UNISYS | Securing Your Tomorrow®

Security Opportunities

AWS and Azure IoT Services

By: David Bennett
Brent Lee George
Sanket Arvind Panchamia

White Paper

Table of Contents

Background

In the past decade, there has been a massive increase in IoT (Internet of Things) connected devices. These devices range from complex medical equipment to simple Wi-Fi connected switches. Unfortunately, many of these devices lack proper security and are vulnerable to malicious attacks.

Amazon offers an IoT solution called AWS IoT, which provides an end-to-end solution for device management and security. The central service for creating and managing IoT Devices in Azure is IoT Hub. It is built on top of Azure Event Hub. The main feature is routing messages from devices. It also contains ways to manage IoT device and edge device configurations.

The goal of this paper is to identify gaps in these IoT solutions that can resolved. The Unisys Emerging Technology Blockchain and IoT team carried out this investigation.

Azure Security

Overall Azure contains most capabilities listed through SDKs if they are not available through a managed service. This allows high customizability in a solution. It offers an opportunity to the enterprises to fill the gaps by utilizing managed solutions and services of their managed service providers to do things that currently cannot be accomplished as they do not fall in the purview of Azure managed services.

Device Provisioning

Azure offers a managed service for Device Provisioning. Azure Device Provisioning Service is a managed service that provides automatic provisioning features with zero touch. It is a highly scalable and secure service that can provision up to millions of devices in a short period of time.

The steps involved in the DPS, as shown below in figure 1, are explained below

1. Manufacturer adds device registration information to the Azure Portal's enrollment list
2. Device contacts the DPS, passes its identity information to prove itself
3. DPS validates the identity against the enrollment list
4. DPS registers the device with an IoT Hub and populates the device's desired twin state
5. The IoT hub returns device information to DPS
6. DPS returns the IoT hub connection information to the device and the device can now start sending data to the Hub
7. The device connects to the Hub
8. The device gets desired state from its device twin in the Hub

Accessing Azure Devices

Two methods for accessing devices were explored. The first is the use of a device twin to do remote management operations. The Azure Device Agent is a service that runs on the device and detects changes in the device twin and acts accordingly to update underlying OS configurations or run commands that are passed directly via device twin.

Additionally, Azure Device Agent is an expandable offering with the ability for a device manufacturer or user to expand on the operations that it can handle. This area offers a slight opportunity in creating the service on the cloud side with a clean UI to update the device twin.

The second is a preview feature offered by Azure called Device Stream. Device stream allows the devices to run a proxy SSH server connection to the Azure Hub on a special message queue called the device stream. The client then runs a similar client proxy that also connect to the stream. From there an SSH session can be started to the device.

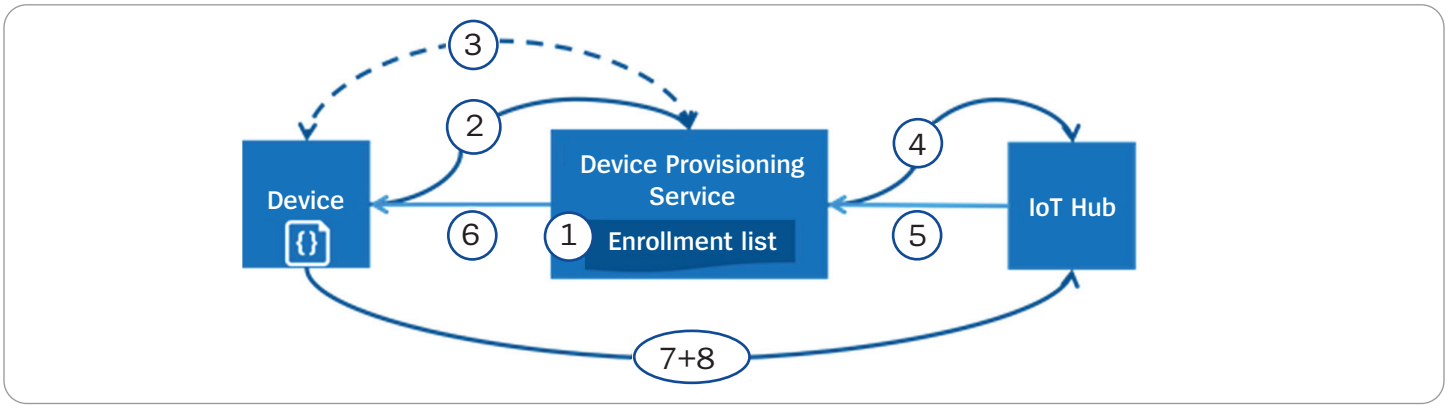

Figure 1: Azure Device Provisioning Service

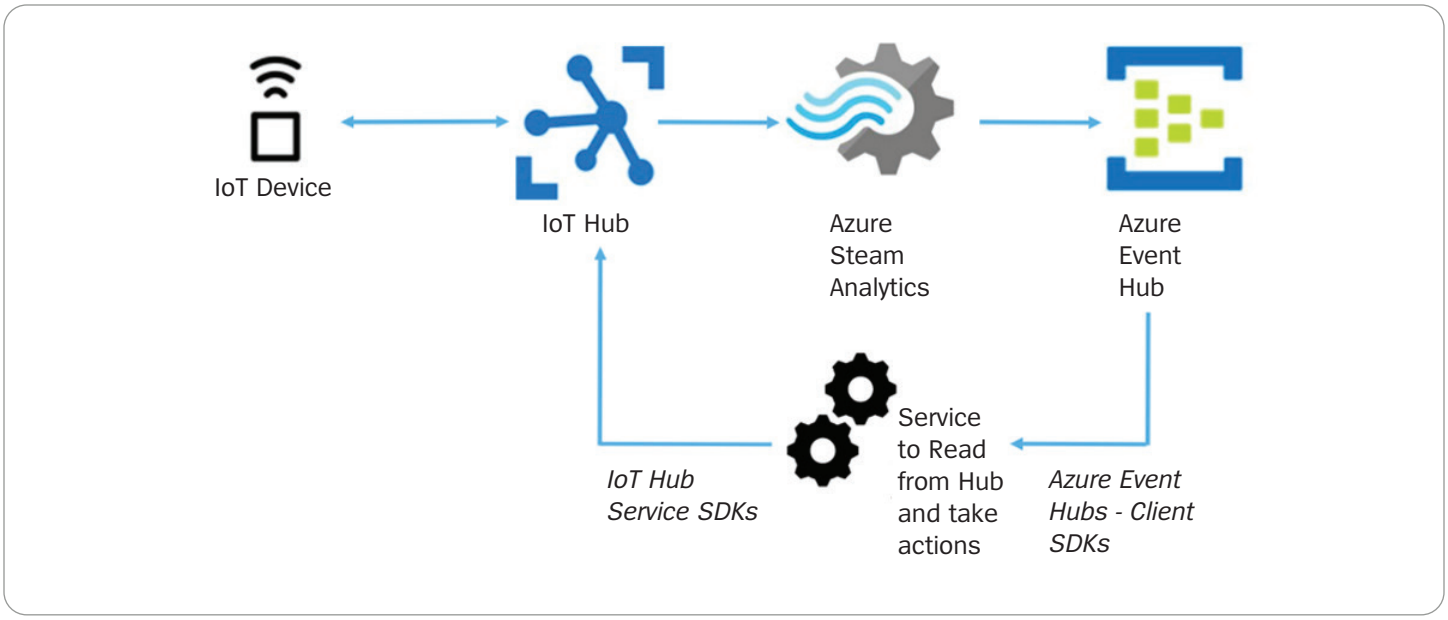

Figure 2: Azure Stream Analytics

Securely Updating Firmware

Azure offers a great way to do secure firmware updates using device twin. This is accomplished by setting the desired state for a device to a new level of the firmware. Through a set of SDKs, the device manufacturer can create a service on the device that periodically checks the device twin and when the update is found it pulls the image from a server and flashes the device.

This area offers an opportunity of writing the complete software to manage the device update of the device, as well as a service to update the device twin state with a new firmware version and monitor updates.

Securing Communication

All device communication in Azure is secured using mutual TLS. Certificates for this communication can be either self-signed X.509, CA signed X.509 or symmetric key.

Communication is secured from the device to the edge and from the edge to the cloud. This offers a full and complete solution for secure communication without any more tooling or code around the solution.

Anomaly Detection

Anomaly detection in Azure can be accomplished using Azure stream analytics. Stream analytics allows an easy way to take telemetric data and find issues (such as spikes and dips) in the data that are abnormal. Once detected, a service using the Azure SDK can read the anomalies and take appropriate action based on the data. This could disable the device in the environment.

This offers is an opportunity for enterprises and their managed service providers to add value in this area as the actual mitigation of issues is not handled natively by Azure and must be done through SDKs.

A tool can be conceived that allows users to define the type of anomalies, their triggers and actions. They can define the action that can be taken (i.e. remove device, limit device, or disable the device).

AWS Security

Device Provisioning

In AWS IoT, devices are provisioned either through the CLI or through AWS IoT console. Access to the CLI is controlled by the policies defined in AWS Identity and Access Management. Furthermore, it is encouraged to use temporary conditionals when performing operations through the CLI.

When devices are provisioned through AWS IoT, a X.509 certificate is generated along with a pair of public and private keys. A device must use this certificate and private key to establish a secure TLS connection to the AWS IoT service.

This offers a foreseeable opportunity to the organizations and their Managed Service Providers for device provisioning in AWS IoT.

Accessing AWS Devices

Device administrators might need to remotely access IoT devices to perform scheduled maintenance or troubleshoot issues. AWS provides a mechanism to do this through AWS Secure tunneling.

Secure tunneling provides a bidirectional communicate to remote devices.

The endpoint is secured with Identity and Access Management and Communication happens over Transport Layer Security. Once the Tunnel is initiated, authentication tokens are created and passed to the device. After a remote host authenticates to the Secure Tunneling service, a token will be delivered to both the host and IoT device.

There is no foreseeable opportunity for organizations and their managed service providers to securely accessing AWS devices besides the one currently offered by AWS Secure tunneling.

Securely Updating Firmware

If an embedded device is running the FreeRTOS operating system, then AWS IoT allows the firmware of the device to be securely updated Over-The-Air (OTA). A new firmware file is uploaded to Amazon S3 bucket and the devices are selected to perform the upgrade.

If the device is not running FreeRTOS, updates can still be performed through scheduled jobs. Scheduled jobs allow the device administrator to issue pre-defined remote commands to an IoT device. This could be anything ranging from system reboots, to connecting to a remote server.

There is no foreseeable opportunity in enhancing this functionality that's already offered by AWS.

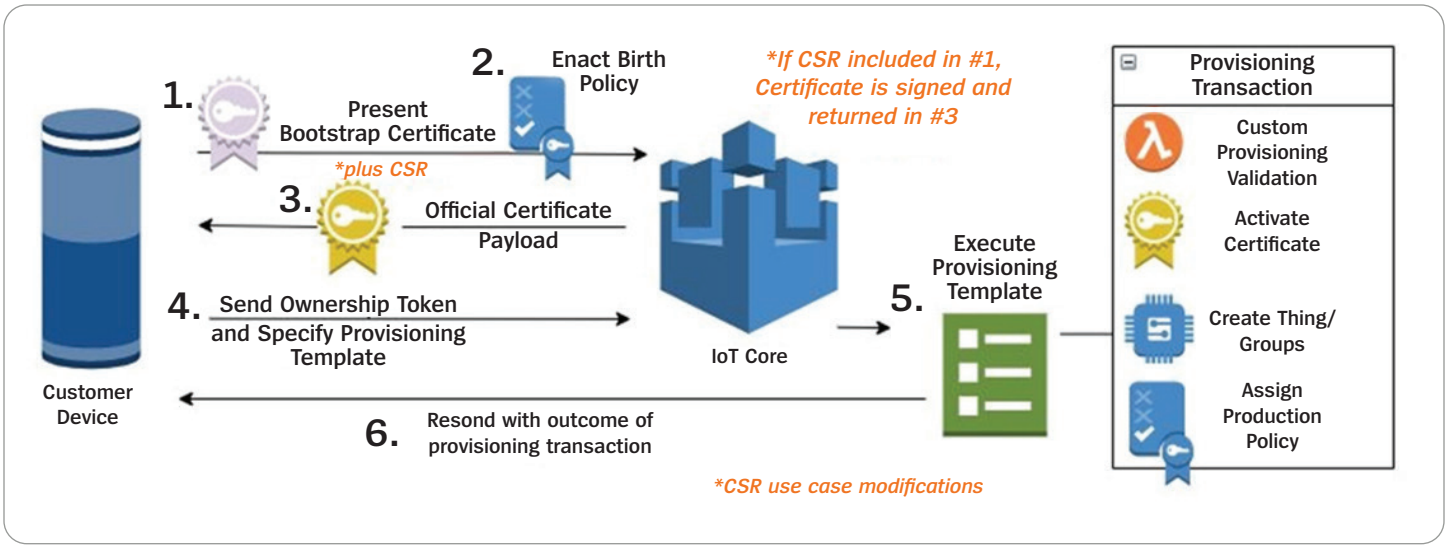

Figure 3: AWS CSR Use Case

Anomaly Detection

Detecting abhorrent behavior of IoT devices is possible through AWS Device Defender. AWS Device defender triggers an alert when metrics collected from an IoT device fall outside a pre-defined range.

AWS Device defender also performs a security audit of all registered IoT devices. If the audit reports a negative result for a particular device, a mitigation action can be taken to either quarantine or deactivate the device.

We are currently investigating the possibility of combing the offering of AWS Device Defender with Stealth Dynamic Isolation™.

Conclusion

The purpose of this effort was to identify any potential gaps in security with AWS IoT or Azure IoT services. This investigation also serves as a foundational guide for organizations and their service providers in filling those gaps.

The results of this investigation conclude that there are substantial opportunities for organizations and their service providers to add security enhancements in Azure vs AWS. Currently Azure lacks native anomaly detection capabilities, so there is an opportunity to create a tool that allows users to define the type of anomalies, their triggers and actions.

12/20

20-0700