The Intelligent Cloud Connect smart terminal enables you to simply, easily and securely connect your thing to Amazon Web Services (AWS) from anywhere in the world over cellular networks. This enables you to remotely extract data from your thing for a variety of industrial and commercial applications, such as metering, monitoring, transportation, security, and so on.

Getting Started
Before you can use the Intelligent Cloud Connect smart terminal, refer to the AnyNet IRIS Quick Start Guide to complete the following steps:

1. Create an Amazon Web Services (AWS) account.
2. Subscribe to AnyNet Cellular Connectivity for AWS IoT in the AWS Marketplace.
3. Create a mandatory, dedicated AWS IAM user with specific permissions.
4. Install and configure AnyNet IRIS.

Creating a thing in AWS IoT Core

**Do not create things until you have verified the advisory and alert email addresses that you supplied during AnyNet IRIS configuration. Receiving a verification link may take up to 30 minutes. If you have not received an email, contact Eseye Support: support@eseye.com.**

**Before you begin**
- Ensure that the AnyNet Secure Cellular Connectivity configuration process is complete on your AWS account.
- Eseye connects your thing to AWS over a cellular network. To purchase the requisite SIM cards, search for Eseye SIM on Amazon.com.
- You need a AnyNet Secure SIM number, which is the unique serial number printed on the back of the SIM card.

**To create a thing in AWS IoT Core:**

1. Sign in to the AWS Management Console: [aws.amazon.com/console](http://aws.amazon.com/console)
2. In the AWS Services section, find the IoT Core service.
3. If this is your first time to sign into the AWS Management Console, select Get Started.

4. In the left-hand AWS IoT menu, select Manage > Things.

5. In the top right corner, select the correct AWS Region you want to use.

6. If this is your first time creating things, select Register a thing, otherwise select Create.
   The Creating AWS IoT things page appears.
7. Select Create a single thing.
   The Add your device to the thing registry page appears.
8. Type a Name for your thing.
9. In the Thing Type drop-down box, select AnyNetThingType.

10. If required, add the thing to a group.
11. In the Set searchable thing attributes section, leave the ActionRequest Value field blank.
12. Leave the PolicySelector Value field blank.
13. In the SimId Value field, type the SIM number (described above).

14. Select Next.
   The Add a certificate for your thing page appears. Eseye manages the certificate process so you don’t have to.
15. Select Create thing without certificate.

   When you use AWS IoT in your AWS account for the first time, an AWS error may appear because Device Gateway endpoint provisioning is not complete on your account. If this occurs, allow AWS services 5-10 minutes to complete, and try again.

A Successfully created thing message appears, and the new thing appears on the Things page.
Establishing a cellular connection
Ensure you have connected the terminal antenna to the Antenna connection.

To power on and establish your cellular connection:
1. Fully insert the SIM card into the Intelligent Cloud Connect.
   You can see how to insert the SIM on the underside of the terminal. The socket is a push-to-insert, push-to-release type socket. When the SIM is properly inserted, it clicks into place and sits flush with the edge of the SIM slot.

2. Turn on the Intelligent Cloud Connect.
The Intelligent Cloud Connect connects to a cellular network using the Cinterion® PLS62-W Wireless Module, and is now ready to receive security and identity information from AWS.

Provisioning the Cinterion® PLS62-W Wireless Module
Eseye automatically provisions the AnyNet Secure SIM. During this process, the security and identity information is downloaded and programmed into the SIM card. You can observe the provisioning progress in the device shadow. You can access the shadow using Lambda functions, programatically, or through the AWS IoT Console.

- Using the AWS IoT Console, select the thing you created using the matching SIM number, then select Shadows > Classic Shadow to view the Shadow Document.

Use the Shadow state pane to view the certificate delivery progress. The certificate is delivered after the status changes from Pending to Provisioned. You can also view smart terminal message consumption and location information.

Provisioning normally takes 5-10 minutes to complete, although the process may take up to an hour. If your Cinterion® PLS62-W Wireless Module has not connected in 24 hours, contact Support: support@eseeye.com.

During provisioning, the Cinterion® PLS62-W Wireless Module will reset four times. Use a terminal emulator to observe this process.

For more information about the LEDs, refer to the Thales AT Command documentation.

Sending data from your thing to the cloud
Before you begin
- Using the terminal emulator, send an AT<return> command to ensure that the smart terminal can receive AT commands.
- Send an AT+ETMINFO=version<return> command to verify that the version number is 2.0.20 or higher.
- Ensure you know the thing name that you set up in AWS.

To test that your thing can publish information to AWS:
1. Create two publish topics in the Cinterion® PLS62-W Wireless Module.
   - Using a terminal emulator, type:
     ```
     at+emqpubopen=0,"PublishToCloud0"
     ```
     ```
     at+emqpubopen=1,"PublishToCloud1"
     ```
   - Check that the first two index numbers are assigned a topic each. Type:
     ```
     at+emqpubopen?<return>
     ```

A list of index numbers and their assigned topics appears.

2. Subscribe to the newly created publish topics in AWS.
a. Using AWS IoT, in the left hand menu, select Test.
b. In the Subscriptions panel, select Subscribe to a topic.

c. In the Subscription topic box, type: 
\(+/<\text{ThingName}>\)

This subscribes AWS to all topics related to your thing.
d. Select Subscribe to topic.

A subscription appears listed in the Subscriptions panel.

3. Publish information to the topics you created in the Cinterion® PLS62-W Wireless Module.

You can send a maximum payload of 1000 characters to AWS.

a. Using the terminal emulator, type:
\(\text{at+emqpublish=0,1,}\"\text{\{"Temperature": 24\}\"}<\text{return}>\)
\(\text{at+emqpublish=1,1,}\"\text{\{"BatteryPower": \"Low\\}\"}<\text{return}>\)

The emqpublish command uses the following syntax:
\(\text{at+emqpublish=IndexNumber, QoS, }\"\{\text{PublishDataInJSON}\}\"\)

These messages instantly appear in AWS.

4. View the published information in AWS.

a. In the Subscriptions panel, select \(+/<\text{ThingName}>\) to view all published messages.

If you can see the messages in AWS, then your thing can successfully publish data into the Cinterion® PLS62-W Wireless Module.

Next, test that the cloud can publish messages to your thing.

---

**Sending data from the cloud to your thing**

**Before you begin**

Ensure your thing can send information to the cloud.

For more information, see *Sending data from your thing to the cloud* on the previous page.

**To test that AWS can publish information to your thing:**

1. Create two subscribe topics in the Cinterion® PLS62-W Wireless Module.

   a. Using a terminal emulator, type:
   \(\text{at+emqsubopen=0,\"SubscribeFromCloud0\"}<\text{return}>\)
   \(\text{at+emqsubopen=1,\"SubscribeFromCloud1\"}<\text{return}>\)

   b. Check that the first two index numbers are assigned a topic each. Type:
   \(\text{at+emqsubopen?}<\text{return}>\)
   
   A list of index numbers and their assigned topics appears.

2. Use AWS to publish a message to each topic.

   a. In the Subscriptions panel, select Publish to a topic.

   b. In the Publish box, type:
   \(\text{SubscribeFromCloud0/ThingName}\)

   c. In the coding window, replace
   \(\text{Hello from AWS IoT console}\) with
   \(\text{Turn heating on}\)

---
d. Select Publish to topic.

![Publish to topic](image)

e. In the Publish box, type:

```
SubscribeFromCloud1/ThingName
```

f. In the coding window, replace

```
Turn heating on
Heat for 1 hour
```

g. Select Publish to topic.

View the AWS messages in the terminal emulator, in the following format:

```
+EMQ: <indexnumber>,<messageLength>
{
    "message": "<messageText>"
}
```

If you can see the messages in the terminal emulator, then AWS can successfully publish data to your thing through the Cinterion® PLS62-W Wireless Module.