AWS Connected Camera

OEM Guide

Integrate AWS IoT and KVS SDKs in the camera firmware.

Information regarding this step has been previously communicated and may be incorporated into this guide in the future.

Support ONVIF discovery of the camera

The config app uses the node-onvif library for camera discovery. It's expected that the camera publishes it's information such that the output from that tool includes at least the following fields:

```
{
  urn: "urn:uuid:fa92ecc4-b8ca-473f-a5a4-1e2e4824f9c0",
  name: "Some-Description",
  xaddrs: [ "http://[cameraip]/some/path" ]
}
```

Fields are derived from the ONVIF discovery SOAP Envelope.Body.ProbeMatches.ProbeMatch fields, i.e.:

- urn should contain a unique camera id from the EndpointReference.Address field
- name may contain a description such as camera model from the Scopes onvif://www.onvif.org/name Scope
- first xaddrs entry contains the camera ip in its url from the XAddrs field

Implement HTTP (or HTTPS) APIs to support provisioning through config app:

1. PUT http://[cameraip]/provisioning/pair
   Request Headers:
   Authorization: Basic [base64 encoded credentials]
   JSON Request Body:

```
{
  ThingName: "", //name of provisioned thing, to be used as unique clientId
when connecting to iot endpoint and when publishing to topics
  StreamName: "", //name of kvs stream
  Region: "", //aws region
  IoTCertificate: "", //aws iot thing certificate
  IoTPrivateKey: "", //aws iot thing private key
  IoTCACert: "", //ca cert that created the iot thing cert
  IoTEndpointUrl: "", //iot broker endpoint to publish metrics (host to
configure iot sdk)
  IoTCredentialUrl: "", //iot credential provider endpoint to retrieve kinesis
video credentials (full https path)
  KMSKeyId: "" // (optional) kms key id to encrypt stream content
}
```

Example pair request body:

```
{
   "ThingName":"b4cf0b67-ed25-418d-2910-55ac6ce5dadb",
   "StreamName":"b4cf0b67-ed25-418d-2910-55ac6ce5dadb",
   "Region":"us-east-1",
   "IoTCertificate":"-----BEGIN CERTIFICATE-----\nCertContents\n-----END
CERTIFICATE-----\n",
   "IoTPrivateKey":"-----BEGIN RSA PRIVATE KEY-----\nCertContents\n-----END RSA
PRIVATE KEY-----\n",
   "IoTCACert":"-----BEGIN CERTIFICATE-----\nCertContents\n",
   "IoTEndpointUrl":"b11hkcmzrcv4og.iot.us-east-1.amazonaws.com",
   "IoTCredentialUrl":"https://d2odos4oixgocg.credentials.iot.us-east-
1.amazonaws.com/role-aliases/some-role-alias/credentials",
   "KMSKeyId":"arn:aws:kms:region:account:alias/aws/kinesisvideo"
}
```

Response: 200 OK if successfully processed, 400+ if failed.
Response Body on success: not applicable
Response Body on failure:

```
{
  Error: "" //error information.
}
```

2. GET http://[cameraip]/provisioning/status
   Request Headers:
   Authorization: Basic [base64 encoded credentials]
   Request Body: not applicable
   Response: 200 OK

```
Response Body:
{
  Status: "", //one of "UNPAIRED", "PAIRING", "PAIRED", or "ERROR". PAIRING
indicates an intermediate state where the pair API has been invoked but the
camera is not yet streaming to KVS.
  Error: "" //error information in case of ERROR status
}
```

NOTE: the 2nd api (status) is not yet used in the beta config app.

Run the config app and AWS provisioning stack to test camera implementation

- Download and launch config-app for your platform
  - [mac dmg (https://s3.amazonaws.com/aws-connected-camera-beta/config-app-1.0.0.dmg)](https://s3.amazonaws.com/aws-connected-camera-beta/config-app-1.0.0.dmg)
  - [windows exe (https://s3.amazonaws.com/aws-connected-camera-beta/config-app+1.0.0.exe)](https://s3.amazonaws.com/aws-connected-camera-beta/config-app+1.0.0.exe)
  - [linux AppImage (https://s3.amazonaws.com/aws-connected-camera-beta/config-app-1.0.0-x86_64.AppImage)](https://s3.amazonaws.com/aws-connected-camera-beta/config-app-1.0.0-x86_64.AppImage)
- Once running, the config app will request Stack Endpoint and Provisioning Key fields from the provisioning stack.

- Create/login to target AWS account.
- [Launch AWS provisioning stack. (https://console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks/new?stackName=aws-connected-camera-provisioning-stack&templateURL=https://s3.amazonaws.com/aws-connected-camera-beta/provisioning-template.json)](https://console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks/new?stackName=aws-connected-camera-provisioning-stack&templateURL=https://s3.amazonaws.com/aws-connected-camera-beta/provisioning-template.json)
- Click Next through CloudFormation prompts (no additional configuration required).
- Acknowledge IAM roles will be created by CloudFormation and Create the stack.
- Refresh until stack creation completes (it may take a few minutes).
- View the stack outputs. Copy the `StackEndpoint` and `ProvisioningKey` outputs into the config app and click `Save`.
- Once endpoint is successfully configured, the config app will allow camera discovery.
- Click `Discover Cameras`. After a few seconds, a list of discovered cameras will be presented.
    - If cameras are not discovered automatically, you may manually enter an ip address, unique id, and name.
- Select all cameras you wish to provision with the corresponding checkboxes.
- Choose the scheme supported by the camera provisioning API (http or https).
- Enter the camera API username and password if required.
- Click `Provision Selected Cameras`. Config app will coordinate with provisioning stack and camera APIs to provide certificates and other information required for the camera to stream to the configured AWS account.
- View the KVS stream for the provisioned cameras in the AWS account.