Module 27 – Amazon EventBridge – Part 2

Overview

In the previous module you used the EventBridge SDK to put an event in an EventBridge bus. An application usually does that when it wants to advertise that some state has changed. Other components in the system can react to that change or ignore it. EventBridge **rules** are used to check for messages, filter them out (e.g., a component in your system might be interested in accounts that got deactivated for some reason), and forward events to other systems (e.g., a Lambda, queue, etc.) for processing (Figure 1).

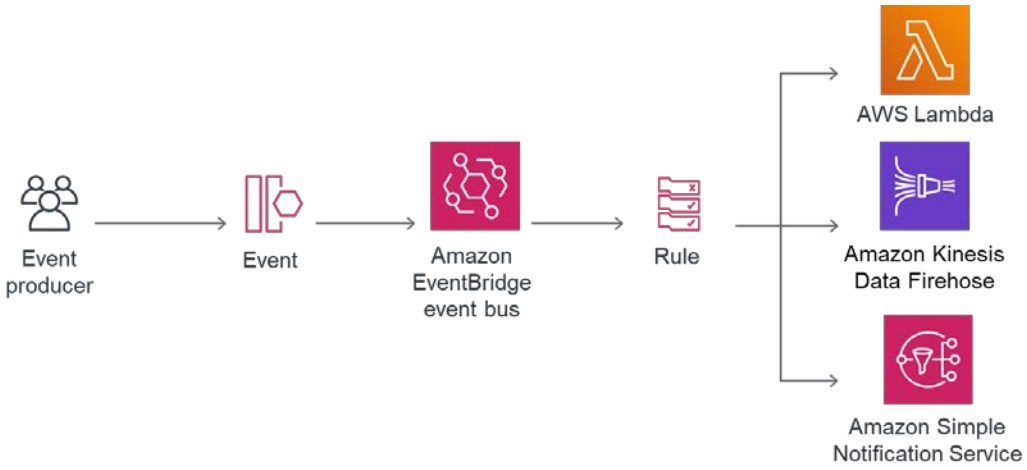


Figure 1: Use EventBridge **rules** to filter out events of interest and forward them to other systems.
This picture shows 3 systems on the right (Lambda, Kinesis Data Firehose, and SNS). These are example services that events can be forwarded to via a rule. However, this is not all – a rule can forwarded events to more services that what is shown above.

One important detail that is not evident from Figure 1 is that a rule not only forwards events but also filters them out. That is, when you define a rule you configure it to behave like "I want to forward events having these values and ignore others". For example, assume that your application has an AccountStatus concept. The AccountStatus can be: *Initiated*, *Active*, *Locked*, or *Deleted*. Every time the AccountStatus value changes your system submits an event. However, assume that you want to do an additional action if AccountStatus becomes *Locked* (but not when it is *Initiated*, *Active*, or *Deleted*). In this case you can create a rule that filters on "AccountStatus" : "Locked". Now this rules only forwards the event to downstream target services when AccountStatus is equal to *Locked*.

In the previous module you created an application that put some events in an EventBridge event bus (the 3 steps starting from the left in Figure 1). **If you are not done with it, stop here and complete it now. You cannot do this module if you don't have events in a bus**. Resume here only after you finish

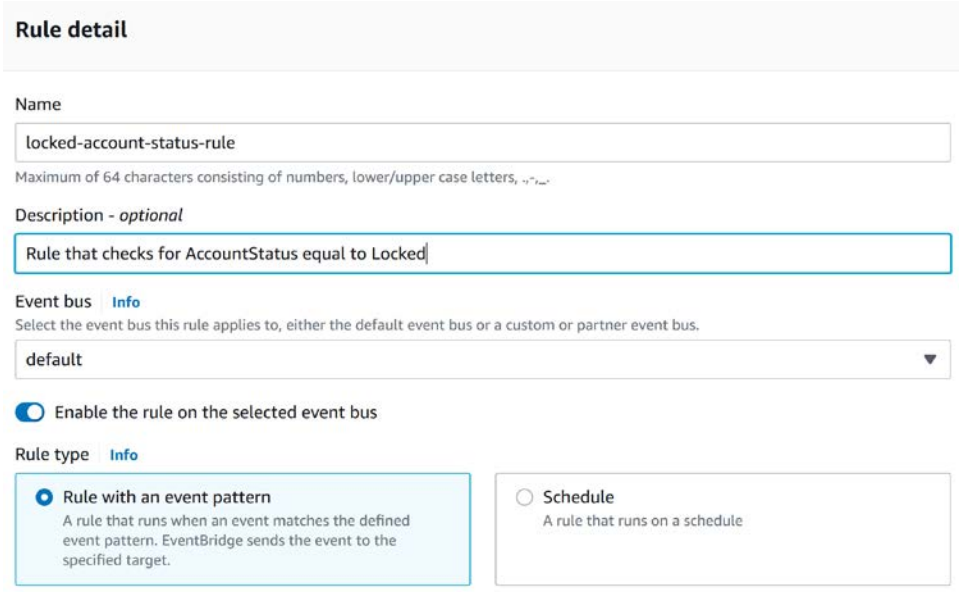
the previous module.

Create an EventBridge Rule

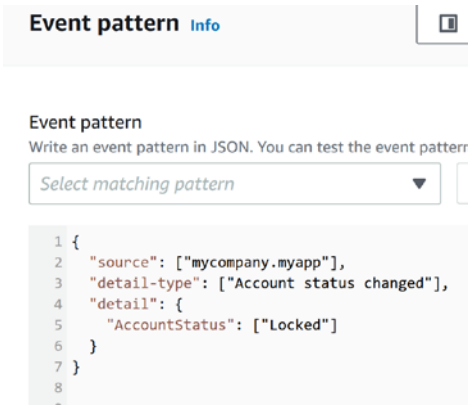
1. Go to Canvas, start the lab, and go to the AWS console.

2. Go to the **EventsBridge** service. Then click on **Rules** from the navigation pane on the left.

3. Click the **Create rule** button. Then complete the **Rule detail** section with something that resembles the below (change the rule name and description to match the event data you sent).



4. Click the **Next** button.
   Scroll down to the **Event pattern** section. Click button **Custom patterns (JSON editor)**. Enter something like the below (change the JSON to reflect your event data - What I am showing below is just an example). Note that I am creating my event to filter on AccountStatus = "Locked". My fictitious application is interested in acting on this event.



5. Click the **Next** button. In the **Select target(s)** section:

    a. Choose AWS service option
    b. From the **Select a target** dropdown, choose SQS queue.
    c. From the Queue dropdown, select an existing queue you have (note: if you don't have any queue, open a new browser tab, go to the SQS service, and create a queue).

    Click the **Next** button.

6. Click **Next** on the Tags page.

7. On the **Review and create** page, click the **Create rule** button.

Testing the Rule

1. Use your console app from the previous module to put a few events (send many events where some match the rule you created and some don't). For example, send 5 events where 3 match the rule and 2 don't.

2. Go to the queue and observe that you have 3 events only. If it is the case, then your rule did the following:
    a. It correctly filtered out events based on your application logic needs.
    b. It correctly forwarded your events to the target you specified (the queue).

Go back to your event bridge rule and notice that you could have selected a target other than a queue (e.g., a Lambda or something else). There is a good variety of target options to choose from.

What to Submit

Nothing to submit for this module.