

# IBM Resilient



## Security Orchestration, Automation and Response Platform

Carbon Black Protection Integration V1.0.2

Release Date: September 2019

Resilient functions simplify development of integrations by wrapping each activity into an individual workflow component. These components can be easily installed, then used and combined in Resilient workflows. The Resilient platform sends data to the function component that performs an activity then returns the results to the workflow. The results can be acted upon by scripts, rules, and workflow decision points to dynamically orchestrate the security incident response activities.

This guide describes the Carbon Black Protection Integration.

## Overview

This integration consists of 12 functions which call various APIs to perform different actions, such as retrieving approval request details, updating approval requests and deleting files. It also contains a polling component to create incidents in the Resilient platform that correspond to approval requests in Carbon Black Protection.

## Installation

You download the function package to a Resilient integration server, and from there you deploy the functions and components to a Resilient platform. These procedures are provided in the [Resilient Integration Server Guide \(PDF\)](#).

The functions included this package have the following requirements, which are above and beyond those listed in the *Resilient Integration Server Guide*.

- Resilient platform is version 30 or later.
- Carbon Black Protection v8.1 or later.

After installing the package, Resilient Circuits creates a new section, [fn\_cb\_protection], in the app.config file. You need to edit the following settings in that section.

```
[fn_cb_protection]
# Name or IP address of your CbProtect server
server=10.200.1.1

# Access token issued by the CbProtect administrator
token= XXXX-XXXX-XXXX-XXXX

# If your CbProtect server has a self-signed TLS certificate, you cannot verify
it:
# verify_cert=false

# Interval (seconds) for automatic escalation of approval requests, set 0 to
disable
# Suggest 300 as a starting point, which will check CbProtect every 5 minutes
escalation_interval=0

# Optional: query for which requests to escalate; default is to escalate all
open approval requests
# escalation_query=resolution:0

# Optional: path to a custom template file for the escalated incident
# template_tile=/usr/integration/bit9_escalation.jinja

# Optional: set this to only escalate a single request ID, e.g. when testing a
custom template
# test_single_request=999
```

## Create a Custom Layout

In order to view the status of the Carbon Black Protection integration, go to the Layout tab in Resilient and create an Incident Tab named Cb Protection. Drag the Cb Protect incident fields from the Fields column on the right to the Incident: CbProtection tab column in the middle of the page and then hit the Save button.

The screenshot shows the Resilient platform's 'Customization Settings' interface under the 'Layouts' tab. A new incident tab named 'Cb Protection' is being created. The 'Fields' panel on the right lists various Carbon Black Protection-related fields, which are then being dragged into the 'Incident: Cb Protection' tab area in the center. The fields listed in the 'Fields' panel include:

- Address
- Alberta Health Risk Assessment
- Assessed Liability
- Cb Protect Computer
- Cb Protect Computer ID
- Cb Protect Details
- Cb Protect Enforcement Level
- Cb Protect File Catalog Id
- Cb Protect File Path
- Cb Protect Filename
- Cb Protect Request Id
- Cb Protect Request Priority
- Cb Protect Request Status
- Cb Protect Triggered On
- Cb Protect User
- Cb Protect File Path

The 'Incident: Cb Protection' tab currently contains the following fields:

- Cb Protect Computer
- Cb Protect Computer ID
- Cb Protect Details
- Cb Protect Enforcement Level
- Cb Protect File Catalog Id
- Cb Protect File Path
- Cb Protect Filename
- Cb Protect Request Id
- Cb Protect Request Priority
- Cb Protect Request Status
- Cb Protect Triggered On
- Cb Protect User

Now you can go to the Cb Protection tab and view that status of the Approval and Ban requests.

The screenshot shows the Resilient platform interface for a Cb Protection Approval Request. The top navigation bar includes 'Dashboards', 'Simulations', 'Incidents' (selected), and 'Create'. A search bar and user information ('Resilient Sys... resilient') are also present. The main title is 'Cb Protection Approval Request for firefox.exe - Cb Protect ...'. Below the title, there are two tabs: 'Summary' and 'Description'. The 'Summary' tab displays incident details: ID 2186, Phase Initial, Severity -, Date Created 09/04/2019, Date Occurred 09/04/2019, and Date Discovered 09/04/2019. It also shows 'Was personal information or personal data involved?' as No and 'Incident Type' as -. The 'Description' tab shows the test approval message: 'test approval 09/04/2019 11:10 a.m.'. Below these tabs is a horizontal navigation bar with links: Tasks, Details, Breach, Notes, Members, News Feed, Attachments, Stats, Timeline, and Artifacts. The 'Cb Protection' link is highlighted. The main content area is titled 'Cb Protection' and contains an 'Edit' button. It lists various parameters: Cb Protect Computer (WORKGROUP\RSWINDOWS10-3), Cb Protect Computer ID (1), Cb Protect Details, Cb Protect Enforcement Level (Low (Monitor Unapproved)), Cb Protect File Catalog Id (32013), Cb Protect File Path (c:\users\rswindows10-3\appdata\local\mozilla firefox), Cb Protect Filename (firefox.exe), Cb Protect Request Id (46), Cb Protect Request Priority (Medium), Cb Protect Request Status (Submitted), Cb Protect Triggered On (09/04/2019 11:11:19), and Cb Protect User (a@a.com). At the bottom left, there is a 'People' section with 'Created By' and 'Owner' both listed as 'Resilient Sysadmin', and 'Members' noted as 'There are no members.' Finally, a 'Related Incidents' section lists three items: '#2188 Cb Protection Approval Request f...', '#2187 Cb Protection Approval Request f...', and '#2185 Cb Protection Approval Request f...'. The entire interface is framed by a light gray border.

## Function Descriptions

Once the function package deploys the functions, you can view them in the Resilient platform Functions tab, as shown below. The package also includes example workflows and rules that show how the functions can be used. You can copy these workflows and rules for your own needs.

The screenshot shows the Resilient platform interface with the 'Functions' tab selected. The page title is 'Customization Settings' and the sub-section is 'Functions'. A search bar is at the top left, and a 'New Function' button is at the top right. The main area displays a table of functions with columns for Name and Description. Each row has a delete icon on the far right. The functions listed are:

Name	Description
CbProtect Delete File	Deletes a file from provided systems.
CbProtect Delete File Rule for Id	Delete a file rule.
CbProtect Get Approval Request for Id	Get an approval request by ID.
CbProtect Get Approval Request for Query Condition	Return approval requests that match the given criteria.
CbProtect Get File Catalog for Id	Get a file catalog item by ID.
CbProtect Get File Catalog for Query Condition	Return file catalog objects that match the given criteria.
CbProtect Get File Instance for Query Conditions	Return file instance objects that match the given criteria.
CbProtect Get File Rule for Id	Get a file rule by ID.
CbProtect Get File Rule for Query Condition	Return file rules that match the given criteria.
CbProtect Update Approval Request	Update an approval request.
CbProtect Update File Instance Local State	Update the approval state of a file instance.
CbProtect Update File Rule	Create or update a File Rule.

At the bottom center of the page, there is a copyright notice: © Copyright IBM Corporation 2019.

## bit9\_approval\_request\_get: CbProtect Get Approval Request for Id

Given an approval request's ID, the function returns the details of the approval request. The function takes one input, bit9\_approval-request\_id, which is a number. The following is an example of this function in the (Example) CbProtect Get Approval request workflow.

Customization Settings

Workflows / (Example) CbProtect Get Approval Request

Name \* (Example) CbProtect Get Approval Request

API Name \* cbprotect\_get\_approval\_request

Description When a Carbon Black approval request is received, fetch the details including the File Catalog ID, and create hash artifacts.

Object Type \* Incident

Creator Orchestration Engine Last Modified 06/19/2019 11:58 Last Modified By Orchestration Engine Associated Rules (Example) CbProtect Get Approval Request

```
graph LR; Start(( )) --> Incident(( )); Incident --> ApprovalRequest{CbProtect Get Approval Request for Id}; ApprovalRequest --> FileCatalog{CbProtect Get File Catalog for Id}; FileCatalog --> End(( ));
```

This workflow should be triggered when the incident is created. Read full details of the approval request. Update the incident with the file catalog item. Get the file information and write hash artifacts to the incident.

Input

Input Parameter	Value
bit9_approval_request_id	[Empty Input Field]

## bit9\_approval\_request\_query: CbProtect Get Approval Request for Query Condition

This function takes one input, bit9\_query which is a query string, and returns the approval requests that match the given query condition. The following is an example of this function in the (Example) CbProtect Get Appr Req for Q 'fileName:notepad.exe' workflow. You can set a different query condition following the guidelines

<https://developer.carbonblack.com/reference/enterprise-protection/8.0/rest-api/#query-condition>

and review the all approval request properties to query here

<https://developer.carbonblack.com/reference/enterprise-protection/8.0/rest-api/#approvalrequest>.  
fileName:notepad.exe represents name of the file on the agent.

The screenshot shows the Resilient platform interface for creating a new workflow. The top navigation bar includes 'Dashboards', 'Simulations', 'Incidents', 'Create', and search/filter options. The main title is 'Customization Settings' under the 'Workflows' tab. The workflow details are as follows:

- Name:** (Example) CbProtect Get Appr Req for Q 'fileName:notepad.exe'
- API Name:** example\_cbprotection\_query\_approval\_request
- Description:** Queries for approval requests based on the provided query.
- Object Type:** Incident

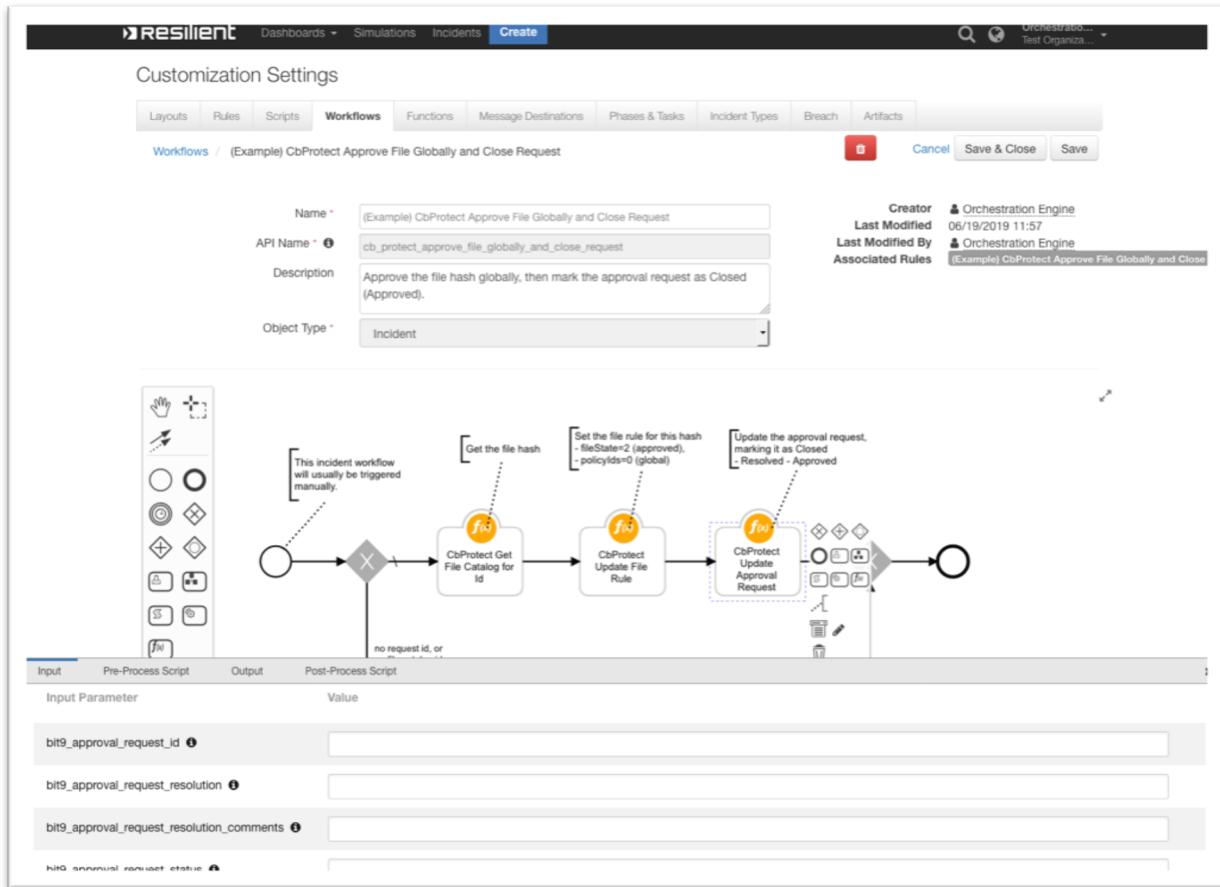
**Creator:** Orchestrator Engine, Last Modified: 06/19/2019 11:57, Last Modified By: Orchestrator Engine. Associated Rules: (Example) CbProtect Get Approval Request for Query Condition.

The workflow diagram consists of a start node, a function node labeled 'CbProtect Get Approval Request fo...', and an end node. Annotations explain the function's purpose: 'Queries for approval requests with the file name notepad.exe' and 'A note is created with the results'. A tooltip 'Start your workflow here' points to the start node. The workflow is associated with the rule '(Example) CbProtect Get Approval Request for Query Condition'.

Input Parameter: bit9\_query Value: fileName:notepad.exe

## bit9\_approval\_request\_update: CbProtect Update Approval Request

This function accepts as input a request ID, approval request resolution, comments, and status. With these, it updates an approval request. The following is an example of this function in a workflow:



## bit9\_file\_catalog\_get: CbProtect Get File Catalog for Id

Returns the file catalog details based on the catalog ID provided. The following is an example of this function in the (Example) CbProtect Approve File Globally and Close Request workflow:

The screenshot displays the Resilient platform's 'Customization Settings' interface for a workflow. The workflow is titled '(Example) CbProtect Approve File Globally and Close Request'. The configuration includes:

- Name:** (Example) CbProtect Approve File Globally and Close Request
- API Name:** cb\_protect\_approve\_file\_globally\_and\_close\_request
- Description:** Approve the file hash globally, then mark the approval request as Closed (Approved).
- Object Type:** Incident
- Creator:** Orchestration Engine
- Last Modified:** 06/19/2019 11:57
- Last Modified By:** Orchestration Engine
- Associated Rules:** (Example) CbProtect Approve File G...

The workflow diagram shows the following sequence:

```
graph LR; Start(( )) --> Decision{ }; Decision -- "no request id, or no file catalog id" --> End(( )); Decision --> GetFileHash[CbProtect Get File Catalog for Id]; GetFileHash --> SetRule[Set the file rule for this hash  
- fileState=2 (approved),  
- policyIds=0 (global)]; SetRule --> UpdateApproval[CbProtect Update Approval Request]; UpdateApproval --> End;
```

Input parameters:

Input Parameter	Value
bit9_file_catalog_id	(empty)

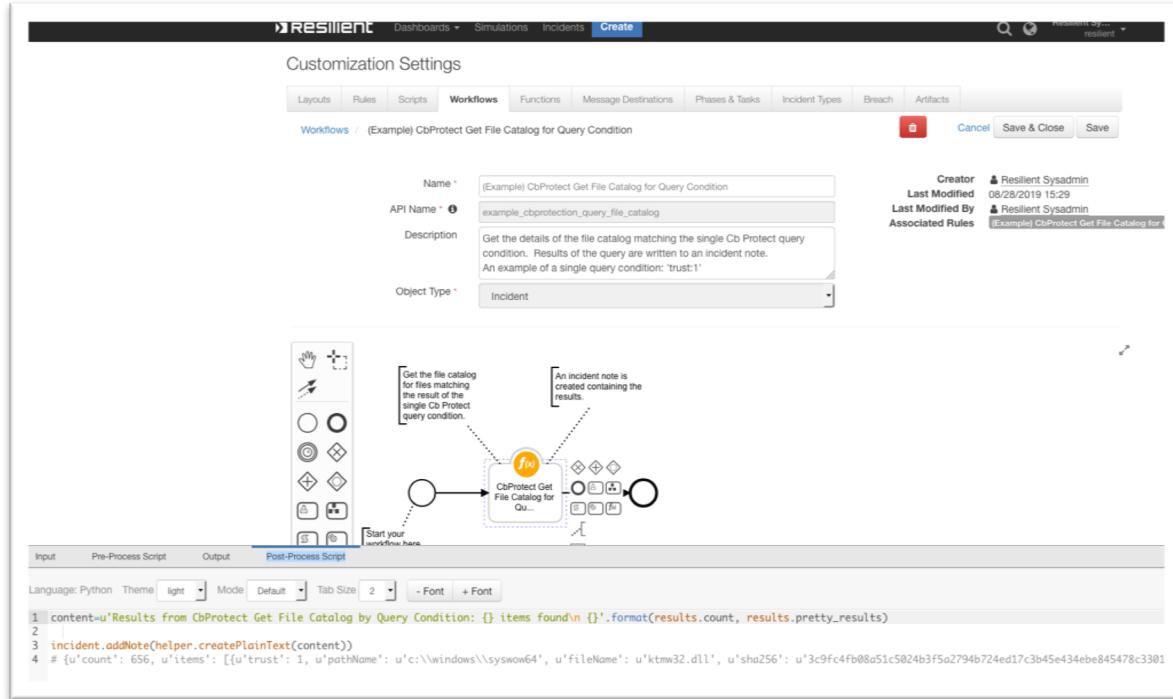
## bit9\_file\_catalog\_query: Cbprotect Get File Catalog for Query Condition

Returns file catalogs and their details from a provided query string. The following is an example of this function in (Example) CbProtect Get File Catalog for Query Condition workflow. You can set a different query condition following the guidelines

<https://developer.carbonblack.com/reference/enterprise-protection/8.0/rest-api/#query-condition>

and review the all file catalog properties to query here

<https://developer.carbonblack.com/reference/enterprise-protection/8.0/rest-api/#filecatalog>.



## bit9\_file\_instance\_query: CbProtect Get File Instance for Query Conditions

Returns file instance objects that match the given criteria from the inputs. The following is an example of this function in the (Example) CbProtect Approve File Locally and Close Request workflow:

Customization Settings

Workflows / (Example) CbProtect Approve File Locally and Close Request

Name: (Example) CbProtect Approve File Locally and Close Request

API Name: cb\_protect\_approve\_file\_locally\_and\_close\_request

Description: Approve the file locally, then mark the approval request as Closed (Approved).

Object Type: Incident

Creator: Orchestrator Engine

Last Modified: 06/19/2019 11:57

Last Modified By: Orchestrator Engine

Associated Rules: (Example) CbProtect Approve File

This incident workflow will usually be triggered manually.

Locate the file instance

Update the file instance setting local approval

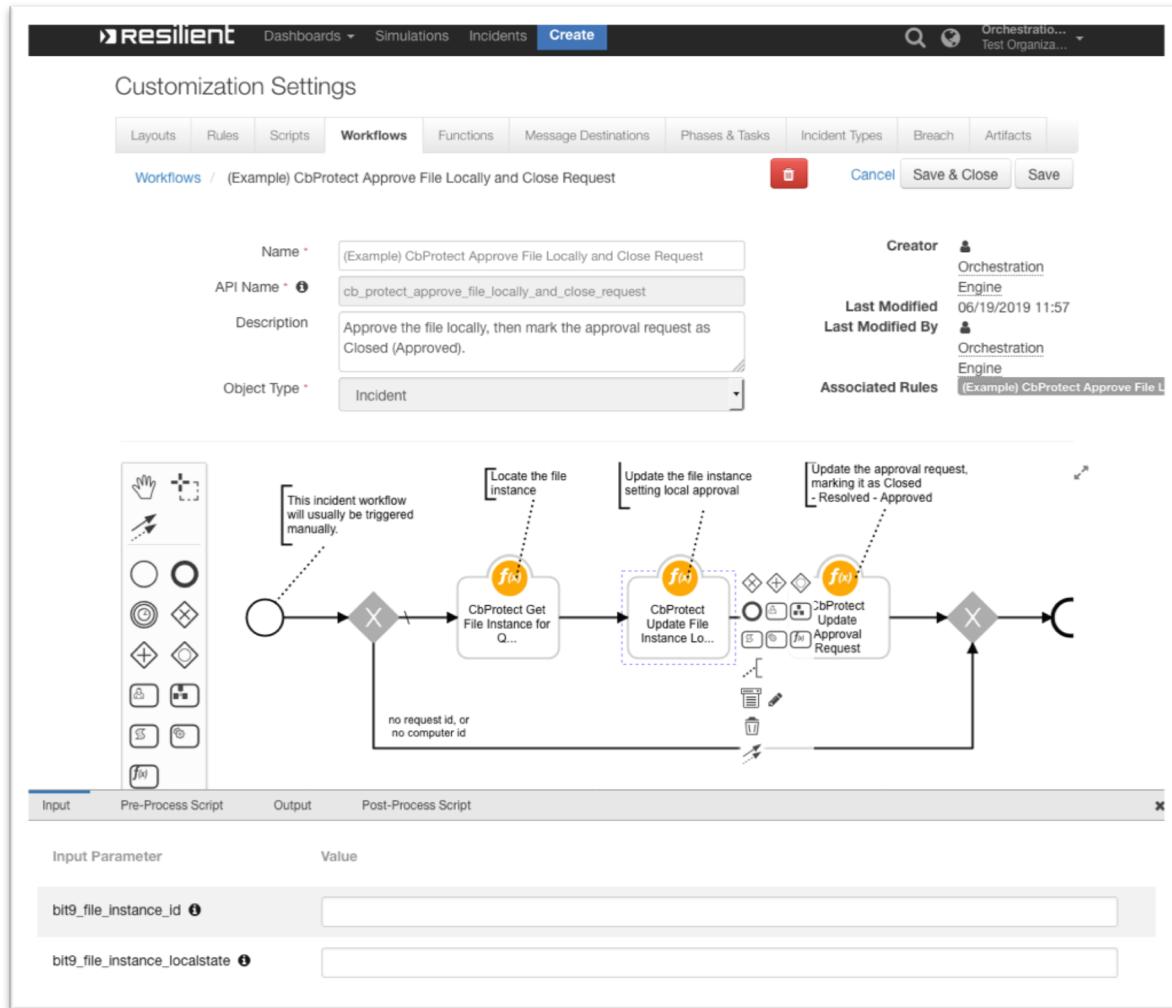
Update the approval request, marking it as Closed - Resolved - Approved

Input Parameter Value

- bit9\_computer\_id
- bit9\_file\_catalog\_id
- bit9\_file\_path
- bit9\_file\_instance\_localstate

## bit9\_file\_instance\_update: CbProtect Update File Instance Local State

Updates a file instance's local approval/banned setting. This function has inputs for the file instance ID and the local state (for example, approved = 2). The following includes an example of this function in the (Example) CbProtect Approve File Locally and Close Request workflow:



## bit9\_file\_rule\_update: CbProtect Update File Rule

This function updates a file rule in Carbon Black based on the data passed as inputs. The following is an example of this function in the (Example) CbProtect Approve File Globally and Close Request workflow:

Customization Settings

Workflows / (Example) CbProtect Approve File Globally and Close Request

Name: (Example) CbProtect Approve File Globally and Close Request  
API Name: cb\_protect\_approve\_file\_globally\_and\_close\_request  
Description: Approve the file hash globally, then mark the approval request as Closed (Approved).  
Object Type: Incident

Creator: Orchestrator Engine  
Last Modified: 06/19/2019 11:57  
Last Modified By: Orchestrator Engine  
Associated Rules: (Example) CbProtect Approve File

```
graph LR; Start(( )) --> Decision{ }; Decision -- "no request id, or no file catalog id" --> End(( )); Decision -- "This incident workflow will usually be triggered manually." --> GetCatalog[CbProtect Get File Catalog for Id]; GetCatalog --> UpdateRule[CbProtect Update File Rule]; UpdateRule -- "Set the file rule for this hash - fileState=2 (approved), - policyIds=0 (global)" --> ApprovalRequest[cbProtect Update Approval Request]; ApprovalRequest -- "Update the approval request, marking it as Closed - Resolved - Approved" --> End;
```

Input Parameter Value

bit9_file_rule_id	1
bit9_file_catalog_id	
bit9_file_rule_name	
bit9_file_rule_description	
bit9_file_rule_filestate	
bit9_file_rule_sourcetype	5
bit9_file_rule_policyids	
bit9_file_rule_hash	

## **bit9\_file\_delete: Cb Protect Delete File**

This function deletes a file by hash or filename using a Cb Protection API call based on the data passed as inputs. The Cb Protection API call that is used in the function is a beta solution and should not be used in production.

## **Carbon Black Protection Resilient Polling Component**

This integration contains a polling component that automatically escalates approval requests into the Resilient platform. To enable this feature, the `escalation_interval` variable in the `app.config` file must be set to an integer greater than 0. This integer represents the interval in number of seconds for the automatic escalation of approval requests. It is recommended to start at 300, which checks every 5 mins.

You can also set optional values, such as `escalation_query`, which escalates approval requests that match the query; if not set, it defaults to all open approval requests. In addition, you can set `template_file` to the location of a custom jinja template file; if not set, the default template file is used. To create your own custom jinja file, you should use the default jinja file as a reference. This file can be found when expanding the package in the following directory:

```
fn_cb_protection-<version#>/fn_cb_protection/data/
```