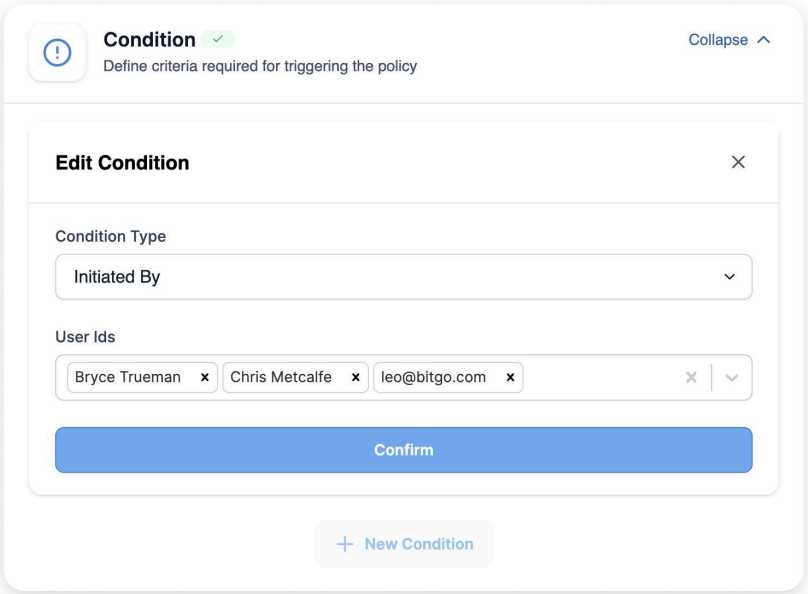
    → Scope: All self-custody hot wallets
    → Touchpoint: Withdrawals
    → Condition: To non-whitelisted address
    → Action: Deny
- The reason the automatically created rule needs to be deleted is because any time aggregated actions include a ”deny”, the result is a denial.



Initiated by

Description: Verifies if the withdrawal was initiated by a specified user.

Example: Require 1 wallet admin approval if a withdrawal is initiated by Bryce, Chris, or Leo.



Webhook

Description: On every withdrawal, BitGo sends a webhook with a payload to a specified URL. Based on the response, the associated policy actions are either required or not required:

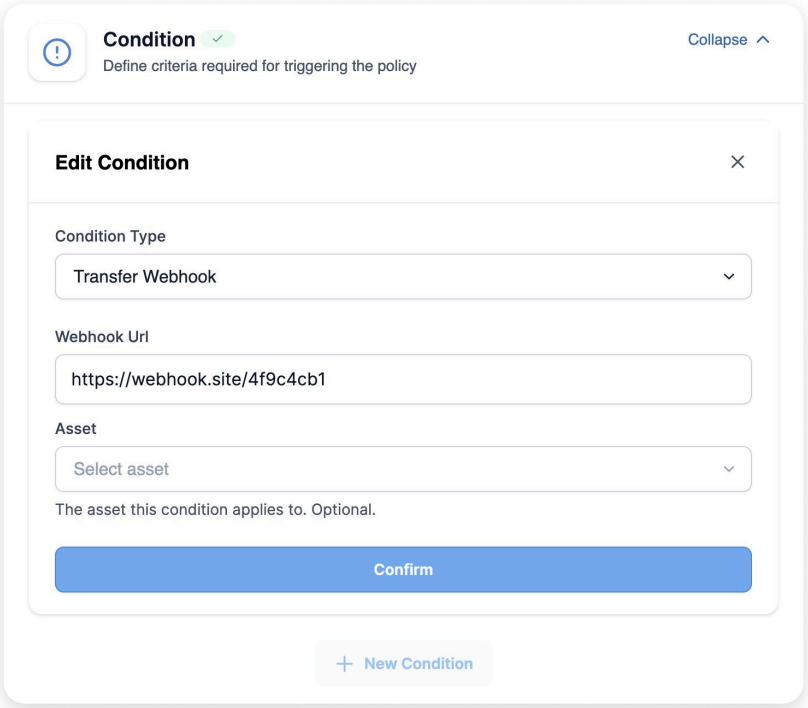
- 200 response = no actions required
- Non-200 response = actions required

This type of condition is useful if there is additional verification that you want to build on your side outside of the Policy Engine. For example, platforms who want to perform a second level of verification on withdrawals to unknown addresses. There's also an optional asset condition so that only withdrawals of that asset will send the webhook.

Example: Require 1 wallet admin approval if a non-200 response is received from the receiving webhook.

Example payload

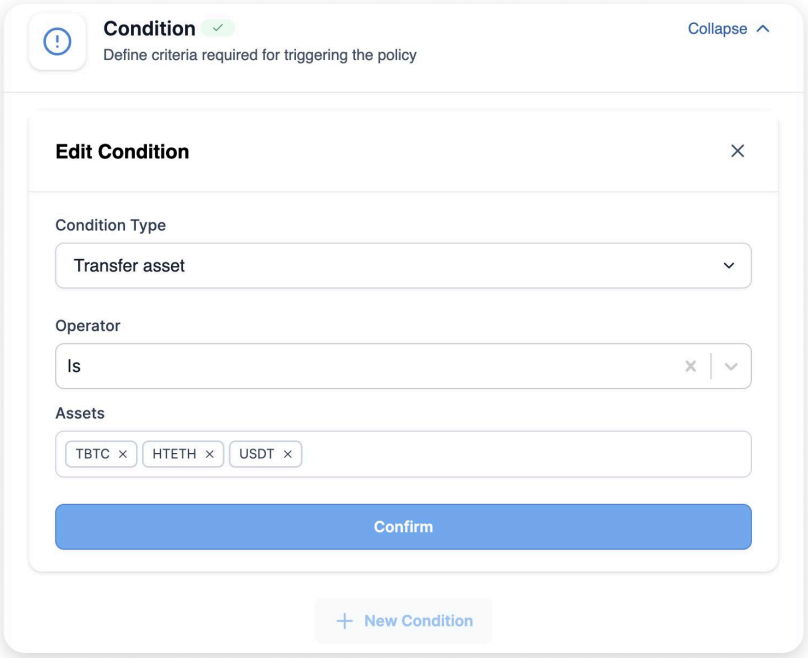
```
JavaScript
{
  "idempotencyKey": "661ee7c74ae97f7c401fe592f9595123",
  "walletId": "65417326b197d7000711bdbc8645b625",
  "ruleId": "f2ee77f5-0447-55ac-b73f-175aa71b9880",
  "type": "webhook",
  "spendAmount": "120000000000000000000000",
  "usdCentAmount": "64.092",
  "sequenceId": null,
  "approvalCount": 0,
  "halfSigned": {},
  "outputs": [
    {
      "address": "76259bef04815e3f0a2b68c73f2c5a6a4b9116b8b68b5529626c6ccea18a2f1b",
      "value": "1.2e+23",
      "wallet": "64ff7bc240a60c00070f2c1a992ff22c"
    }
  ]
}
```



Asset

Description: Policy triggers if withdrawing a specified asset. This enables you to build policies that target specific assets, which could be across many wallets or within a single wallet with tokens.

Example: Require 1 wallet admin approval for all BTC, ETH, or USDT withdrawals.

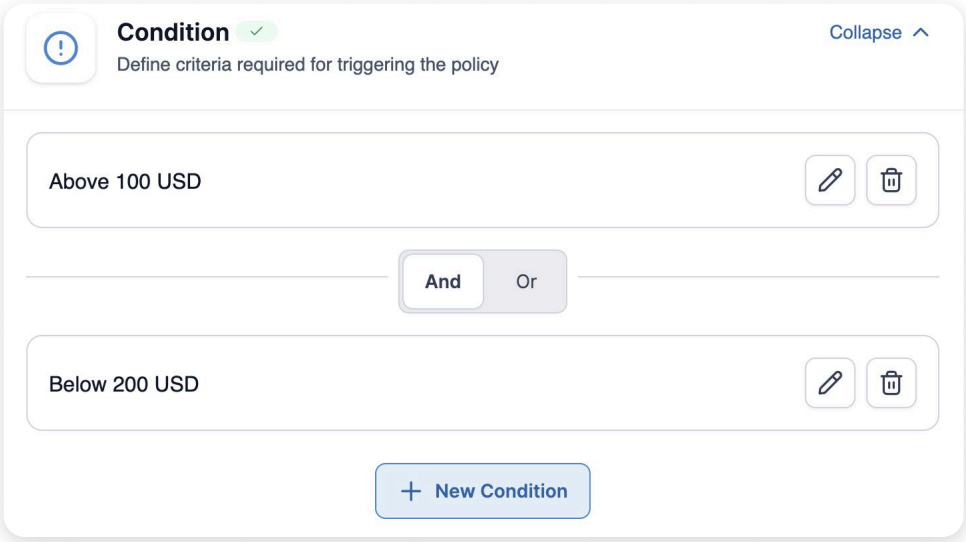


Combining Conditions

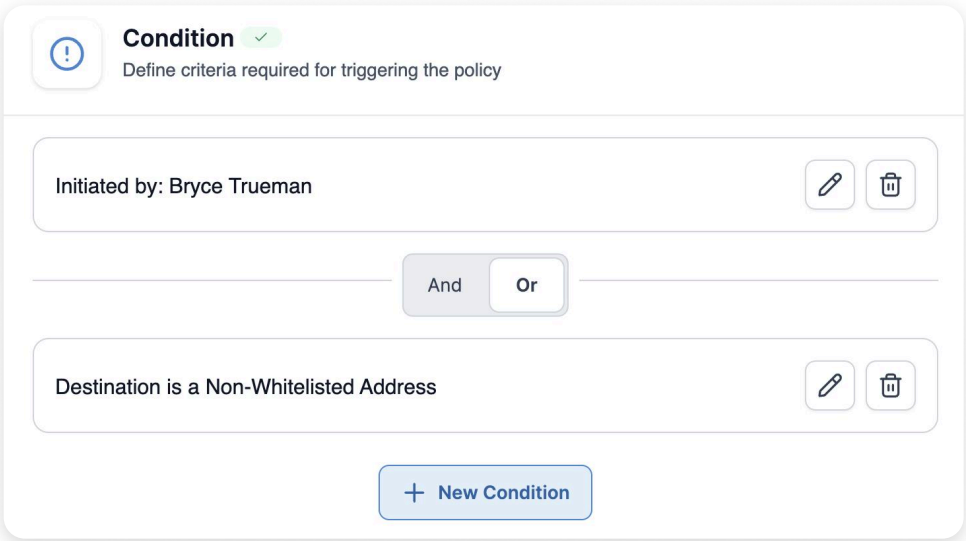
You can combine as many conditions as you like in a single policy. Once you've added more than one condition, you can choose whether you want:

1. At least one of the conditions to evaluate to true (conditions combined with OR)
2. All of the conditions evaluate to true (conditions combined with AND)

This policy would only trigger if the withdrawal is both above $100 and below $200.



This policy would trigger if the withdrawal is either initiated by Bryce or is going to a non-whitelisted address.



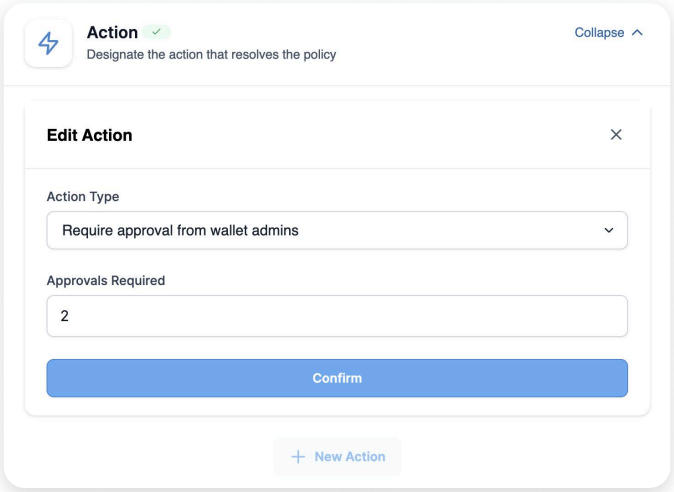
Actions

Require approval from wallet admin

Description: This action requires approvals from the admins of the initiating wallet at the time of withdrawal.

Example: Require 2 wallet admin approvals for all withdrawals greater than $10,000.

Note: If there aren't enough wallet admins to satisfy the rule, the withdrawal is automatically rejected. Additionally, a wallet admin can't approve their own withdrawal. Therefore, if a wallet admin plans to initiate the withdrawal, there needs to be n+1 wallet admins, where n = the number of admin approvals required.

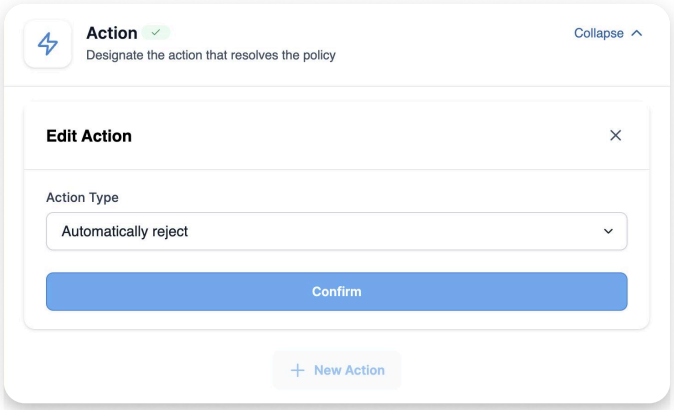


Require final approval

Description: This action requires 1 final approval from a list of users that hold a role on the wallet (admin, spender, or viewer). The approval occurs after the wallet admin approval and the approval from a group of users. The sequential nature of this action is useful for ensuring certain approvals take place before others. For example, only notifying and requiring approval from a high level executive if all other approvals have taken place

Example: Require final approval from Bryce, QA User, or Enterprise Owner 1 for all withdrawals greater than $10,000.

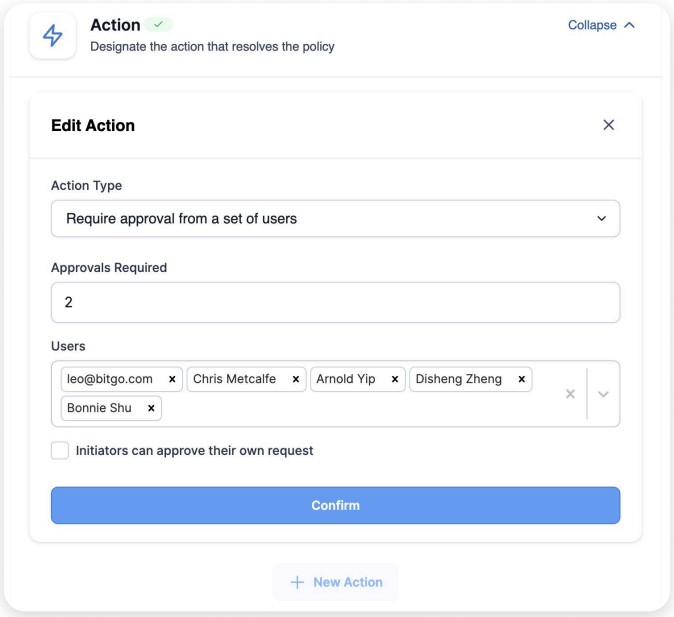
Note: Only available for the scope of a single wallet, because the final approver needs to be a member on the initiating wallet. The default behavior is that the transaction initiator can provide their own final approval, but you can toggle this on or off.



Require approval from a set of users

Description: New action type that enables you to request approvals from any users in the enterprise, regardless of whether or not they are a user on a specific wallet. This can enhance your overall security by segregating duties among team members. You must have the View All Wallets option enabled from the enterprise settings. Optionally, specify whether the transaction initiator can provide an approval for their own transaction.

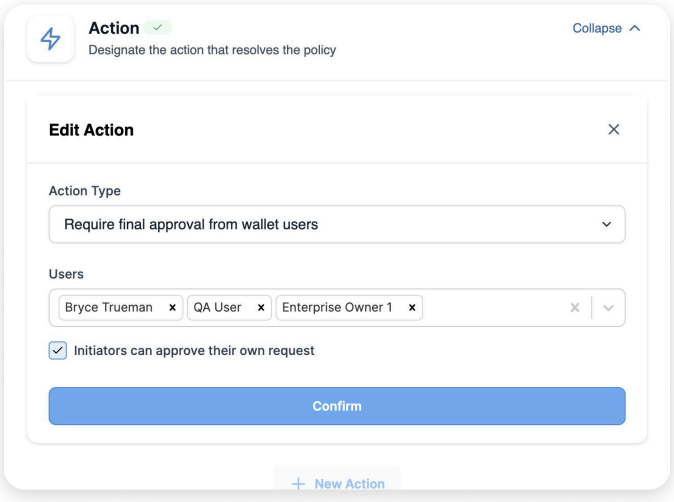
Example: Require 2 approvals from Leo, Chris, Arnold, Disheng, or Bonnie for all withdrawals from any wallet.



Automatically Reject

Description: If the withdrawal triggers a policy with this action and the conditions evaluate to true, the withdrawal is automatically rejected. This is useful when you don't want to allow exceptions.

Example: Automatically reject withdrawals to non-whitelisted addresses. This means no exceptions - all withdrawals must go to a whitelisted address.



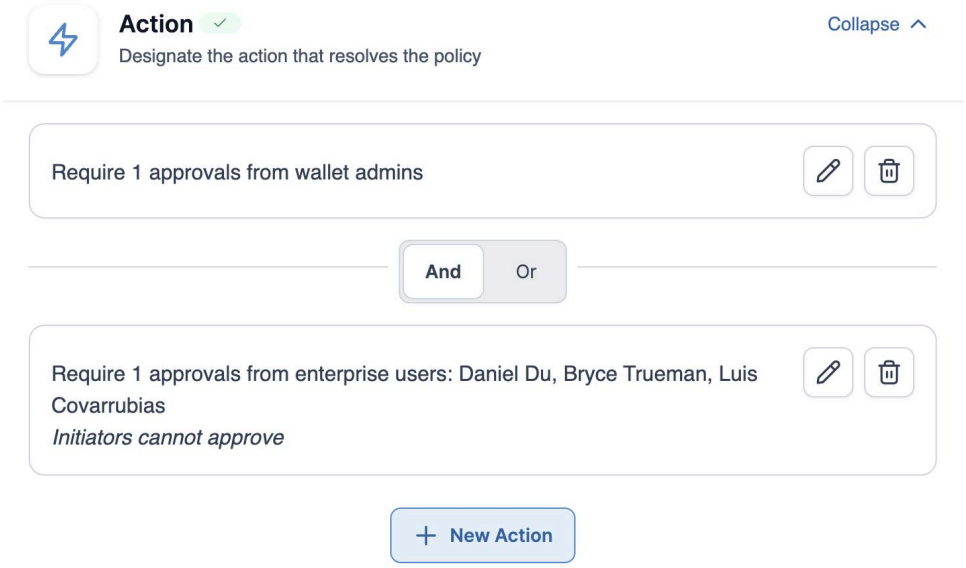
Combining Actions

Similar to combining conditions, you can also combine multiple actions and require that either:

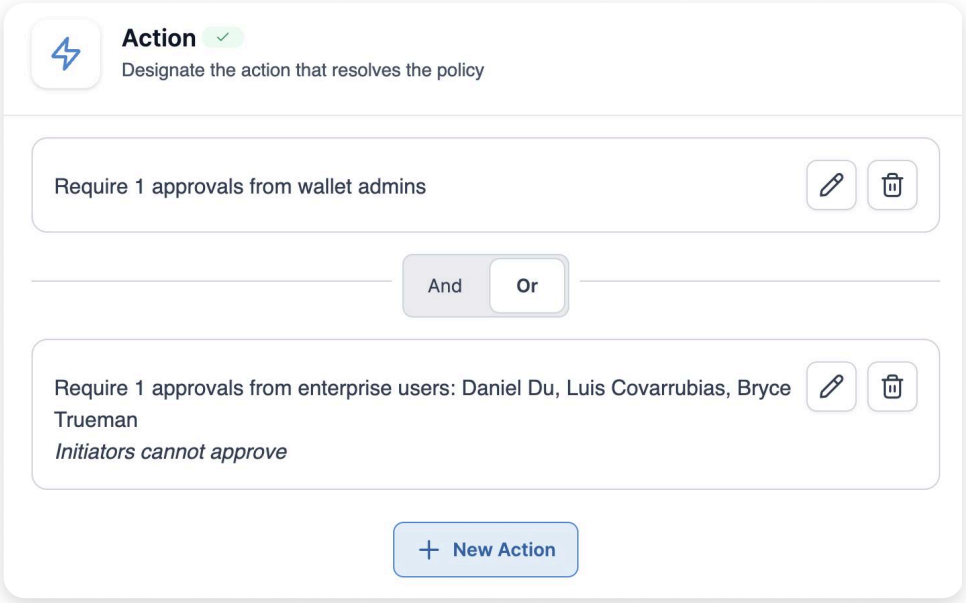
1. Only one of the actions must be completed
2. All of the actions must be completed

Note: the automatically reject action trumps all other actions

To resolve this policy, a wallet admin needs to approve **and** 1 of the enterprise users needs to approve

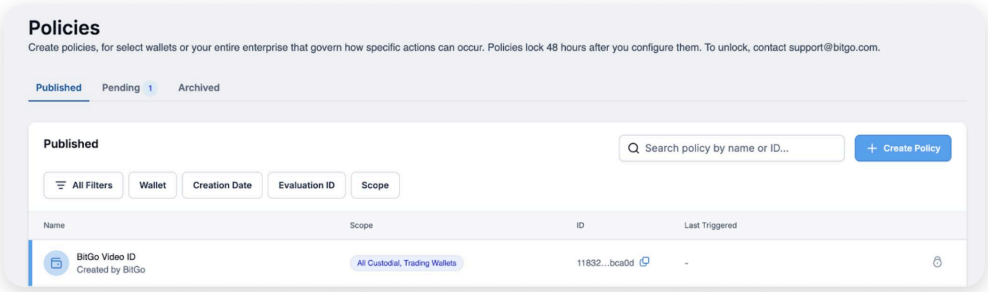


To resolve this policy, **either** a wallet admin needs to approve **or** 1 of the specified users needs to approve.



BitGo-managed policies

BitGo-managed policies are implemented on behalf of our clients to ensure the highest level of security. These policies establish robust safety measures to safeguard clients' digital assets and, in some cases, are necessary to uphold our status as a regulated custodian and the insurance coverage on assets. Because of this, not all policies are customizable or archivable by clients. Enterprise Admins do however have the ability to view or edit certain policies. By leveraging BitGo-Managed Policies, your assets remain secure with our industry-leading security protocols. As our platform evolves, we plan to provide greater flexibility in customizing certain BitGo-Managed policies to balance security needs with user control.



BitGo Video ID

This policy dictates what transactions require an eligible Video ID user to perform a Video ID verification call with BitGo Support to confirm their identity and details of the transaction. By default, a Video ID verification call is required for all transactions that exceed $250,000 in a rolling 24 hours.

Should you wish to customize this policy, the following options are available:

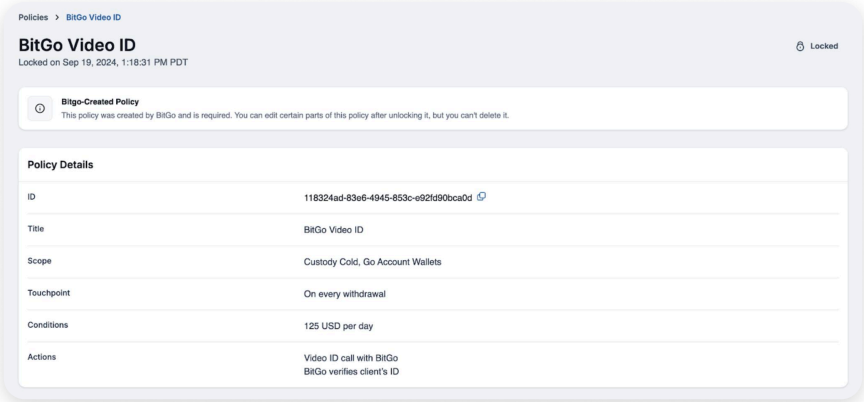
1. Modify the velocity limit to require video ID verification for a designated value between $0 and $5,000,000
2. Opt into BitGo's risk rating model - a new enhanced mechanism that looks at several factors to determine the level of risk on every transaction, requiring video ID for all transactions deemed high-risk
3. Opt in to Video ID for every transaction

To change this policy, follow the same steps required to modify an existing policy:

- Unlock the policy
- Modify the policy
- Submit the policy for approval, if there is at least one other enterprise admin

Please note: you cannot perform the following actions on this policy:

1. Archive to opt out of all video verification IDs
2. Change the name, scope, touchpoint, or actions of this policy



Note on the two Actions:

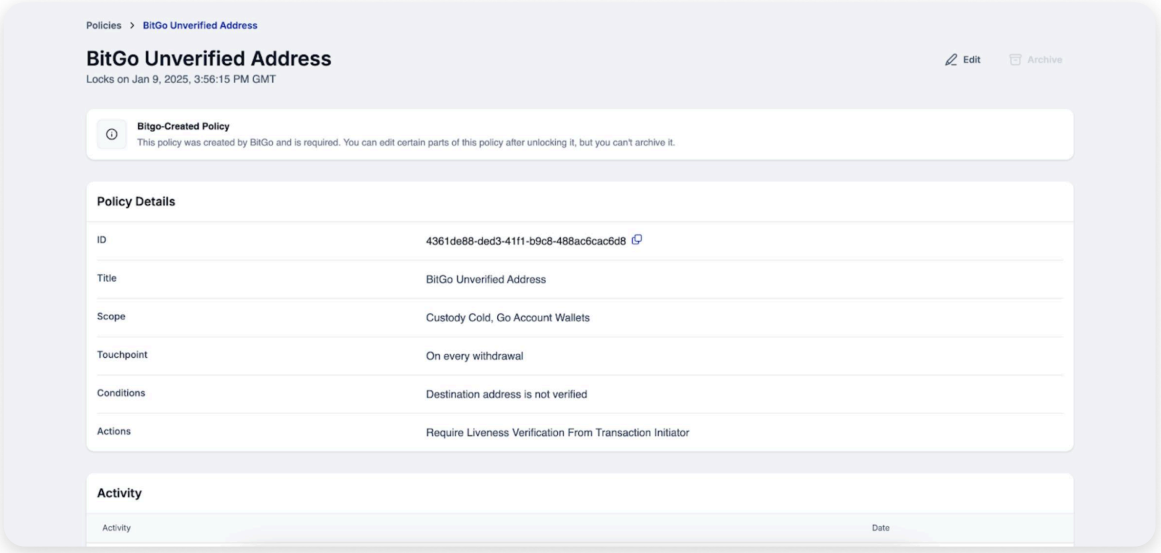
1. Video ID call with BitGo - this indicates the client performed a video ID verification call with BitGo Support to confirm their ID and the transaction
2. BitGo verifies client's ID - this indicates no client action required and is a step performed by BitGo Trust before signing the transaction

BitGo Unverified Address

This policy dictates what identity verification is required when sending to an unverified address.

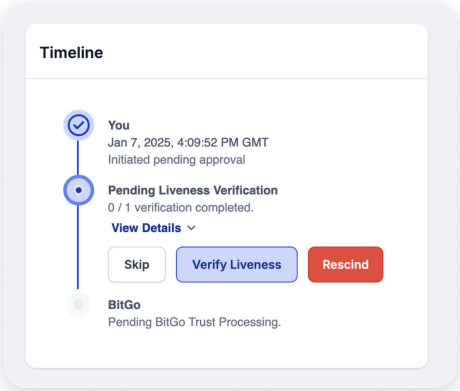
- Unverified Address: an address that has been added to a wallet whitelist , but you have never made any withdrawals to it. When withdrawing to that address for the first time, there is a requirement for additional verification.
- Verified Address: Once you've completed a verification and withdrawal for an address, all future withdrawals will not require additional verification. Note: a withdrawal may still require a Video ID if the trust velocity limit is exceeded.

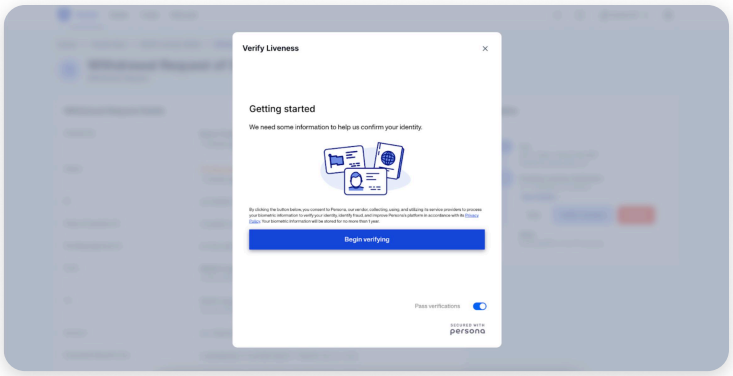
The way that you verify a new address is by completing a liveness check from the user that initiated the transaction. Using the policy engine, you can change who is required to complete the selfie (wallet admin, video ID user), or you can opt in to a full Video ID verification call with BitGo, if you prefer.

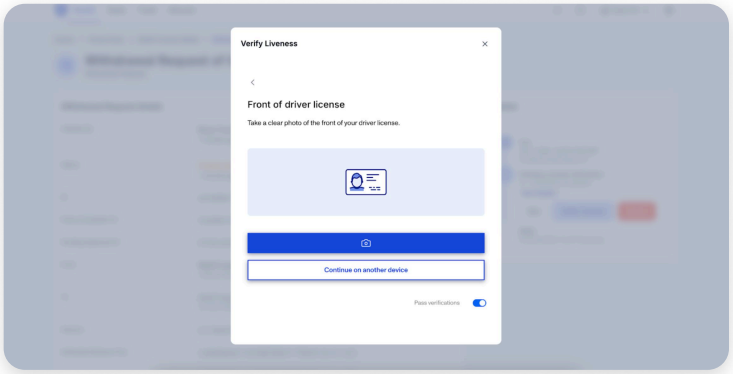


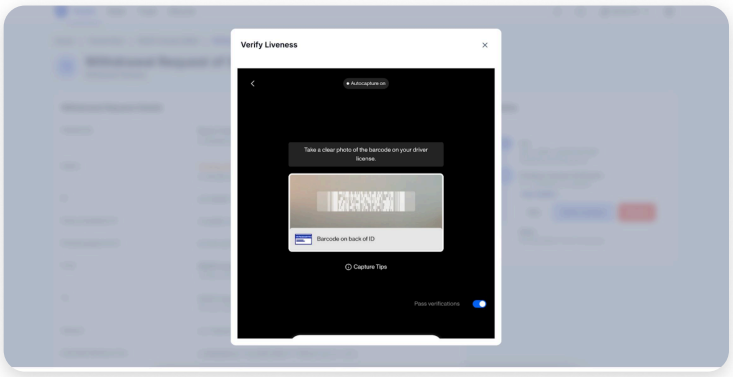
How to complete a liveness check

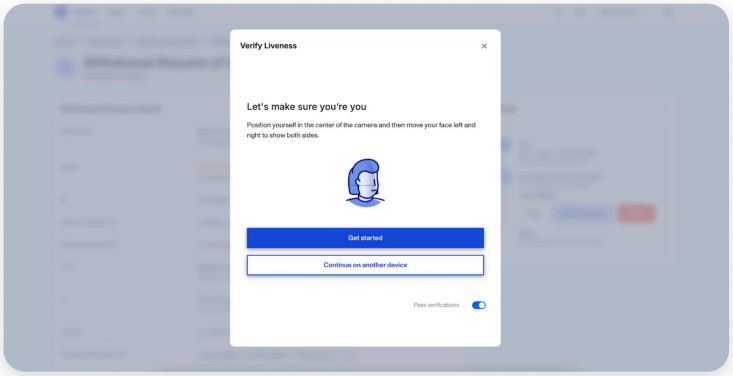
1. Add a new address to the whitelist
2. If there is another admin on the wallet, they will need to approve the new address
3. As an Admin or Spender, when you initiate a transaction to the new address for the first time, you will be prompted to complete a selfie liveness check
4. This consists of uploading a picture of a government issued ID and taking a selfie
5. If the verification succeeds, the withdrawal will then continue to be processed (all other policies will still be evaluated as normal)
6. If the verification fails, or if you chose to skip the selfie, you will be prompted to complete a Video ID verification call with BitGo.

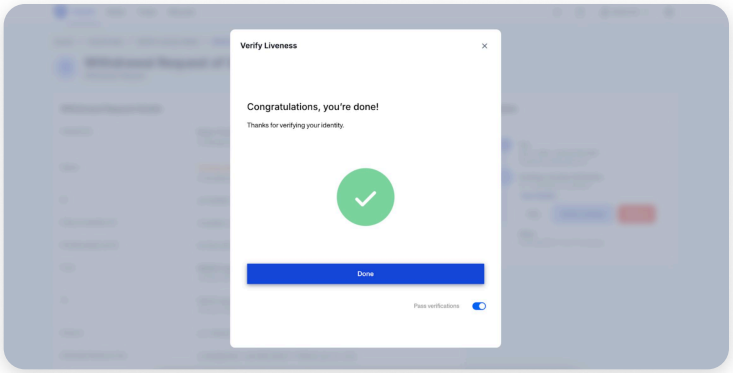

Timeline
You
Jan 7, 2025, 4:09:52 PM GMT
Initiated pending approval
Pending Liveness Verification
0 / 1 verification completed.
View Details
Skip
Verify Liveness
Rescind
BitGo
Pending BitGo Trust Processing.


Verify Liveness
Getting started
We need some information to help us confirm your identity.
Begin verifying
Pass verifications
persona

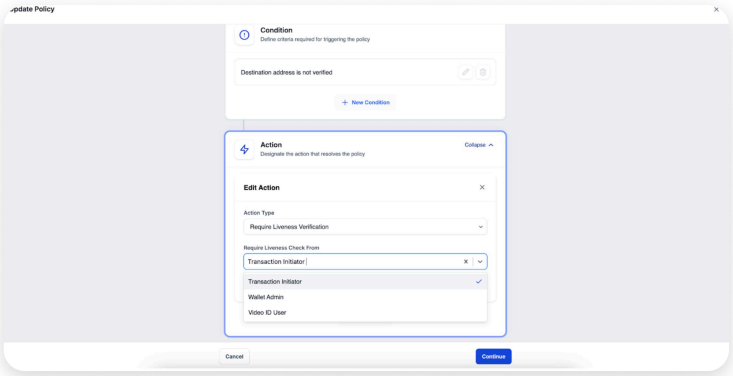

Verify Liveness
Front of driver license
Take a clear photo of the front of your driver license.
Continue on another device
Pass verifications


Verify Liveness
Autocapture on
Take a clear photo of the barcode on your driver license.
Barcode on back of ID
Capture Tips
Pass verifications





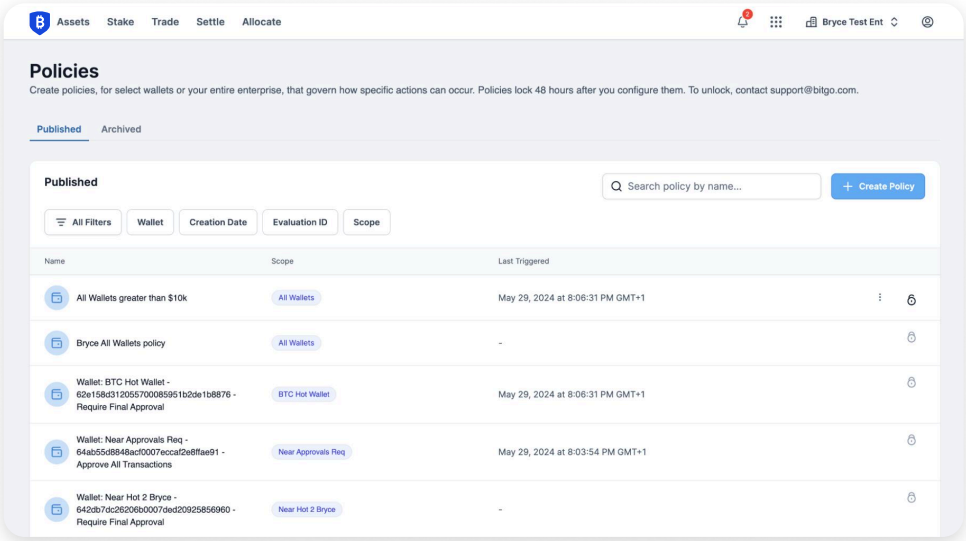
If you want to change the requirement from a selfie liveness check to a video ID, you can do so by unlocking the 'BitGo Unverified Address' policy and changing the Action.



Managing Policies

Where to Access?

You can access all policy-management operations from the Policies tab at the top of the page, in the nine-dot menu option. You no longer need to navigate to each wallet to set up your policies.



Permissions

Create

- Enterprise owners with approval from another enterprise owner (if applicable) can create policies for all wallets in an enterprise
- Wallet admins can create policies only for the wallets they are an admin of

View

- Enterprise owners can view all policies with scope of All Wallets or Wallets by Type
- Wallet admins can view the policies that affect their wallets

Edit

- Enterprise owners with approval from another enterprise owner (if applicable) after first unlocking the policy through video ID verification
- Wallet admins can unlock and modify policies for the wallets they're admins of, but will require other wallet admins' approval (based on the required # of approvals setting)

Archive

- Enterprise owners with approval from another enterprise owner (if applicable) after first unlocking the policy through video ID verification
- Wallet admins can unlock and modify policies for the wallets they're on, but require other wallet admin approval (based on the required # of approvals setting)

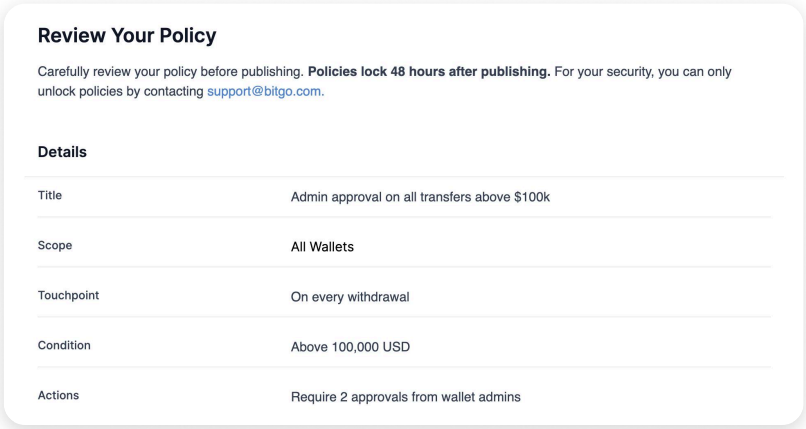
Creating a Policy

1. **Name -** custom recognizable name for the policy
2. **Scope -** should the policy cover all, some, or just 1 of your wallets?
3. **Touchpoint -** what specific user operation are you trying to regulate?
4. **Condition -** what conditions must be true in order to require the actions

Example Policies

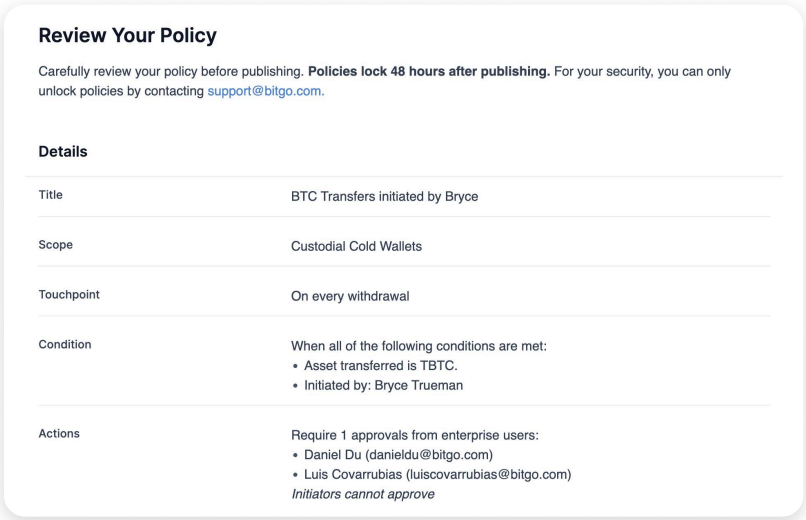
For all transfers greater than $100k, require 2 admin approvals.

- Scope - all wallets
- Touchpoint - withdrawal
- Conditions - spending limit above $100k
- Action: require 2 admin approvals



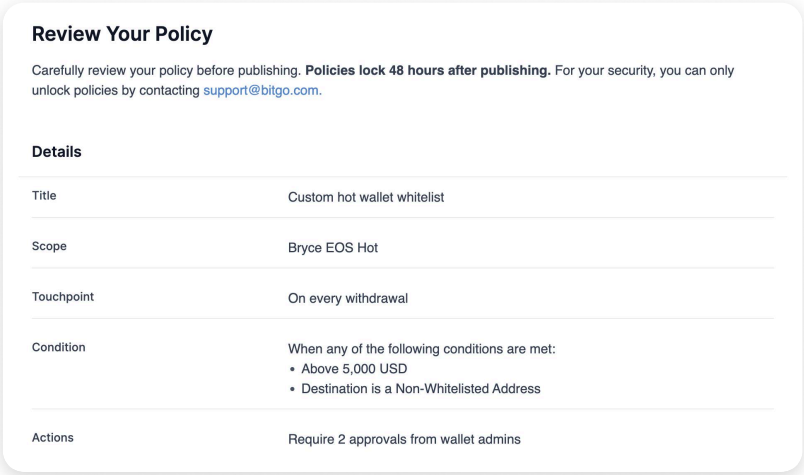
For all BTC transfers initiated by UserX, require approval from UserY or UserZ.

- Scope - All Custodial wallets
- Touchpoint - Withdrawal
- Conditions - Asset = BTC **and** Initiated by Bryce
- Actions: require approval from Daniel or Luis



For all transfers greater than $5k and to a non-whitelisted address, require 2 wallet admin approvals

- Scope: Bryce EOS Hot
- Touchpoint - Withdrawal
- Condition - Spending limit greater than $5k **or** Destination non-whitelisted address
- Action: require 2 wallet admin approvals



Review Your Policy

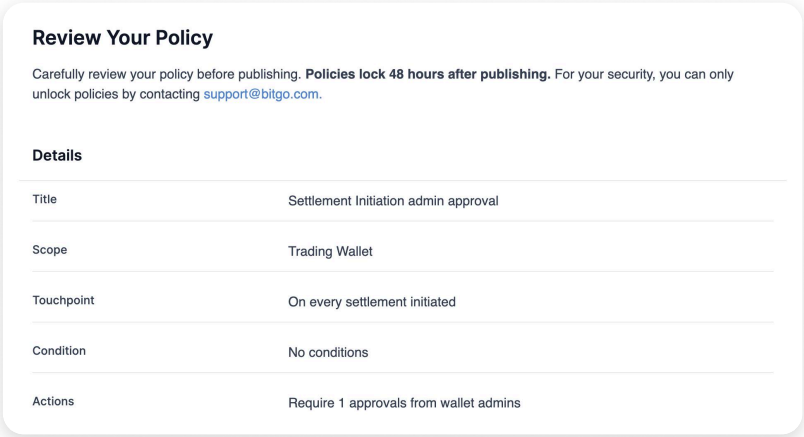
Carefully review your policy before publishing. **Policies lock 48 hours after publishing.** For your security, you can only unlock policies by contacting support@bitgo.com.

Details

Title	Custom hot wallet whitelist
Scope	Bryce EOS Hot
Touchpoint	On every withdrawal
Condition	When any of the following conditions are met: • Above 5,000 USD • Destination is a Non-Whitelisted Address
Actions	Require 2 approvals from wallet admins

For all settlement initiations, require 1 wallet admin approval.

- Scope: Go Account
- Touchpoint: Settlement Initiation
- Conditions: N/A
- Action: require approval from Wallet Admins, Approvals required: 1



Review Your Policy

Carefully review your policy before publishing. **Policies lock 48 hours after publishing.** For your security, you can only unlock policies by contacting support@bitgo.com.

Details

Title	Settlement Initiation admin approval
Scope	Trading Wallet
Touchpoint	On every settlement initiated
Condition	No conditions
Actions	Require 1 approvals from wallet admins

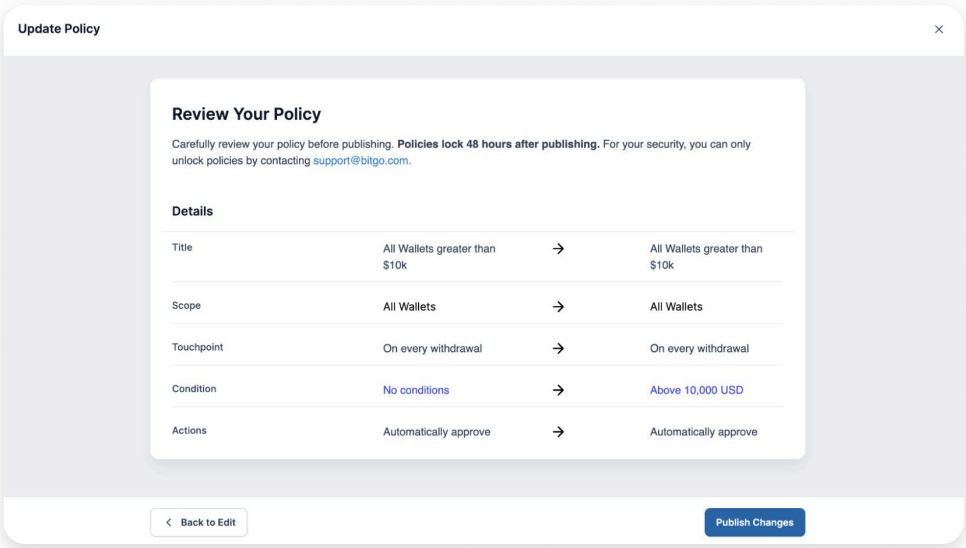
Modify an existing policy

In order to enhance the security of the policies and limit the potential harm of a bad actor - BitGo locks all policies 48 hours after creation. If you need to modify an existing policy that's locked, you must contact BitGo support to request a policy unlock. During request, you must specify for how long you need your policy to remain unlocked (e.g., 48 hours). During this period, you can make any changes to the policy. Once the time period ends, the policy automatically locks again.

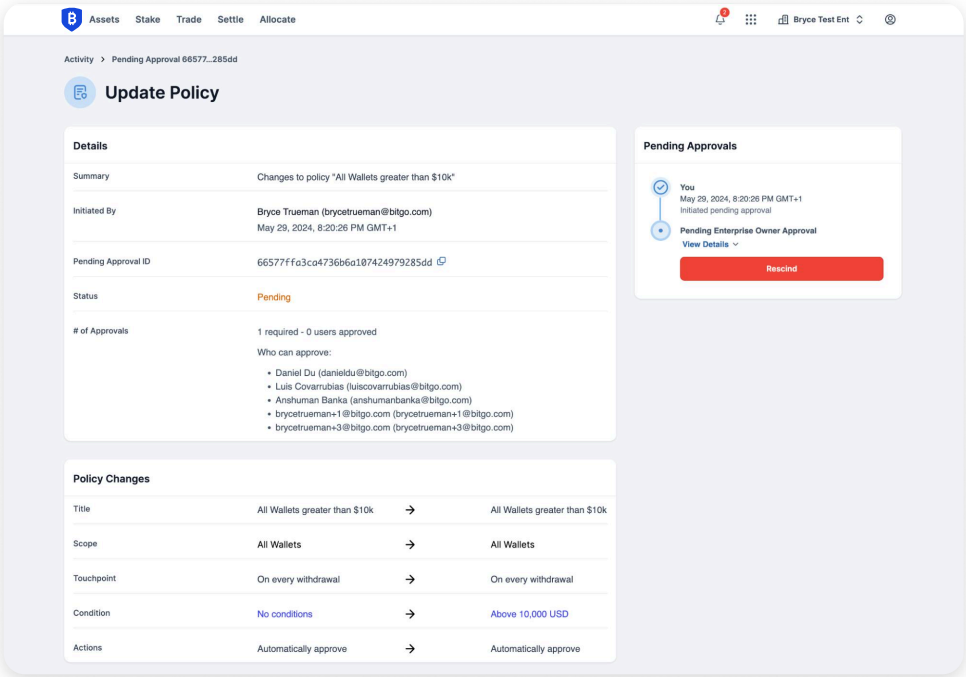
Three important notes about policy locking:

1. Locking and unlocking happens at the individual policy level - not all policies. E.g., you have 10 unique policies, when you request to unlock 1 of them, the other 9 remain locked and uneditable.
2. Once the policy is unlocked, you must make all necessary changes and ensure you submit the changes for approval. The policy can lock again, while the approvals are pending. This is different from prior behavior, where the policy changes needed approvals before the unlock period ended.
3. When a policy is unlocked, any user with the appropriate permissions can modify the policy. Once the changes are submitted, they're locked and no other user, regardless of having their permissions, can attempt to change this policy. I.e., only one in-progress change can occur at a time.

Example: Here I am changing a policy from having no conditions at all, to having a spending limit of greater than $10k USD.



Here is what the approval looks like for the above policy change:



Archive a Policy

If you no longer need a policy - you can archive it. You can do this by: unlocking the policy, opening the policy editor, and archiving the policy. You can access this history in the Archived tab.

Auditing Policies

Audit log

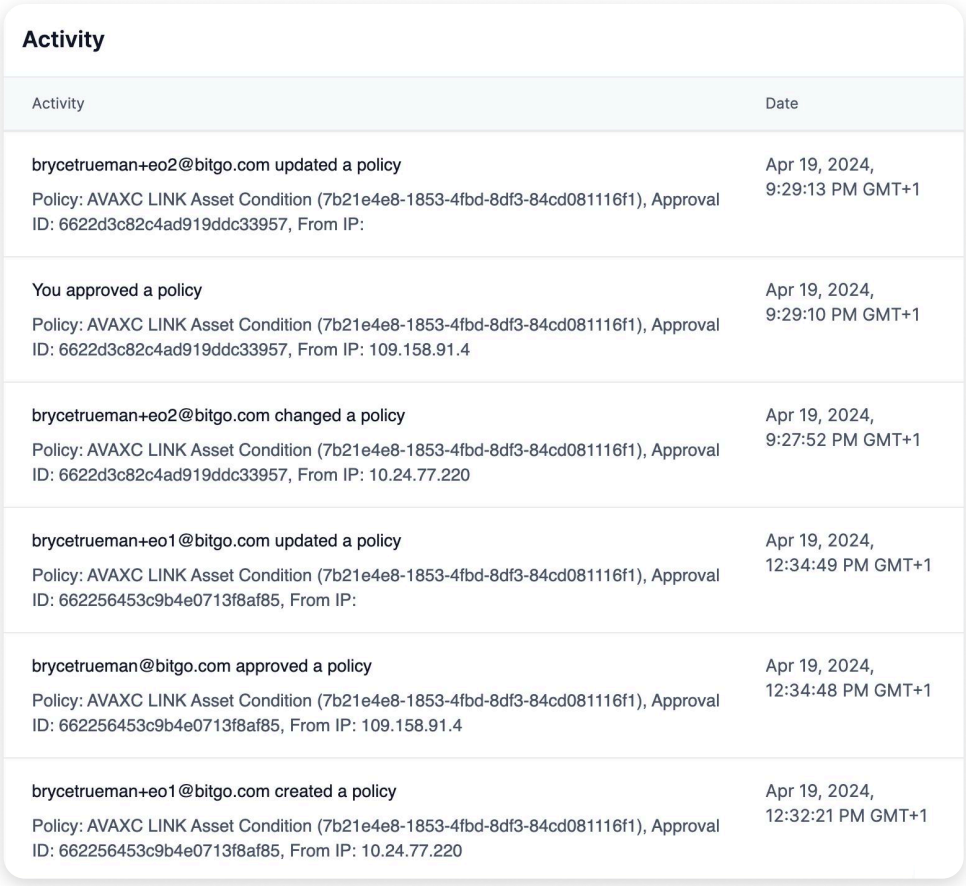
Description: The audit log provides a full history of all actions performed on a policy. For example:

- Creation
- Changes
- Approvals for changes
- Archival

For each event, you can view:

- Who performed the action
- When the action took place
- Name of the policy
- Policy ID
- Associated pending approval ID
- IP address of the account that performed the action

You can view the audit log by visiting the Policies page and clicking on an individual policy:



Search by Evaluation ID

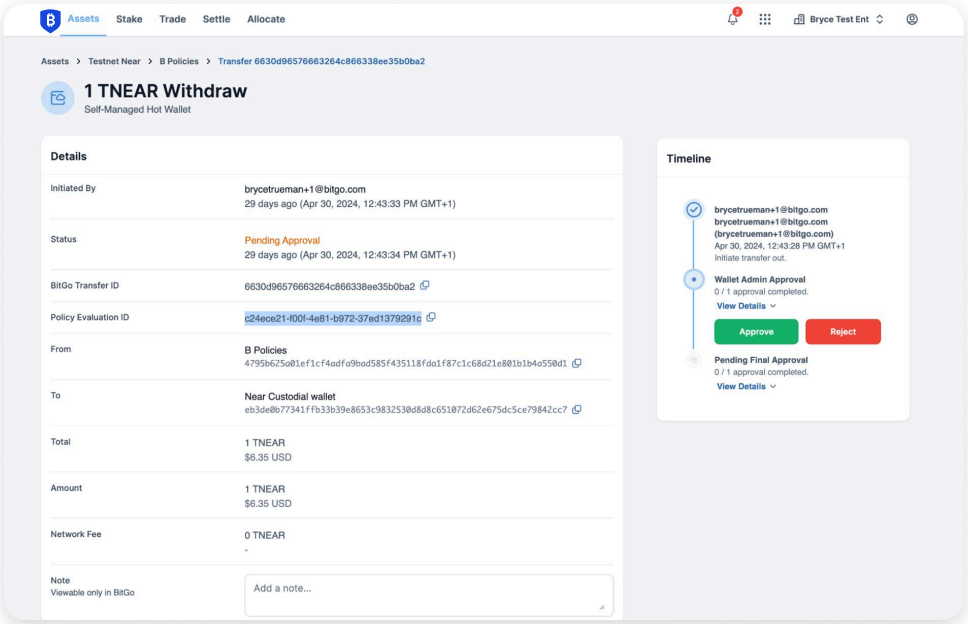
Description: Every transaction has an associated evaluation ID. This ID corresponds to all of the policies that were evaluated when the transaction was initiated. A single withdrawal can trigger multiple policies, depending on how they're configured. Searching by the evaluation ID in the Policies page shows you exactly which policies were involved, and therefore why you're seeing the resulting actions in the timeline.

Note: Some policies are internal to BitGo and are not surfaced to the user. These policies will not show up when searching by Evaluation ID.

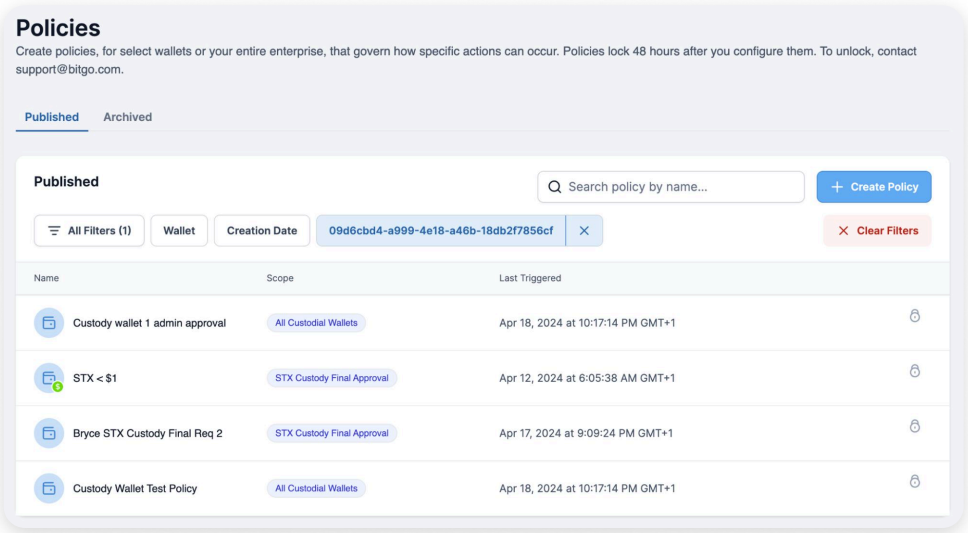
Example: A withdrawal triggered 4 unique policies, because it was a withdrawal from the wallet (STX Custody Final Approval) as well as a Custody wallet, which has its own set of policies.

Note: BitGo may deduplicate some of the actions (e.g., one action is get 1 wallet admin approval, another is get 2 wallet admin approvals, this will result in 2 wallet admin approvals needed).

The Evaluation ID can be copied from the Transfer Details page:



The Evaluation ID can be copied from the Transfer Details page:



Last Triggered

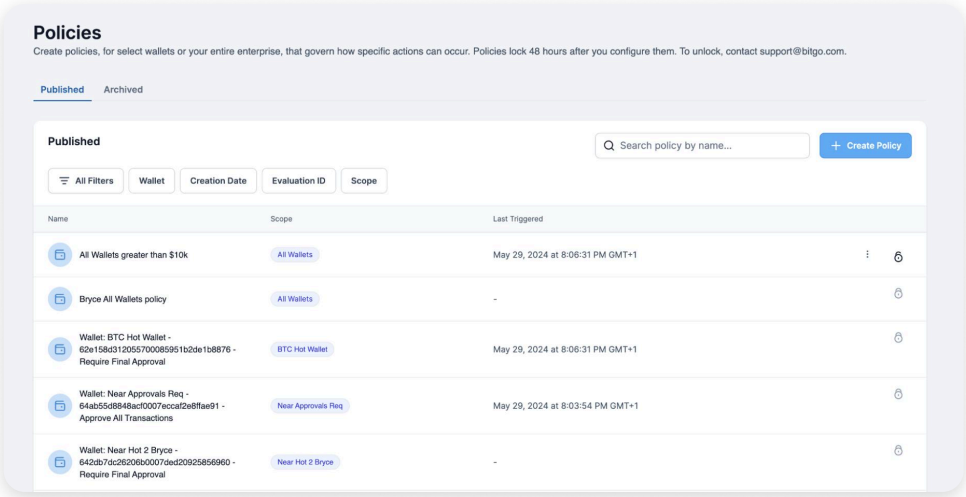
Description: The Last Triggered column in the Policies table tells you the last time that the policy was evaluated and that the conditions of the policy evaluated to true. It is possible that multiple policies are triggered by a single withdrawal. This can give you some insight into how the policies are behaving:

- If the policy has never been triggered - has the policy been set up correctly?
- This policy always has a very recent Last Triggered date - is this policy overly sensitive?
- Etc.

Note: The Last Triggered timestamp resets when a policy has changed.

Example: The screenshot below shows several policies, 3 of which were recently triggered:

- *A single withdrawal triggered both policies: 'All Wallets greater than $10k' and 'Wallet: BTC Hot Wallet - 62e158d312055700085951b2de1b8876 - Require Final Approval'*
- *A different withdrawal triggered the policy 'Wallet: Near Approvals Req - 64ab55d8848acf0007eccaf2e8ffae91 - Approve All Transactions*



Timeline

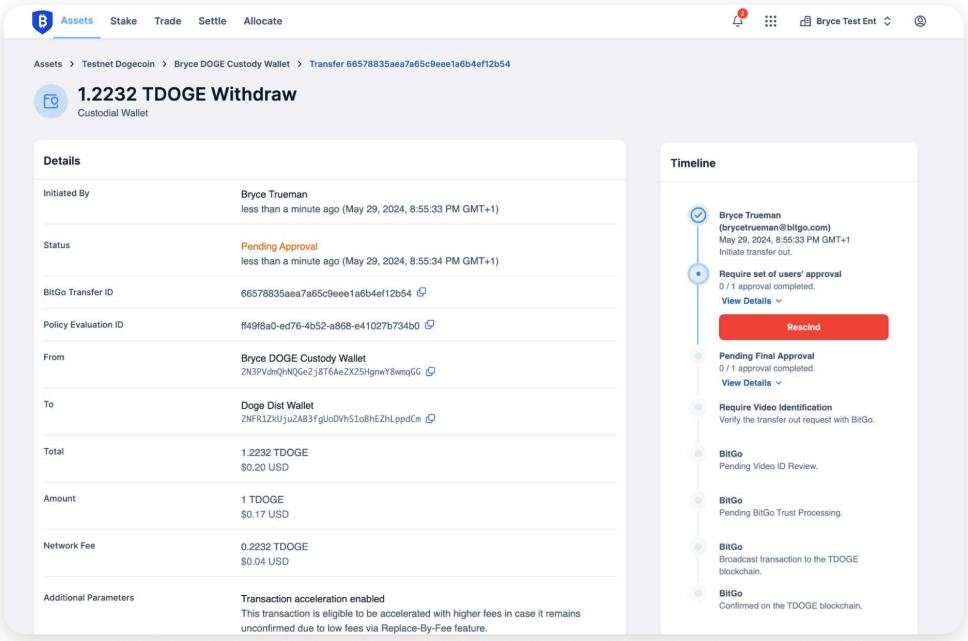
Description: The timeline is the component on the right side of the page that tells you all of the steps that must be completed before the transaction is broadcasted to the blockchain. The timeline can have varying number of steps depending on the asset, the type of wallet, the actions required, etc.

Additionally, the full timeline gives you a sense of:

- Which steps are completed, in progress, or not yet started
- Who completed each step, or who is eligible to complete a present/future step
- Once completed, it will show you a time

Example: Here is a DOGE withdrawal from a Custody wallet requiring:

- 1 approval from a set of users'
- 1 final approval
- All of the BitGo Trust processing steps



Common Mistakes

1. **The number of approvals required for policy changes:**
   - All Wallets - 1 other enterprise owner
   - Wallets by type - 1 other enterprise owner
   - Single wallet - # of wallet admins that's set as required # of approvals in wallet settings
   - Go Account - # of wallet admins that's set as required # of approvals in Go Account settings

2. **What happens when multiple policies trigger?**
   - Aggregate actions and deduplication

3. **Managing addresses on the whitelist remain the same activity**
   - For hot wallets, there's a corresponding rule to dictate the action when sending to non-whitelisted addresses
   - Go to wallet → Whitelist tab (renamed) → Add or remove addresses

4. **Why do distinct lists of users action require enabling the View All Wallets setting**
   - BitGo must ensure anyone listed in the policy has the ability to view the wallet and transactions covered by the policy

5. **V1 wallets**
   - Not supported by the new Policy Engine
   - Even policies with the "all wallets" scope don't include V1 wallets

6. **Creating a policy with the "all wallets" scope requires a specified number of wallet admin approvals**
   - If a wallet doesn't have the required number of admins, withdrawals are automatically rejected until you add enough wallet admins to satisfy the policy.

7. **If for some reason BitGo can't retrieve a USD price for a token, BitGo automatically triggers the policy, assuming the withdrawal exceeds the USD spending limit. Example**
   - For all transfers above $10 from ETH Wallet X, require 2 admin approvals
   - If a client transfers a token from this wallet, and BitGo is unable to retrieve the price systematically, BitGo will require 2 admin approvals, even if the value of the transfer could be less than $10
   - This is done to ensure the policy is never violated.

8. **The Go Account is not covered under the "all wallets" scope and therefore requires its own policies.**

9. **If you use an enterprise whitelist, destination addresses need to also be added to the wallet whitelist.**

Best Practices

1. **Enable the "View All Wallets" option from the enterprise settings**
   - This enables consistent visibility across your wallets
   - This also enables additional functionality such as listing specific users for approval, saving you from needing to add users to wallets just for approving transactions

2. **Review your wallets holistically:**
   - Do you use solely custody wallets?
   - Do you use a mix of custody and hot wallets?
   - Do you use some hot wallets for one purpose and some for another?
   - Etc.

3. **Review at all the users in your enterprise and determine what roles they should have:**
   - Who do you want to be able to initiate transactions?
   - Who do you want to approve small, medium, and large transactions?
   - Who do you want to be able to manage policies?

4. **Define what small, medium, and large transactions look like to you:**
   - For some, a medium transaction could be $1k - $10k, while for others it could be $500k - $1M. This depends completely on you.

5. **Define what you believe are uncommon or risky behaviors:**
   - Is it uncommon for a single transaction to exceed $1m?
   - Is it uncommon to see more than $100k withdraw from a single wallet in a period of 24 hours?
   - Is it uncommon for a specific user in your enterprise to be sending more than $1k per transaction?
   - What does the average day look like in terms of withdrawal volume across your different wallet types?

6. **Develop a strategy around whitelisting:**
   - Are you going to allow withdrawals to non-whitelisted addresses?
   - If yes, do you want to require extra approvals?

7. **Tactical policy design:**
   - Scope: Use standardized policies where possible to ensure consistent handling (e.g., scope of All Wallets or Wallets by Type)
   - Touchpoint: Withdrawal / Settlement
   - Condition: Set up scaling thresholds based on transaction size to ensure appropriate approvals (e.g., between $100k → $500k, do X, between $500k → $1m do Y)
   - Action: Leverage the list of user action to abstract away approvals from strictly wallet users and ensure there's always an approver available
   - Set up failsafe policies (e.g., never allow sending to a non-whitelisted address that is more than $1m)

8. **Periodically, revisit your policies to determine if they're still what you need:**
   - Are there some policies that are too restrictive and add too much friction? (e.g., increase your definition of a medium transaction)
   - Are there some policies that have never been triggered? And if they haven't, is that a good thing they've never triggered?
   - Ideally BitGo can provide some data around your typical transaction behavior as well.

Rules of Policy Creation

Allowed

- ✓ 1 condition → 1:many actions
  a. If coin is BTC → require approval from Jack and Elon

- ✓ Many conditions (evaluated as AND) → 1:many actions
  a. If coin is BTC AND amount is > $100k → require approval from Jack or Elon

- ✓ Many conditions (evaluated as OR) → 1:many actions
  a. If coin is BTC or ETH → require approval from Jack and Elon

- ✓ The order in which you add actions, is the order in which they will be sequenced
  a. Require approval from Jack and Elon (Jack must approve before going to Elon)
  b. Require approval from Elon and Jack (Elon must approve before going to Jack)

- ✓ If you add a policy at the All Wallets level, it will automatically cover all wallets in the enterprise except for the Go Account

- ✓ Hierarchy
  a. Adding an enterprise-level policy affects all wallets underneath, including exchange connections
  b. Adding a wallet-level policy only affects touchpoints relevant to that wallet

Not Allowed

- ✕ You can't combine AND and OR within conditions, i.e., you must choose all AND or all OR
  a. Not allowed: If Coin is BTC AND user is Jack OR amount > $100k
- ✕ You can't combine AND and OR within actions, you must choose one or the other
  a. Not allowed: If [condition], require approval from Jack AND Elon OR Zuck
- ✕ You can't have condition → action, and condition → action
  a. Not allowed: If coin is BTC → do X, and if amount is greater than $5M → do Y

FAQ

1. **Who can create policies for all wallets or wallets by type?**
   - Enterprise admins can create policies of all wallets and wallets by type, but require approval from 1 other enterprise admin
2. **Where do I set how many approvals are required for new policies and policy changes?**
   - Currently, in the wallet settings
3. **What happens if I try to update a policy but I have fewer admins than the required # of approvals?**
   - Automatically rejected. This is what happens today as well.
   - Additionally, there's logic in the wallet settings for required # of approvals, so you can ensure the number you set is n - 1. To reach this edge case you must remove wallet admins from the wallet below that original threshold.
4. **What happens if there's only 1 admin on the wallet?**
   - This admin is able to build any policies they want on the wallet and there will be no approvals required.
5. **What changes to a policy require approval?**
   - Any changes to a policy requires approval. E.g.,
     i. Name change
     ii. Scope change
     iii. Touchpoint change
     iv. Condition change
     v. Action change
6. **Are V1 wallets supported in the new Policy Engine?**
   - No, you continue to manage policies in the V1 wallet policies page
   - Not even All Wallets policies cover V1 wallets

7 What about in-progress policy changes?

- BitGo only allows 1 in-progress change at a time
- Example
    i. userA requests to unlock policyX
    ii. policyX is now unlocked for 48 hours
    iii. userA changes the spending limit of policyX from $50k to $100k
    iv. userA submits and this goes to approval
    v. now while policyX is unlocked userB thinks they can also make a change, so they open the policy but are shown a message along the lines - “this policy has already been edited” and cannot make changes until the existing pending approval is approved/rejected
    vi. If the change userA made is rejected and there is still remaining time before it gets locked, userB can now open policyX and submit changes, which then go to pending approval again

8 Can I build many policies that cover the same wallet?

- Yes, you can create any number of policies that cover a given wallet

9 What happens when a single transaction triggers multiple policies?

- The actions associated with each policy all apply
- BitGo aggregates and deduplicates. For example:
    i. Policy1 Actions: Get 2 approvals from userA, userB, userC, userD
    ii. Policy2 Actions: Get 1 approval from userC, userE, userF
- Both policies need to be fully satisfied in order to let the transaction through.
- Scenarios:
    i. ☑ userA, userB, and userE approve
    ii. ☑ userA, userC approve (userC satisfies ½ of Policy1 and all of Policy2)
    iii. ✖ userC, userE, userF approve (did not satisfy Policy1)

10 What is the order of actions?

- Currently, the only action that takes place after the other actions is the final approval
- Otherwise, all actions are handled at the same time - users are notified at the same time, any action can be resolved at the same time, etc.

11 What happens to my whitelist?

- Whitelists continue to exist in each wallet and at the enterprise level. However, you can customize the behavior of what happens when withdrawing to non-whitelisted addresses. For example:
    i. Scope: All Wallets
    ii. Touchpoint: withdrawal
    iii. Condition: destination - non-whitelisted address
    iv. Action: Deny (or require approval.)

12 Is staking covered by withdrawal policies?

- Yes, we currently treat staking transactions the same as withdrawals, so they can trigger policies

13 What happens if the pending approval for a policy change is not approved before the unlock period ends?

- The pending approval will continue to exist and approvers can still approve or reject the policy change. If approved, the policy update takes effect.
- This is different from today where a user must unlock the policy again just to approve the changes.

14 What is an evaluation ID?

- An evaluation ID corresponds to a specific evaluation that took place.
- Each evaluation ID maps to any number of policies that were triggered
- This helps both the approver and transaction initiator find which policies were triggered

15 Where can I find an audit log of policy change history?

- In the Policy Details page by clicking on a policy in the Policies page
- You can view these event types:
    i. Policy creation
    ii. Approvals for policy changes
    iii. Policy unlocked including lock time
    iv. Policy changes
    v. Policy action approved/ rejected
    vi. Policy evaluations (maybe)

16 Is the API for the new Policy Engine available to me?

- Yes