Legacy Manufacturing Flow (Pre-SDK1.16)

If you are using Amazon Sidewalk SDK release 1.15 or older, please follow the manufacturing flow and steps described in this document.

Manufacturing Amazon Sidewalk devices for mass production

This section describes how contract manufacturers (CM) can mass manufacture Sidewalk-enabled products. This flow is used when you want to manufacture a large number of devices, such as for bulk production applications. After you manufacture these devices, the control log info can be provided to AWS IoT Core for Amazon Sidewalk for bulk provisioning these devices and onboarding them to AWS IoT. For more information, see Bulk provisioning devices with AWS IoT Core for Amazon Sidewalk in the *AWS IoT Core developer guide*.

This section provides an overview of the different types of provision flows, and the onboarding experience for your Sidewalk devices. It details:

- The criteria that a Sidewalk device must meet so that it can be authenticated to join Amazon Sidewalk.
- The process that must be followed by developers and contract manufacturers (CMs) for provisioning a device to use Amazon Sidewalk, including how to bulk provision Sidewalk-enabled products for manufacturing.
- The life cycle of a Sidewalk device or endpoint in the prototyping and manufacturing process.

Sidewalk device provisioning flows

The following shows the type of flows that can be performed for Amazon Sidewalk devices. This section describes the manufacturing or the mass production flow. For information about the prototyping flow, see Provisioning and registering Sidewalk endpoint for device prototyping.

Prototyping flow

The prototyping flow is used to create a Sidewalk prototype device, and is primarily used for testing purposes. When you use this flow with AWS IoT Core for Amazon Sidewalk, you provision devices one at a time, which will help you learn about the workflow and how to prototype your application. To use this flow, you only need to download the certificate JSON and flash them onto your device.

Mass production flow

The mass production flow, also known as the manufacturing flow, uses a number of factory tools and is designed for manufacturing in large volumes. This flow is performed by contract manufacturers where your devices are manufactured. It combines Factory Diagnostic Firmware, Sidewalk Signing Tool, and Amazon’s Provisioning System to securely issue Sidewalk certificates to devices and register them with Sidewalk cloud services. For more information about accessing the tools, contact Amazon Sidewalk support. The mass production flow can accommodate your operation if you have a large number of Sidewalk devices to onboard.

After the devices are manufactured, the control logs are uploaded to Amazon via SFTP or Electronic Data Interchange (EDI) system, and a CSV file containing the Sidewalk manufacturing serial numbers (SMSN) is uploaded to AWS IoT Core for Sidewalk. The CSV file is used to onboard the devices to the AWS Cloud. After this, your devices are provisioned in the Sidewalk network and they can send or receive data. For more information about onboarding the devices in AWS IoT Core for Amazon Sidewalk, see Bulk provisioning devices with AWS IoT Core for Amazon Sidewalk in the *AWS IoT Core Developer Guide*.

---

(1) Components of Amazon Sidewalk manufacturing

The following section describes the key components of Amazon Sidewalk manufacturing process.

(1.1) Device attestation key (DAK)

A Device Attestation Key acts as a certificate authority for a device type. It is used to endorse the device certificate for Sidewalk device authentication with the Sidewalk network server. For prototype devices, the cloud maintains a Prototype DAK which is used to sign the prototype device certificates. For devices that are manufactured in the Contract Manufacturer (CM), the Production DAK is provided in a Hardware Security Module (HSM).

The DAK is tied to the Sidewalk device profile created with AWS IoT Core for Amazon Sidewalk.

(1.2) Sidewalk certificates

Sidewalk certificate chain

The Sidewalk certificate chain is a collection of certificates which consists of Amazon Root Certificate Authority (CA), multiple intermediate CAs including DAK and leaf certificate which corresponds to Device certificates. It provides a chain of trust to the Amazon Root CA. When manufacturing your devices, the entire public certificate chain from device to root is uploaded during control log ingestion.
See Amazon Sidewalk protocol specification, section 4.1.2 Certificate chain for more details.

Application service key pair

This key pair is unique to each application server. It authenticates the application server with the Sidewalk device. Devices that connect to the same application server use the same application server key pair. The public key is located in the manufacturing data storage on your Sidewalk device.

Sidewalk network server certificate

This certificate is used to authenticate the Sidewalk network server with the device. All Sidewalk devices use the same Sidewalk network server certificate. This certificate is located in the Sidewalk SDK on your device.

(1.3) Hardware Security Module (HSM)

The HSM is a hardened, tamper-resistant hardware device, allowing secure key management. To enable device manufacturing, the Amazon Sidewalk team provisions the DAK certificate, including the DAK private key onto the HSM. HSM is used during the provisioning process at the CM to orchestrate signing of the device certificates without exposing the DAK private key. HSM also includes the full intermediate public certificate chain up to Amazon Root. HSMs can be purchased from the YubiHSM webpage

For more information about starting manufacturing and requesting the HSM provisioning and key, contact Amazon Sidewalk Support.

(1.4) Advertised Product ID (APID)

The `APID` parameter is an alphanumeric string which is needed during manufacturing. APID is located in the manufacturing data storage on your Sidewalk device. After receiving HSM key from Sidewalk, you can obtain APID information from the AWS IoT console, or using the GetDeviceProfile API operation, or the get-device-profile CLI command that's provided by AWS IoT Core for Amazon Sidewalk.

If you already have an APID (by interacting with other Amazon systems), Amazon can link it to your Amazon Sidewalk device profile for pre-production or production purposes. If you don't already have an APID, Amazon will generate one and associate it with the device profile you provide.

NOTE
For prototype devices, the DeviceTypeId must be used instead of the APID. The APID must be used only for production or pre-production devices.

---

(2) Amazon Sidewalk manufacturing setup and workflow

This section describes how Amazon Sidewalk manufacturing works. You'll learn the setup required before taking your devices to factory to production, and how you can use this information to bulk provision your Sidewalk devices.

(2.1) Setup for Amazon Sidewalk manufacturing

To prepare your Sidewalk-enabled products for production, you'll need the following hardware to generate the device key pairs on it. For more information, see Setting up the host.

- A computer, also known as *line PC*, used for provisioning,
- A *device under test (DUT)* to which the line PC is connected to,
- A debugging interface, such as *JLink*, provided with the DUT, and

- The HSM issued by Amazon Sidewalk Team for your Sidewalk products.

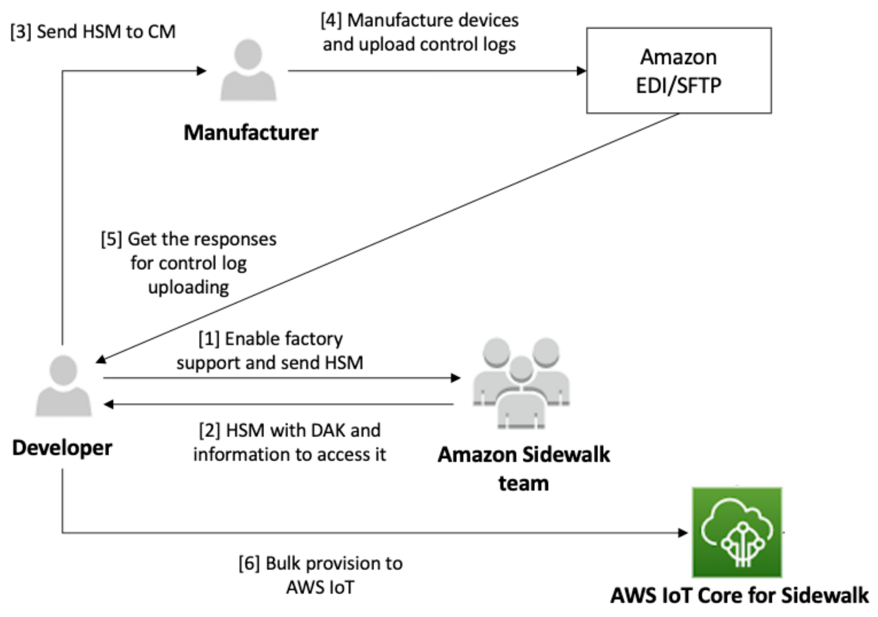
(2.2) Mass production flow for Amazon Sidewalk

The mass production flow follows the steps described below to issue Sidewalk certificates and upload the corresponding record into Sidewalk cloud services. The CM is expected to implement the Line PC script.

1. Line PC script generates the key pairs for the device. There is a key pair for each of the supported elliptic curves.
2. Line PC script passes the generated keys to the Sidewalk Signing Tool to generate the device certificates. The HSM signs the device certificate with the DAK private key.
3. Sidewalk Signing Tool constructs the full Sidewalk Certificate Chain which includes the newly created device certificates and outputs a Sidewalk control log.
4. Line PC script converts the certificate chain to an MFG binary and writes it to the device.
5. Line PC script saves the control logs generated by the Sidewalk Signing Tool in CM's database or internal storage.
6. CM collects and uploads the control logs that contain the Sidewalk Certificate Chain and the device related information via EDI system or SFTP endpoint.

(2.3) How Amazon Sidewalk manufacturing works

This flowchart shows how bulk provisioning works.



[3] Send HSM to CM
[4] Manufacture devices and upload control logs
Amazon EDI/SFTP
Manufacturer
[5] Get the responses for control log uploading
[1] Enable factory support and send HSM
Developer
[2] HSM with DAK and information to access it
Amazon Sidewalk team
[6] Bulk provision to AWS IoT
AWS IoT Core for Sidewalk

The following procedure illustrates the different steps in the manufacturing process.

1. **Create an AWS IoT device profile**
   When your devices are ready for mass production, you can use an existing device profile or Create a new device

profile to enable factory support for your device production.

2. **Enable factory support**
   The Amazon Sidewalk team can link your APID to your device profile. We will update your device profile with the production DAK, and provision it along with the Sidewalk certificate chain onto the HSM you provide, for using AWS IoT Core for Amazon Sidewalk and bulk provisioning your Sidewalk devices. Contact Amazon Sidewalk support for more details and shipping instructions of sending your HSM to the Amazon Sidewalk team for provisioning.

3. **Send HSM to contract manufacturer (CM)**
   After HSM provisioning is complete, we will return the HSM to you, and provide the DAK PIN via encrypted email for your HSM access. You can then send the HSM to your contract manufacturer to enable them to start the manufacturing process for your Sidewalk devices.

4. **Manufacture devices and send control logs and serial numbers**
   The CM manufactures the devices for production, generates control logs, and ingests the logs to into Amazon's device provisioning system via EDI or SFTP. The Amazon's system will process the uploaded log, validate its content and send a notification to the CM.

   The CM also provides you a CSV file that contains a list of manufactured devices and their Sidewalk manufacturing serial numbers (SMSNs). The following code shows a sample control log. It contains the serial number of the device, the APID, and the public device certificates.
   See the Amazon Sidewalk Specification, section 4.3.2 Sidewalk Manufacturing Serial Number (SMSN) for more details.

```
{
   "controlLogs": [
   {
      "version": "4-0-1",
      "device": {
         "serialNumber": "d4c4975ef82728e1e56333b657515a136166afd956aa69745c3efe243660
         "productIdentifier": {
            "advertisedProductId": "abCD"
         },
         "sidewalkData": {
            "SidewalkED25519CertificateChain": "...",
            "SidewalkP256R1CertificateChain": "..."
         }
      }
   }]
}
```

5. **Bulk provision to onboard your Sidewalk devices in AWS IoT**
   You can bulk provision your Sidewalk devices by creating and using import tasks to onboard a large number of your devices to AWS IoT Core for Amazon Sidewalk. Follow steps on Provisioning Sidewalk devices using import tasks.

(3) Sidewalk device lifecycle

The Sidewalk devices that you want to onboard to AWS IoT Core for Amazon Sidewalk can be in either of the following three stages:

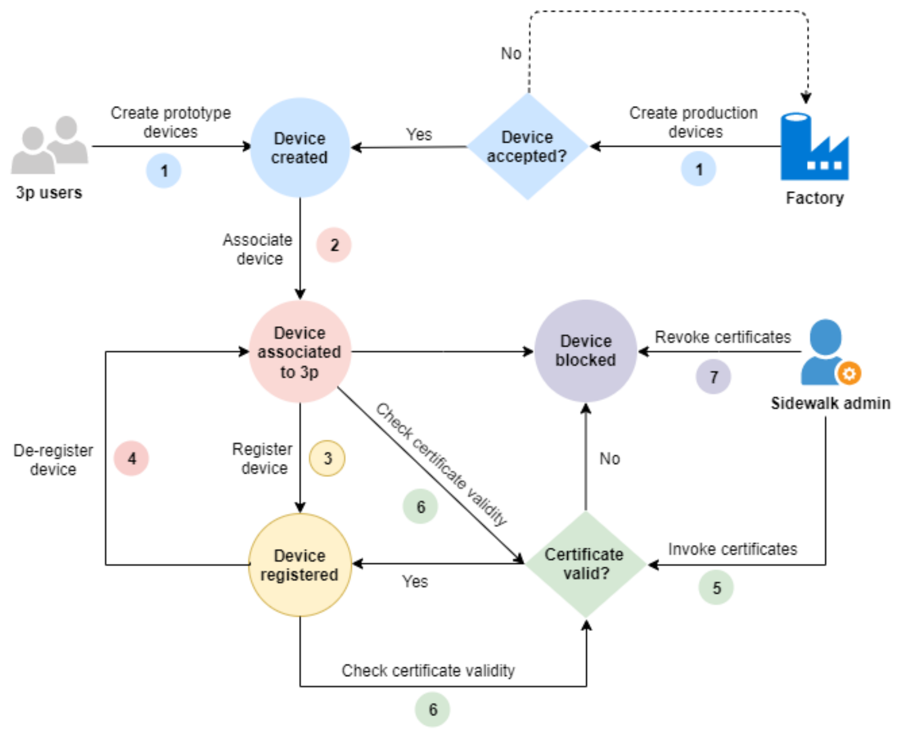
(3.1) Prototype devices

Prototype devices are primarily used for testing purposes and to explore the Amazon Sidewalk onboarding workflow. You can use the prototyping flow to create up to 1,000 Sidewalk prototype devices. You can provision one Sidewalk device at a time since the prototyping flow doesn't support bulk-provisioning to onboard Sidewalk devices.

(3.2) Production devices

Production devices are devices that are manufactured in large quantities without any device limit using the *mass production* flow. These devices can be manufactured after Sidewalk qualification is obtained.

The following diagram shows the various stages in the lifecycle of your Sidewalk prototype, pre-production, or production devices.



Device created
Sidewalk devices can be created either using the prototyping flow or the mass production flow.

- manually flash the manufacturing page (a binary representation of the certificate data from AWS IoT console) onto the

Sidewalk device for authentication.

- In the mass production flow, the contract manufacturer (CM) ingests the device data into Amazon's device provisioning system either via EDI or SFTP, after which the devices are created. If the device data cannot be authenticated, the device entry will be rejected with a notification to the CM to correct the data.

Device associated with account
In this state, after the Sidewalk device is created, it will be associated with an AWS account. The device is now assigned to a Cloud partner for associating the devices with the right developer account. The mapping between device and developer will be owned by the Cloud partner.

Device registered
During device registration, a device presents device certificates to the Sidewalk cloud. If the device is successfully authenticated, then a secure channel is formed between the device and the Sidewalk cloud and then device and application server to establish mutual network and application session keys respectively. These session keys are used for device communication until it is de-registered.

A device can be registered using an Amazon Sidewalk gateway, or a mobile application that integrates the Sidewalk Mobile SDK.

Device de-registered
The de-registration process, also triggered by the end customer, leads to removal of device and end customer association at the cloud partner end. Amazon Sidewalk continues to own the same mapping from creation between the device and cloud partner. De-registration also leads to removal of all the keys established as part of registration process in the cloud and device. The de-registration and registration cycle can occur multiple times in the life-cycle of a device.

Device blocked
The validity of a device on the network is derived from authenticity of the chain of trust. Under unusual circumstances, Sidewalk can revoke certificates for a particular device or device type which would inhibit that device from communicating with the application server. The developer can also revoke established permission between their own device and cloud partner account.

---

(4) How to manufacture and bulk produce Sidewalk devices

This section shows how to mass manufacture your Sidewalk-enabled products for production. You'll learn about the pre-requisites and tools that are used, and the steps that are performed in manufacturing your devices. It also contains information about the steps that you must perform and the steps that must be performed by the contract manufacturer (CM).

(4.1) Required toolkit

To manufacture your Sidewalk devices with AWS IoT Core for Amazon Sidewalk, a collection of tools run on one or more Ubuntu or Windows-based machines. The following tools are required for mass manufacturing :

- **OpenSSL**
  OpenSSL version 1.1.1 or greater. To use OpenSSL with Ed25519 support is required.

- **Sidewalk signing tool**
  The Sidewalk signing tool takes a Sidewalk certificate signing request (CSR) as input for each of the required elliptic curves, and returns a signed certificate chain for the specified EC. To get access to the latest version of the tool, contact Amazon Sidewalk support.

  NOTE
  The signing tool can run on a single machine or distributed in a client-server setup. Running the tool on a single machine can be useful when testing a single line of manufacturing. When you want multiple clients to use a single HSM, you distribute the tool in a client-server setup.

- **YubiHSM connector**
  The YubiHSM connector, *yubihsm-connector*, is a back-end application that's required to communicate with the HSM token.

- **Nginx**
  Nginx is a web server that's required when you're using the client-server setup. This tool serves as a reverse proxy for the YubiHSM connector. It controls access to the yubihsm-connector, such as mutually authenticated TLS providing a secure connection between the components.

- **Provisioning script**
  The provisioning script, `provision.py`, is a manufacturing page generation script. It's used for creating a manufacturing object, which can be flashed into the device memory.

HSMs for Sidewalk signing tool

When manufacturing your Sidewalk devices, each Sidewalk-enabled device that joins the Sidewalk network must be provisioned with a Sidewalk device certificate. The HSM that's issued for use with the Sidewalk signing tool has these two major components.

- The public portions of the device certificates, or the Sidewalk certificate chain. It consists of four certificates up to the Amazon root, and provides a path of valid certificates, or chain of trust, to the Amazon Root Certificate Authority. After the device certificates have been generated, the public device information must be uploaded to Amazon using control log ingestion.
- The private key (DAK), that will be stored in the device.

For more information, see Device attestation key (`DAK`).

Pre-requisites

- Printed circuit board assembly (PCBA) with a supported chipset. When testing HSM-based provisioning, you can use a Sidewalk hardware development kit (HDK).
- HSM (hardware security module) specifically created for your Sidewalk product by Amazon. Amazon can provide

multiple HSMs per product, if needed. Each HSM is identified by a unique serial number, YubiHSM SN, that is printed on the side that does not have the USB contacts. For information about purchasing a YubiHSM, see YubiHSM. The number of YubiHSM needed is based on the mode of operation, as described in Step 1: Setting up the host. For more information, see the YubiHSM key section.

- HSM vendor-provided SDK. For more information, see YubiHSM2 releases.
- Sidewalk device profile that's factory supported and qualified for production. The Sidewalk signing tool requires the APID information from the created profile.
- Computer or native machine running Ubuntu 20.04 or Windows 10.
- Sidewalk signing tool, which corresponds to the Python script `sidewalk-signing-tool.py`. To get access to the latest version of the tool, contact Amazon Sidewalk support.
- Python interpreter, which is required by the signing tool and the provisioning script, `provision.py`.

Manufacturing workflow overview

The following shows you the steps that are involved in mass manufacturing your Sidewalk devices. You'll also learn more about the steps that you need to perform and the steps performed by the contract manufacturer (CM) in the manufacturing workflow.

You only need to create a Sidewalk device profile and obtain factory support so that it's qualified for production use. The CM then sets up the required tools on the host machines, and runs the Sidewalk signing tool to obtain the certificates and generate the control logs. The CM also uploads these logs to Amazon Sidewalk using EDI or an SFTP endpoint, after which you receive a CSV file. You'll then upload this CSV file to an S3 bucket and use AWS IoT Core for Amazon Sidewalk to bulk provision your Sidewalk devices.

For more information, see How Amazon Sidewalk manufacturing works.

Step 1: Setting up the host
The contract manufacturer (CM) sets up the required tools on the host machines for manufacturing your Sidewalk devices. You don't need to take any action.

Step 2: Provisioning with CSR and Sidewalk Signing tool
You create a prototype device profile and request the Amazon Sidewalk team to provide factory support. The CM then generates the CSRs, runs the YubiHSM connector in the background, and uses the Sidewalk signing tool to return the encoded and signed Sidewalk certificate chains.

Step 3: Constructing and uploading Sidewalk control logs
The CM then uses the signing tools to generate and consolidate the control logs, which is then uploaded to Amazon Sidewalk using EDI or an SFTP point. You'll then receive an email with a CSV file attached that contains the status information.

You can now upload the CSV file received to an S3 bucket and provide the information to AWS IoT Core for Amazon Sidewalk for bulk provisioning. If AWS IoT Core for Amazon Sidewalk finds a match in the serial numbers between the CSV file and the control logs that it receives from Amazon Sidewalk, the corresponding devices are then provisioned.

The following sections describe each of these steps in additional detail.

(4.1) Setting up the host

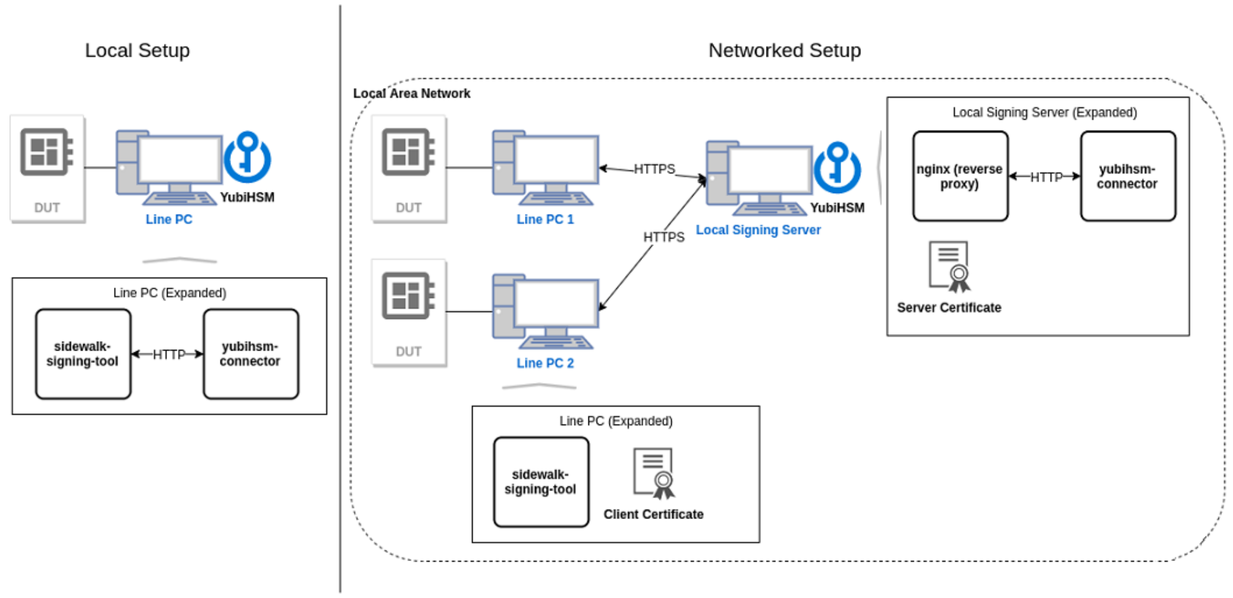
This section shows how the CM must set up the host for mass manufacturing Sidewalk devices. See Required toolkit and Pre-requisites.

Important
The steps in this section need to be performed by the CM who will manufacture your Sidewalk devices for mass production, using the Sidewalk device CLI and the Sidewalk signing tool.

The following steps show you how to set up your host and install the required tools.

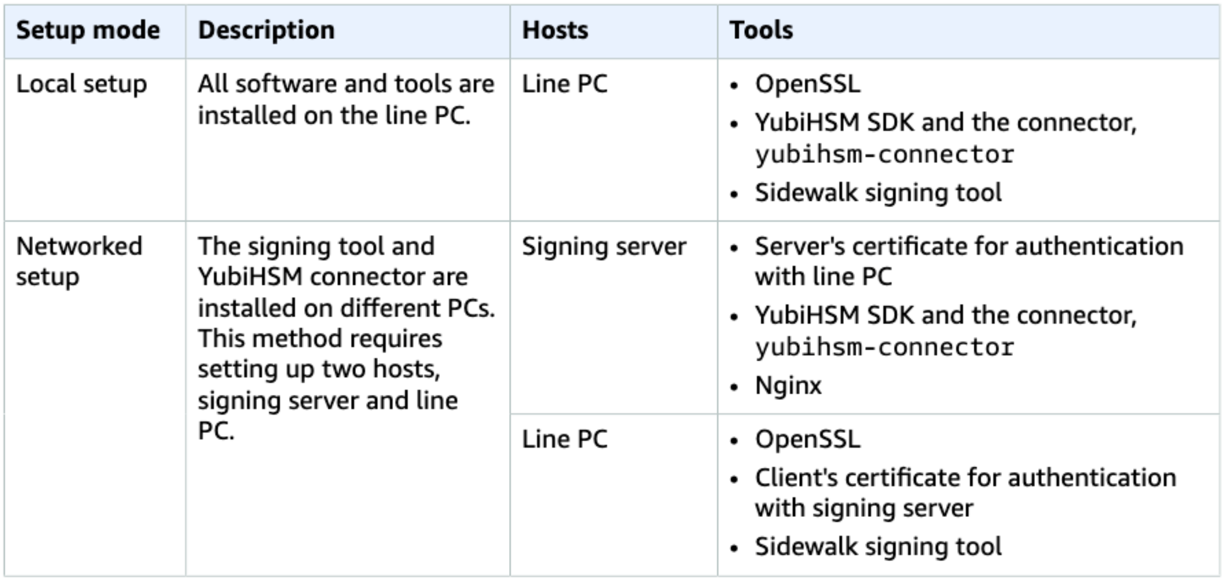
STEP 1: CHOOSE MODE OF SETUP

Depending on how you want to manufacture your devices, you can use a local or a networked setup. A local setup has both the signing tool and the YubiHSM connector running on the same machine. A networked setup moves the connector component into a server PC that is accessible through the local area network. If an HSM attached to a the signing server can support multiple line PCs, we recommend using a networked setup. Using a local setup requires every line PC to be associated with an HSM.



This table shows more about the two modes of setup and the tools that must be installed on the hosts.

Local and networked setup



Setup mode	Description	Hosts	Tools
Local setup	All software and tools are installed on the line PC.	Line PC	• OpenSSL • YubiHSM SDK and the connector, `yubihsm-connector` • Sidewalk signing tool
Networked setup	The signing tool and YubiHSM connector are installed on different PCs. This method requires setting up two hosts, signing server and line PC.	Signing server	• Server's certificate for authentication with line PC • YubiHSM SDK and the connector, `yubihsm-connector` • Nginx
		Line PC	• OpenSSL • Client's certificate for authentication with signing server • Sidewalk signing tool

---

STEP 2: INSTALL YUBIHSM SDK

Visit the YubiHSM2 libraries and tools page for additional information and support on HSM. You can see a list of all available downloads ordered by version, starting with the most recent version on the YubiHSM2 releases page. Select to download the SDK for your Windows or Ubuntu system environment, and follow the YubiHSM2 Usage Guide for installing the YubiHSM 2 tools and software.

STEP 3: INSTALL OPENSSL

To use the line PC, you must have OpenSSL installed. On a Ubuntu machine, OpenSSL is available with the default installation. For more information, see OpenSSL. On a Windows machine, you can install OpenSSL from the OpenSSL binaries page.

STEP 4: INSTALL SIDEWALK SIGNING TOOL

The Sidewalk signing tool accepts and takes the CSR, and outputs the encoded signed Sidewalk certificate chain. To use the Sidewalk signing tool.

1. First contact the Amazon Sidewalk team to get the latest version of the signing tool, `sidewalk-signing-tool.py`.
2. After you receive the tool, install it by following the instructions in the README.md document.
3. To install the required Python dependencies for using the signing tool, such as the Python interpreter, run the requirements file, `requirements.txt` contained in the signing tool. It will then install the required Python dependencies using pip.

```
pip3 install -r requirements.txt
```

STEP 5: GENERATE CERTIFICATES FOR MUTUAL AUTHENTICATION (NETWORKED SETUP)

NOTE
This step is required only if you're using the networked setup.

When using a networked setup, the HSM and the Sidewalk signing tool are located in different machines. To secure traffic over the network and use a secure connection between the two machines, we recommend that you use mutually authenticated TLS for the client and server to authenticate each other.

1. **Generate a local signing authority**
   Generate a local signing authority that will be used to endorse both the client and server certificates. When you run this command, you'll be prompted to enter a passphrase, and any phrase can be used.

```
openssl req -new -x509 -subj "/CN=Sidewalk CA/" -keyout ca.key -out ca.pem -days 365
```

You must securely store the key, `ca.key`. It must not be visible to the line PC or the signing server.

2. **Generate server certificate**
   Generate the server certificate that the server will present to the Sidewalk signing tool when setting up a connection. The certificate is signed by the Local Signing Authority. To generate the server certificate.

a. First create a V3 extension file, `v3_server.ext`.

```
authorityKeyIdentifier=keyid,issuer
basicConstraints=CA:FALSE
keyUsage = digitalSignature, nonRepudiation, keyEncipherment, dataEncipherment
extendedKeyUsage = serverAuth
```

b. Then create the server certificate.

```
# Create the Certificate Signing Request
openssl req -new -newkey rsa:2048 -sha256 -nodes \
-keyout server.key -out server.csr

# Generate the certificate signed by the CA
openssl x509 -req -days 365 -CA ca.pem -CAkey ca.key -CAcreateserial \
-in server.csr -sha256 -out server.pem -extfile v3_server.ext

# Enter the passphase provided when generating the local signing authority
# Enter the IP/domain address of the server when asked for the CN
```

3. **Generate client certificate**
   Generate the client certificate for the signing tool that will be presented to the server during authentication. You create the client certificate similar to how you created the server certificate.

a. First create a V3 extension file, `v3_client.ext`

```
authorityKeyIdentifier=keyid,issuer
basicConstraints=CA:FALSE
keyUsage = digitalSignature, nonRepudiation, keyEncipherment, dataEncipherment
extendedKeyUsage = clientAuth
```

b. Then create the client certificate.

```
# create a certificate signing request
openssl req -new -newkey rsa:2048 -sha256 -nodes -keyout client.key -out client.csr

# create the client certificates signed with the Certificate Authority
openssl x509 -req -days 365 -CA ca.pem -CAkey ca.key \
    -CAcreateserial -in client.csr -sha256 -out client.pem -extfile v3_client.ext
```

STEP 6: INSTALL NGINX ON SIGNING SERVER (NETWORKED SETUP)

NOTE
This step is required only if you're using the networked setup.

Nginx is a web server that you install on the signing server for mutual authentication between the signing server and line PC. To install and configure Nginx, perform the following steps.

1. **Install Nginx**
   Go to the Nginx website and install the tool by following the platform-specific instructions as described in the Installing Nginx open source documentation.

   [Windows]
   On a Windows machine, you can download the prebuilt Nginx binary from the Nginx for Windows web page. Unzip the binary file into the `C:\nginx` folder and then run it to install Nginx.

   [Ubuntu]
   On an Ubuntu machine, you can install Nginx using `apt`.

```
sudo apt install -y nginx
```

2. **Configure Nginx**
   Configure Nginx depending on the platform that you're running it from.

   [Windows]
   On a Windows machine, edit the configuration file at `c:\nginx\conf\nginx.conf` directly and replace the section `server` with the following configuration similar to the Ubuntu section. In this example:

   - `log_path` is the path where the Nginx log is stored, such as `C:/nginx`.
   - `cert_path` is the path where the certificate is stored.

```
server
{
   listen 8081 ssl;
```

```
    listen [::]:8081 ssl;

    access_log C:/nginx/reverse-access.log;
    error_log C:/nginx/reverse-error.log;

    ssl_certificate <path>/server.pem;
    ssl_certificate_key <path>/server.key;

    ssl_client_certificate <path>/ca.pem;
    ssl_verify_client on;
    ssl_verify_depth 1;

    location / {
       if ($ssl_client_verify != SUCCESS) {
          return 403;
       }
       proxy_pass http://127.0.0.1:12345;
    }
}
```

[Linux]
On an Ubuntu machine, perform the following steps to configure and enable Nginx.

a. Create a configuration file, `/etc/nginx/sites-available/reverse-proxy.conf`, with these contents for the server configuration. In this example:

- `log_path` is the path where the Nginx log is stored, such as `/var/log/nginx`.
- `cert_path` is the path where the certificate is stored.

```
server
{
    listen 8081 ssl;
    listen [::]:8081 ssl;

    access_log /var/log/nginx/reverse-access.log;
    error_log /var/log/nginx/reverse-error.log;

    ssl_certificate <path>/server.pem;
    ssl_certificate_key <path>/server.key;

    ssl_client_certificate <path>/ca.pem;
    ssl_verify_client on;
    ssl_verify_depth 1;

    location / {
        if ($ssl_client_verify != SUCCESS) {
            return 403;
        }
        proxy_pass http://127.0.0.1:12345;
    }
}
```

b. Enable the `config` by running the following command.

```
sudo ln -s /etc/nginx/sites-available/reverse-proxy.conf /etc/nginx/sites-enabled/reve
```

3. **Enable Nginx**
   Run the following platform-specific commands to enable Nginx.

[Windows]
On a Windows machine, run this command to enable Nginx.

```
C:\nginx>start nginx
```

[Ubuntu]
On an Ubuntu machine, run this command to restart and run Nginx.

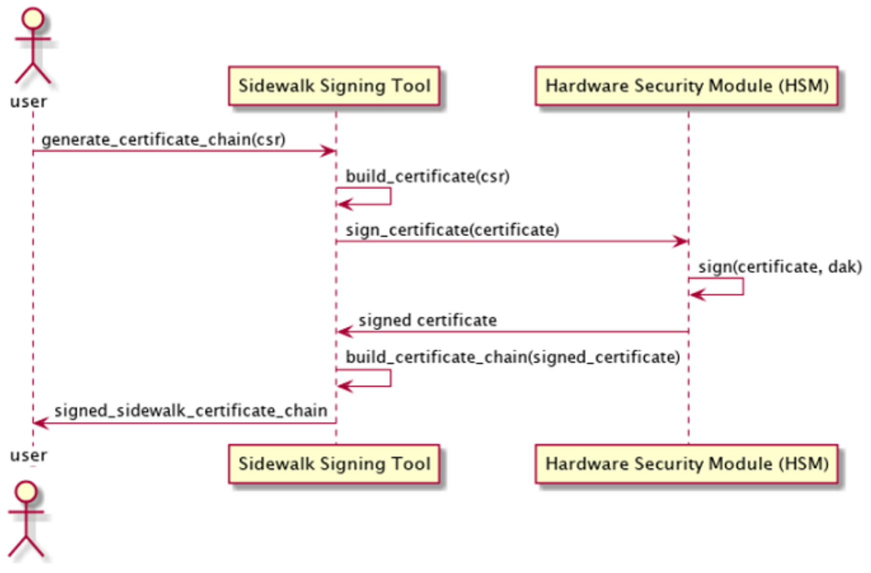
```
sudo systemctl restart nginx
```

(4.3) Provisioning with CSR and Sidewalk Signing tool

Each device being manufactured needs to have a unique device certificate. The Sidewalk Signing Tool running on the Line PC will take PEM key files to generate a Sidewalk certificate and signs it using the Device Attestation Key (DAK) securely stored in the provided Hardware Security Module (HSM). The following steps are involved in provisioning for the manufacturing workflow.

STEP 1: GENERATE CSRS FOR SIDEWALK SIGNING TOOL

Before your Sidewalk signing tool can sign the certificates, generate the required CSRs for the tool as described in the following steps.


user
Sidewalk Signing Tool
Hardware Security Module (HSM)
generate_certificate_chain(csr)
build_certificate(csr)
sign_certificate(certificate)
sign(certificate, dak)
signed certificate
build_certificate_chain(signed_certificate)
signed_sidewalk_certificate_chain
user
Sidewalk Signing Tool
Hardware Security Module (HSM)

Generate the NIST-P256 and ED25519 device keys by using the following commands:

1. **Create the P256R1 device private key.**

```
# Create the private key
openssl ecparam -name prime256v1 -genkey -out p256.priv
```

2. **Create ED25519 device private key.**

```
# Create the private key
openssl genpkey -algorithm ed25519 -out ed25519.priv
```

STEP 2: RUN THE YUBIHSM CONNECTOR IN THE BACKGROUND

NOTE
If you're using a Windows machine, the YubiHSM connector runs automatically as a service in the background. You don't have to manually start the connector.

On the machine that the HSM is attached to, run the YubiHSM connector in the background. If you're using the networked setup, the HSM is on the signing server. If you're using the local setup, run the connector on the line PC. For more information, see Step 1: Choose mode of setup.

On a Linux or Ubuntu machine, you run the connector using the `yubihsm-connector` command without any outputs.

```
# yubihsm-connector is running without any outputs here.
yubihsm-connector
```

STEP 3: RUN THE SIDEWALK SIGNING TOOL

The following shows the parameters that the Sidewalk signing tool can accept, some usage examples, and the output of running the tool.

Sidewalk signing tool parameters

The following table shows the parameters that the Sidewalk signing tool can accept and their description. For more information about the signing tool parameters, run the following command.

```
sidewalk-signing-tool --help
```

Signing tool parameters

	Parameter	Description
1	--product	The product tag identifying which DAK in HSM is used for the signing operation. This tag will be provided by Amazon for some specific product and usually starts with prefix ‘PREPROD_DAK’ or ‘RNET_DAK’.
2	--pin	The PIN code to access the HSM. Provided by Amazon along with the product tag.
3	--ed25519_private_key_file and —p256r1_private_key_file	The paths to PEM files for device private keys. The Sidewalk Signing Tool will extract the private/public keys from these files and generate CSRs for signing automatically.
4	--device_profile_json (AWS Flow only)	The JSON file downloaded from AWS for the device profile, from which the Sidewalk Signing Tool get the app server key and put it to the output file so provision.py can take use of it. This is only for AWS flow; The main reason of having this input is to pass the application server public key from the device profile to provision.py. In ACS flow the public key will be provided explicitly with the command line argument of provision.py.
5	-o	The filename of the output from the Sidewalk Signing Tool, which contains the device certificate and Sidewalk certificate chains that will be used as input of provision.py
6	--connector	The URL for the singing server, including the information for protocol (http or https), IP\/domain name, and the port number. If you are using local installation, it should be the URL to the local yubihsm-connector, which is normally “http://localhost:12345”
7	--generate_smsn	Ask the Signing Tools to generate the SMSN based on the information provided with --device_type\/--dsn\/\–apid. Please always use this argument.
8	--ca_cert, --client_cert and —client_key	These 3 arguments are mandatory for establishing HTTPS channel with the signing server.

General usage example

The following command shows an example of how you run the tool. In this example:

- The arguments `ca_cert`, `client_cert`, and `client_key` are only required when you use the networked installation since it requires authentication between the line PCs and the signing server.
- The argument `--generate_smsn` automatically generates and includes the Sidewalk manufacturing serial number (SMSN).

```
# Usage Example
python3 sidewalk_signing_tool.py --product=<model label> --pin=<hsm pin> \
--ed25519_private_key_file=<ed25519_private_key_file> --p256r1_private_key_file=<p256r1_key_file> \
--connector=<connector url> --generate_smsn --device_type=<device_type> --dsn=<device serial number> \
--apid=<advertised product id> --device_profile_json=<device_profile.json> \
-o out.json [--ca_cert=<CA for signing server> --client_cert=<client certificate> --client_key=<client key>]
```

Local installation example
The following command shows an example of how you run the tool when using local installation. In this example, the tool runs on the line PC.

> # Example for local installation:
> python3 sidewalk_signing_tool.py --product=RNET_DAK_PROJNAME --pin=password \
> --ed25519_private_key_file=ed25519.pem --p256r1_private_key_file=p256r1.pem \
> --connector=http://localhost:12345 --generate_smsn --device_type=Test --dsn=test --apid=Test \
> --device_profile_json=device_profile.json

Networked installation example
The following command shows an example of how you run the tool when using local installation. In this example, the tool runs on the line PC and the YubiHSM connector runs on the signing server so a secure connection is required between the two machines.

> # Example for networked installation where the signing server is at 192.168.10.100.
> # In the signing tool, the ca_cert, client_cert and client_key options
> # can be used to set the certificate and specify the expected certificate authority.
> python3 sidewalk_signing_tool.py --product=RNET_DAK_PROJNAME --pin=password \
> --ed25519_private_key_file=ed25519.pem --p256r1_private_key_file=p256r1.pem \
> --connector=https://192.168.10.100:8081 --generate_smsn --device_type=TEST --dsn=test --apid=TEST \
> --ca_cert ca.pem --client_cert client.pem --client_key client.key --device_profile_json=device_profile.json

Sidewalk signing tool output
After you run the signing tool using the commands listed below, if the tool runs successfully, the Sidewalk certificate chains will be returned as output. The following code shows a sample output.

```
{
    "p256R1": "JNTwXmILBd2ODPSSedYVApPAGe3L106QZ0O1h7wfHTUS2bl5RhLFmNv2wONst3AJrP4EeF
    "eD25519": "JNTwXmILBd2ODPSSedYVApPAGe3L106QZ0O1h7wfHTUKYxUkbsfdzOECzOBCSTKyCoqF2
    "metadata": {
        "smsn" : "24D4F05E620B05DD8E0CF49279D6150293C019EDCBD74E906743B587BC1F1D35",
        "apid" : "CfnC"
    }
}
```

Include private keys in output
The output of the JSON files does not include the private keys, which are required for a device to join the Sidewalk network. To include this information into the JSON file, specify the arguments `--p256r1_private_key` and `--ed25519_private_key` when using the Sidewalk signing tool. The tool will then insert the fields `devicePrivKeyEd25519` and `devicePrivKeyP256R1` into the JSON file that contains this information. For more information about the parameters, see Sidewalk signing tool parameters.

The following code shows a sample signing tool output with the `--ed25519_private_key 5041b68494e5ead77df088c245d2fa618f71e84a8f23494752e6547acf8bdd63`, and `--p256r1_private_key ec4df3bee946213636f3478ce334415c380f75c0a008afd5bbe001e09826a874`.

```
{
    "p256R1": "JNTwXmILBd2ODPSSedYVApPAGe3L106QZ0O1h7wfHTUS2bl5RhLFmNv2wONst3AJrP4EeF
    "eD25519": "JNTwXmILBd2ODPSSedYVApPAGe3L106QZ0O1h7wfHTUKYxUkbsfdzOECzOBCSTKyCoqF2
    "metadata": {
        "smsn" : "24D4F05E620B05DD8E0CF49279D6150293C019EDCBD74E906743B587BC1F1D35",
        "apid" : "GZBd",
        "devicePrivKeyEd25519": "5041b68494e5ead77df088c245d2fa618f71e84a8f23494752e65
        "devicePrivKeyP256R1": "ec4df3bee946213636f3478ce334415c380f75c0a008afd5bbe001
    }
}
```

(4.3) Constructing and uploading Sidewalk control logs

CM follows these steps to construct the control logs from the output of the Sidewalk signing tool with some additional information. They can consolidate the logs if there are multiple devices, and then upload them to the Amazon Sidewalk system for provisioning your Sidewalk devices.

STEP 1: CONSTRUCT CONTROL LOGS

The signing tool can generate the control logs file using a format that can be ingested into Amazon's device provisioning system via EDI or SFTP endpoints.

To generate the control logs and specify the directory where the logs will be generated, use the parameter `--control_log_dir` when running the Sidewalk signing tool. For more information and how to run the tool, see Sidewalk signing tool parameters

The signing tool will generate a Sidewalk control log file for the device in the specified directory, with a name `C_CONTROL_LOG_<date and time in yyyyMMddhhmmss format>.txt`. The control logs generated will use a format that complies with the version `4-0-1` and contains the required device information to be uploaded to Amazon Sidewalk, such as the SMSN, APID, and the Sidewalk certificate chains. For more information, see the **control logs specification** section.

In this code, the `serialNumber` and the certificate chains `sidewalkED25519CertificateChain` and `sidewalkP256R1CertificateChain` are obtained by running the Sidewalk signing tool. You obtain the APID when creating a Sidewalk profile.

```
{
  "controlLogs" : [
    {
      "version" : "4-0-1",
      "device" : {
        "serialNumber": "device1SN",
        "productIdentifier": {
          "advertisedProductId": "GZBd"
        },
        "sidewalkData": {
          "sidewalkED25519CertificateChain": "ZfZFVIghs+3EJrr...qRB+Aw==",
          "sidewalkP256R1CertificateChain": "ZfZFVIghs+3EJrr...BZ1Bw==",
          "label": "PRODUCTION / PREPRODUCTION"
        }
      }
    }
  ]
}
```

STEP 2: CONSOLIDATE CONTROL LOGS

The control log file generated by the Sidewalk signing tool contains the control log information only for a single Sidewalk device as the certificates are signed only for a single device at a time. If you have multiple Sidewalk devices, their control logs can be consolidated into a single control log file for control log ingestion. To consolidate the control logs, use the tool `consolidate_cl.py` provided by the Sidewalk signing tool.

For example, the following command shows how to run this tool. You can move all the control logs to be consolidated into a single directory and then run the tool from that directory.

```
python3 consolidate_cl.py /tmp/cl/C_CONTROL_LOG_*.txt
```

In this example, the command reads all control log files that are in the `/tmp/cl` directory. After the tool runs successfully, it generates a new control log file, for example `C_CONTROL_LOG_20221021155511.txt`, that will contain the control log content of its inputs in the directory.

```
Processing /tmp/cl/C_CONTROL_LOG_20221018122452.txt
Processing /tmp/cl/C_CONTROL_LOG_20221018141057.txt
Processing /tmp/cl/C_CONTROL_LOG_20221018141105.txt
Processing /tmp/cl/C_CONTROL_LOG_20221018141106.txt
Processing /tmp/cl/C_CONTROL_LOG_20221018141108.txt
Processing /tmp/cl/C_CONTROL_LOG_20221018141109.txt
Processing /tmp/cl/C_CONTROL_LOG_20221018141110.txt

...


C_CONTROL_LOG_20221021155511.txt
```

The consolidated control log file will contain the required information for multiple devices. The following code shows a sample control log file.

```
{
  "controlLogs" : [
    {
      "version" : "4-0-1",
      "device" : {
        "serialNumber": "device1SN",
         "productIdentifier": {
          "advertisedProductId": "abCD"
        },
        "sidewalkData": {
          "sidewalkED25519CertificateChain": "ZfZFVIghs+3EJrr...qRB+Aw==",
          "sidewalkP256R1CertificateChain": "ZfZFVIghs+3EJrr...BZ1Bw==",
          "label": "PRODUCTION / PREPRODUCTION"
        }
      }
    },
    {
      "version" : "4-0-1",
      "device" : {
        "serialNumber": "device2SN",
        "productIdentifier": {
          "advertisedProductId": "CfnC"
        },
        "sidewalkData": {
          "sidewalkED25519CertificateChain": "3OJknQsyH949Ism...qRB+Aw==",
          "sidewalkP256R1CertificateChain": "3OJknQsyH949Ism...BZ1Bw==",
          "label": "PRODUCTION / PREPRODUCTION"
        }
      }
    }
  ]
}
```

STEP 3: UPLOAD CONTROL LOGS

After you've provisioned your Sidewalk device, you must upload the control logs file to Amazon Sidewalk that includes information about the provisioned device. This information includes the device identifier, APID, SMSN, and Sidewalk certificate chain.

To upload your control logs, use either of the following approaches:

Upload control logs using EDI

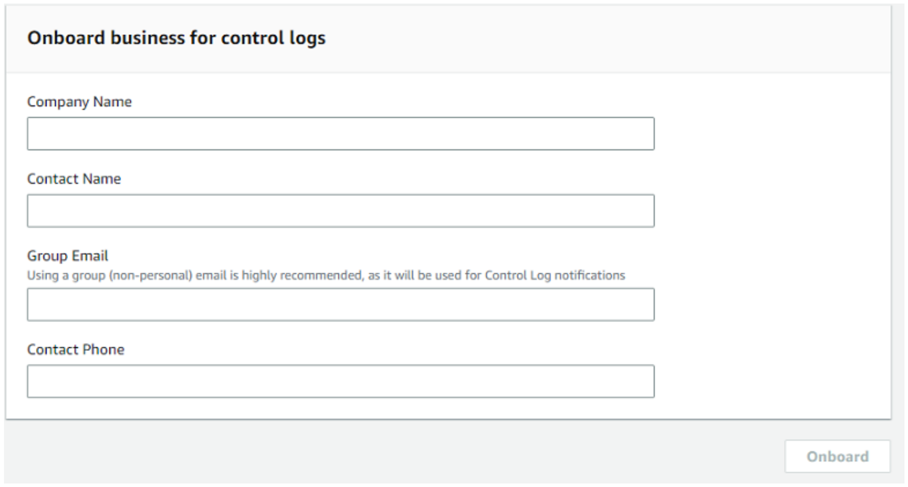
After a contract manufacturer (CM) has been issued a YubiHSM key, the CM must set up the factory line to provision devices using the Sidewalk certificates. For your Sidewalk devices to connect to the cloud and use other AWS services, the control logs must be uploaded to Amazon.

For information about uploading the control logs using Amazon's electronic data interchange (EDI) system, contact Amazon Sidewalk support.

Upload control logs using SFTP endpoint

To upload the control logs using an SFTP endpoint, perform the following steps.

1. Sign in to your developer account and go to the Frustration-Free Setup (FFS) developer console.
2. Go to the **Control Logs** section of the FFS developer console and choose **Manage Control Logs**.
3. Enter information about the business you want to onboard, which includes the **Company Name**, **Contact name**, **Group Email**, and **Contact phone**. Choose **Onboard**.


Onboard business for control logs

Company Name

Contact Name

Group Email
Using a group (non-personal) email is highly recommended, as it will be used for Control Log notifications

Contact Phone

4. Generate a secure RSA key and upload the public key on the portal.

   a. To generate the RSA key, run the following command.

      a. To generate the RSA key, run the following command.

```
ssh-keygen -t rsa -b 2048 -m PEM
```

   b. Enter the file name "`control_log_key`" in which to save the key and the passphrase when prompted. Running this command generates two files `--control_log_key` and `--control_log_key.pub`.

   c. Upload the public key `--control_log_key.pub` on the portal by choosing **Choose file** and then choose **Create**. A pair of SFTP endpoints will be generated for sending the control logs and for receiving feedback.

   NOTE
   The **Upload endpoint** is for sending control log to Amazon, and the **Feedback endpoint** is for receiving responses from Amazon about the uploaded control log.

5. Choose the control log file that you want to upload. This example uses the control log file that was created in Step 2: Consolidate control logs.

```
{
  "controlLogs": [
    {
      "version": "4-0-1",
      "device": {
        "serialNumber": "418A07E3811B8CED614BD27BD2445FAE50A7376A3EB9993CA2017F497A87A
        "productIdentifier": {
          "advertisedProductId": "vLpm"
        },
          "sidewalkData": {
            "sidewalkP256R1CertificateChain": "QYoH44EbjO1hS9J70kRfrlCnN2o+uZk8ogF/SXqHp
            "sidewalkED25519CertificateChain": "QYoH44EbjO1hS9J70kRfrlCnN2o+uZk8ogF/SXqH
          }
        }
      },

      {},
      ...
    ]
  }
```

6. To upload the control logs, perform the following steps.

   a. First start the SFTP endpoint.

```
sftp --oIdentityFile={path_to_key_file}/control_log_key \
    --oHostKeyAlgorithms=+ssh-dss sftp://{upload_endpoint}/To_Amazon
```

   b. Next upload the control log file to the endpoint.

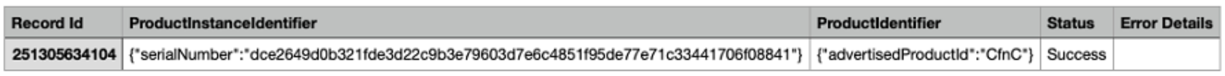
```
# Display local directory listing.
sftp> lls

# Upload control log file name.
sftp> put <local_path>/<control_log_filename>

# Exit out of the SFTP connection.
sftp> exit
```

   It takes few minutes for the control log to be processed and feedback to be received. Once completed, an email will be sent to the email address provided in the onboarding phase in step 3 where you specified the onboarding business information for control logs.

Next steps

Now that the control log has been sent via email, you can use AWS IoT Core for Amazon Sidewalk to provision these devices to AWS IoT in bulk.

The user can open the email received and download the CSV file attached to the email. The CSV file summarizes the control log upload status, as shown below. This file will be used to provision the Sidewalk devices in bulk using AWS IoT Core for Amazon Sidewalk. For more information, see Bulk provisioning devices with AWS IoT Core for Amazon Sidewalk in the *AWS IoT Core developer guide*.



Record Id	ProductInstanceIdentifier	ProductIdentifier	Status	Error Details
251305634104	{"serialNumber":"dce2649d0b321fde3d22c9b3e79603d7e6c4851f95de77e71c33441706f08841"}	{"advertisedProductId":"CfnC"}	Success	