Azure Config Queue Integration

v1.0

Table of Contents

[Introduction 3](#_Toc514683063)

[High Level Flow 3](#_Toc514683064)

[Initial ServiceNow Configuration 3](#_Toc514683065)

[Azure Configuration 3](#_Toc514683066)

[ServiceNow Configuration 3](#_Toc514683067)

[Update set 3](#_Toc514683068)

[Azure Queue Integration Configuration 4](#_Toc514683069)

Introduction

For customers who have enabled IP Access Controls having Azure Configuration alerts forwarded to the instance can be a challenge due to the large ranges of IPs that may need to be whitelisted.

This document outlines an alternative approach, put together by the ServiceNow ITOM Rangers, utilizing an Azure Storage Queue to receive configuration notifications, and a probe that is executed on a MID server to collect the queue messages. This setup removes the need to whitelist any IPs.

High Level Flow

Azure first collects and forwards configuration alerts via configuration rules and configuration rule action groups to an Azure LogicApp. The Azure LogicApp, in turn, forwards the configuration alerts to an Azure storage queue. The ServiceNow instance executes a JavaScript probe on the MID server that collects messages from the Azure storage queue. The sensor for the probe then utilizes the out-of-the-box Azure config processor to process the configuration messages.

NOTE: This document outlines the use of a LogicApp to forward Azure Alerts to an Azure queue, however it is also possible to use an Azure Function if desired.

Initial ServiceNow Configuration

The first step in the configuration process is to setup the out-of-the-box Azure configuration integration following the steps outlined on the ServiceNow documentation site. This will create the necessary alert rules that will be used in Azure to collect the Azure configuration alerts. You do not need to make the scripted web service public since you will be pulling alerts from Azure instead of forwarding them to the instance.

Azure Configuration

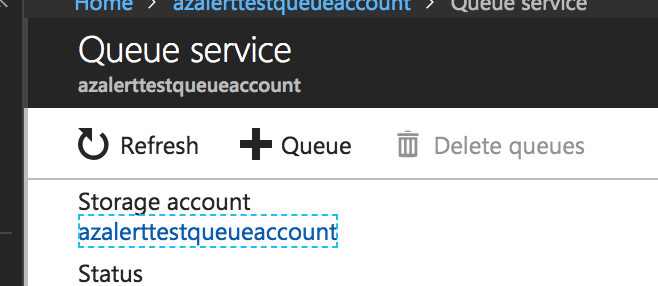
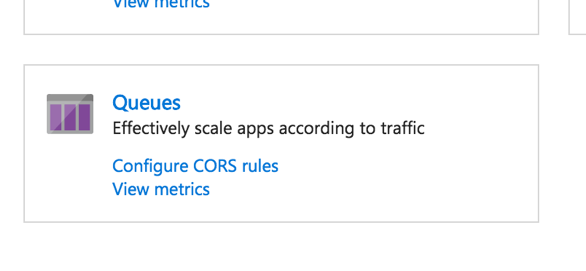
The out-of-the-box Azure Alert Configuration will setup the necessary alert rules and action groups necessary to send events directly to the ServiceNow instance. However, updates will need to be made to forward events to the queue.

Create or Select an Azure Storage Account

An Azure Storage Queue must be created in an Azure Storage Account. If you already have a Storage Account setup in Azure and you wish to use this account then you can proceed to create the Azure queue. Otherwise, using the +Create a resource option in Azure, create a new Storage Account. The Storage Account must be compatible with Azure Queues.

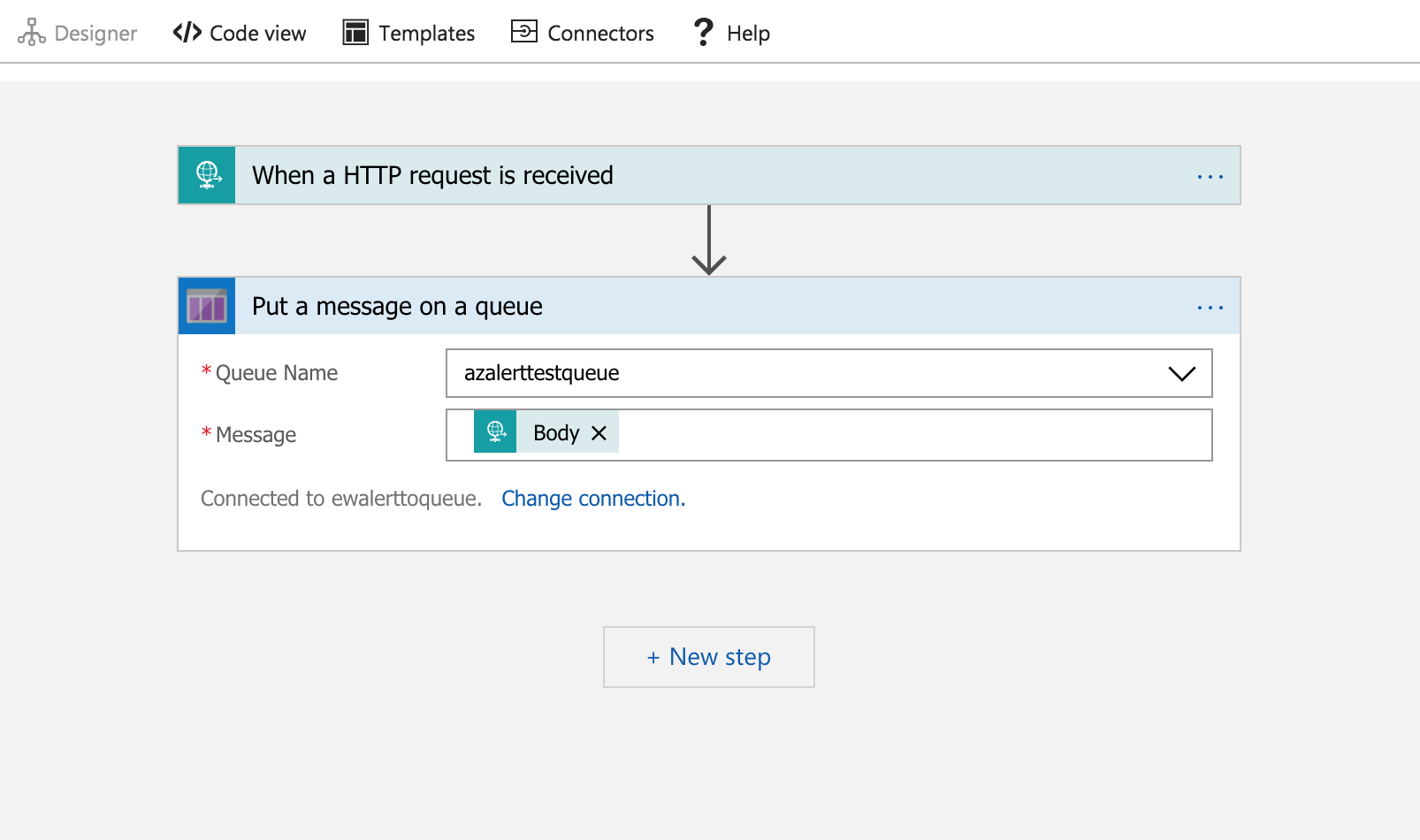
Create an Azure Queue

Once you have created an Azure Storage account you can create an Azure Storage Queue by going into your storage account, selecting the Queues tile and then use the +Queue button.

Create a LogicApp

Once the queue is created a LogicApp will need to be created to forward incoming Alerts to the Storage Queue. Use the +Create a resource option and select a LogicApp. The trigger for the App will be “When an HTTP request is received”. The next step will be to “Put a message on a queue”. Select the previously created queue as the queue name. Put the Body of the HTTP request as the Message contents. The resulting app should look similar to this screenshot.



Create an Action Group

The last step is to create an Action Group to forward the alerts to the LogicApp. Go to All Services -> Monitor and select Action Groups under Settings in the Monitor blade. Here you will see the Action Group created by ServiceNow. Open this Action Group and you will see the Webhook Action created by ServiceNow. You can remove this Action to stop Azure from forwarding alerts directly to the instance. Add a new Action of type LogicApp and select your previously created LogicApp. Once you save your new Action Group Action you are ready to proceed to the final ServiceNow configuration.

NOTE: It may be necessary to run a discovery of the Logical Datacenter in which you configured the Azure Storage Account so the Storage Account is available for selecting during the ServiceNow configuration.

ServiceNow Configuration

The rest of the configuration for the Azure Queue integration will be done on the ServiceNow side.

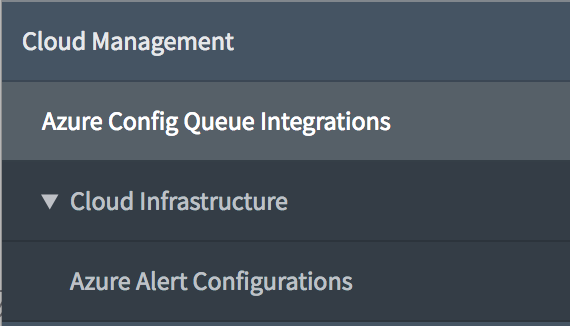
Update set

There is one update set to apply:

* Azure Config Queue Integration v1.0
  + Azure Queue Probe/Sensor
  + Azure Queue Mid Server Script include
  + Azure Storage Token Patch Mid Server Script includes
  + Azure Queue Configuration Reference Qualifier Script Include
  + Azure Queue Probe Runner Script include
  + Azure Queue Configuration Integration table and business rules

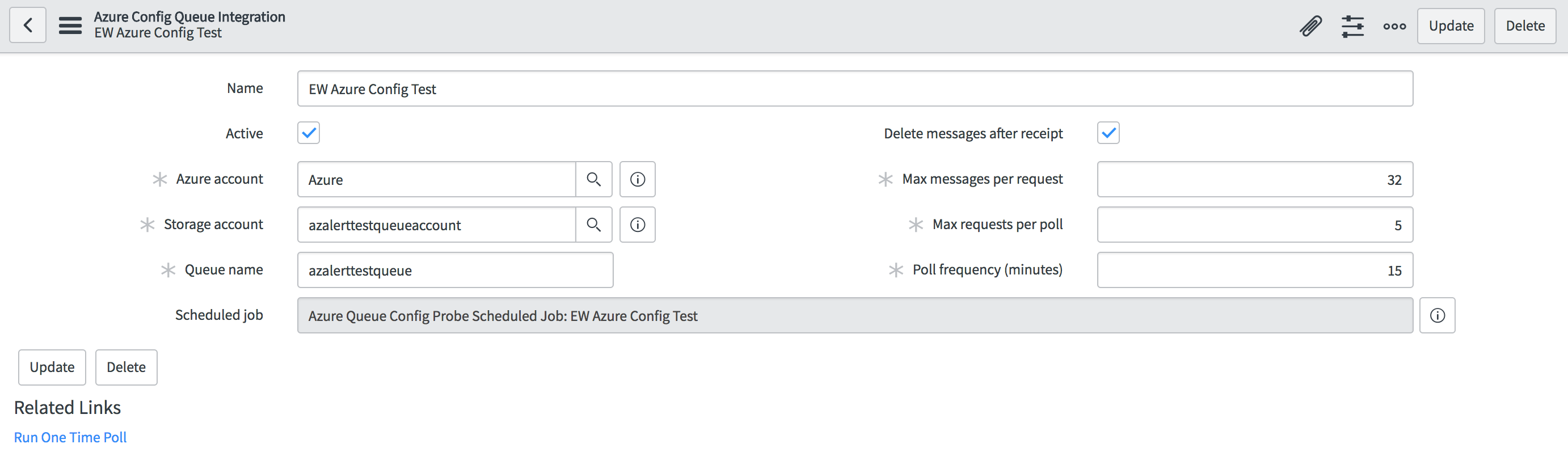
MID Server configuration

A MID server that can access the public Azure API and explicitly configured with the “Azure” capability is required. The “All” capability is not sufficient.



Azure Queue Integration Configuration

Once the update set has been applied a new menu option will be available – Azure Config Queue Integrations. Selecting this option will take you to the table of Azure Config Queue Integration configurations.



The Configuration Page consists of several options.

* **Name**: Name of the Azure Config Integration Configuration. Updates to the name of the config cascade to the underlying scheduled job for ease of reference.
* **Active**: Marks this configuration as active or inactive – changing this flag will cascade down to the underlying scheduled job
* **Azure** **Account**: Azure Service Account that will be used when connecting the Azure queue. This must match the account under which the Azure queue is defined.
* **Storage Account**: Azure Storage Account in which the queue is defined. This must match the storage account the Azure queue is defined in.
* **Queue name**: Name of the Azure Queue to collect configuration events from.
* **Delete messages after receipt**: Flag to indicate if messages should be removed from the queue after reception. This is recommended unless you have other processes also relying on the queue and cleaning up the messages. Not checking this may result in the same message being processed over and over.
* **Max messages per request**: Azure supports a maximum return size of 32 messages per request. This value can be set from 1 to 32.
* **Max requests per poll**: The maximum number of requests the probe will attempt against the Azure queue before returning the results to the instance. If any request results in no returned messages prior to the max requests per poll being reached, then polling for that cycle will stop.
* **Poll frequency (minutes)**: Time, in minutes, between Azure queue polls. A poll is a series of Azure queue requests to the Azure queue. Each request will be for a number of messages up to the “Max messages per request” messages from the Azure queue. During a single poll requests are repeated until either the max requests per poll is reached OR the Azure queue returns an empty response.
* **Scheduled Job**: A read-only field with a reference link to the underlying scheduled job. The scheduled job is created when the initial configuration record is created and is removed when the configuration record is removed via business rules. Business rules are also used to synchronize the Active flag, Job name, and Job Frequency with the configuration entry.
* **Run One Time Poll UI Action**: This action will initiate a single poll utilizing the current configuration. The one-time poll executes regardless of the state of the Active flag. The poll will update CIs if Azure config messages are in the queue.