Real-Time Big Data Processing

Lab 5 – Final Challenge

Overview

In this final challenge lab, you will out into practice some of the techniques you have learned in this course to build a streaming data solution that aggregates real-time data from a simulated building security system.

What You'll Need

To complete the labs, you will need the following:

- A web browser
- A Microsoft account
- A Microsoft Azure subscription
- A Windows, Linux, or Mac OS X computer
- The lab files for this course

Note: To set up the required environment for the lab, follow the instructions in the Setup document for this course. Specifically, you must have signed up for an Azure subscription and installed Node.JS on your computer.

Challenge 1: Capture Security Events in an Event Hub

The scenario for this lab is based on a simulated building security system. In this system, entry to the building is controlled by nine turnstiles, which can only be accessed by presenting a valid, uniquely identifiable, keycard. The system logs each entry through a turnstile and emits a JSON record containing the turnstile, card number, and entry time.

Your first challenge is to create an Event Hub, and then configure a Node.JS client application to submit details of each turnstile entry event to the Event Hub.

1. Create an Azure Event Hub.
2. Add a Shared Access Policy to the Event Hub. This should provide the minimum permission required for a client application to send messages to the Event Hub.
3. Modify the **turnstile.js** script in the **turnstile** folder where you extracted the lab files to replace the placeholder text ***<EVENT-HUB-CONNECTION-STRING>*** with a valid connection string for the Shared Access Policy you created for the Event Hub.
4. Initialize a Node.JS application in the turnstile folder to create a **package.json** file for your solution, specifying **turnstile.js** as the entry point.
5. Import any packages that your Node.JS application requires to access your Event Hub.

6. Test your application by running the following command:

   ```
   node turnstile.js
   ```

7. Observe the application as it runs. It may wait for up to a minute to start generating security events, and then it will display the JSON security events as they occur. Leave the application running until the script completes (this will take several minutes).

Challenge 2: Use a Stream Analytics Job to Aggregate Security Events

Now that you have a solution for capturing security events in an Event Hub, you are ready to process the captured data as it arrives in real-time.

You must implement a Stream Analytics job that reads the turnstile data from the Event Hub and counts the number of entries for each turnstile within one minute time intervals. The results of the job should be stored in an Azure Storage blob container, and they should contain an entry for each turnstile that was accessed in non-overlapping, fixed width, contiguous time intervals of one minute.

The results should include the following fields:

- **windowstart**: The time one minute before the timestamp for the window end.
- **windowend**: The timestamp for the end of the window. This should be based on the **entrytime** field.
- **turnstile**: The turnstile number
- **entries**: A count of the number of entries through the turnstile within the temporal window.

For example, your solution should produce results similar to the following:

```
windowstart,windowend,turnstile,entries
2017-01-18T05:30:00.0000000Z,2017-01-18T05:31:00.0000000Z,7,4
2017-01-18T05:30:00.0000000Z,2017-01-18T05:31:00.0000000Z,3,12
2017-01-18T05:30:00.0000000Z,2017-01-18T05:31:00.0000000Z,2,9
2017-01-18T05:31:00.0000000Z,2017-01-18T05:32:00.0000000Z,7,7
2017-01-18T05:31:00.0000000Z,2017-01-18T05:32:00.0000000Z,4,8
2017-01-18T05:31:00.0000000Z,2017-01-18T05:32:00.0000000Z,2,4
2017-01-18T05:32:00.0000000Z,2017-01-18T05:33:00.0000000Z,7,2
2017-01-18T05:32:00.0000000Z,2017-01-18T05:33:00.0000000Z,3,1
```

In these sample results, the first window spans from 05:30:00 to 05:30:31 in which there were 4 entries through turnstile 7, 12 entries through turnstile 3, and 9 entries through turnstile 2. The next window spans from 05:31:00 to 05:32:00 in which there were 7 entries through turnstile 7, 8 entries through turnstile 4, and 4 entries through turnstile 4. The final window spans from 05:32:00 to 05:33:00 in which there were 2 entries through turnstile 7 and 1 entry through turnstile 3.

1. Create an Azure Storage account.
2. Create a second Shared Access Policy for the Azure Event hub with the minimum permissions required to read event messages from the Event Hub.
3. Create an Azure Stream Analytics Job, and configure it as required to:
   a. Read the events in JSON format from the Event Hub using the Shared Access Policy you created in the previous step.
   b. Save the results of the job in a container in the Azure storage account, in a folder hierarchy based on the year, month, and date.

    c. Process the data using a query that generates the output in the required format.
4. Run the Azure Stream Analytics job, and then start the **turnstile.js** Node.JS application and wait until it has finished.
5. Review the output generated by the job, and verify that it contains the required output.
6. Stop the Azure Stream Analytics job.
7. Run the Azure Stream Analytics Job again, and restart the **turnstile.js** Node.JS application and wait until it has finished. This should create a new output file that contains the output for exactly one run of the simulation.
8. Review the output file.