Common SIA Service Use Cases

19.2.1

October 23, 2025

Akamai Technologies, Inc.
150 Broadway
Cambridge, Massachusetts
United States 02142
www.akamai.com

Copyright © 2025 Akamai Technologies, Inc. All rights reserved.

Akamai powers and protects life online. Leading companies worldwide choose Akamai to build, deliver, and secure their digital experiences — helping billions of people live, work, and play every day. Akamai Connected Cloud, a massively distributed edge and cloud platform, puts apps and experiences closer to users and keeps threats farther away. Learn more about Akamai's cloud computing, security, and content delivery solutions at akamai.com and akamai.com/blog, or follow Akamai Technologies on X and Linkedin.

Akamai is headquartered in Cambridge, Massachusetts in the United States with operations in more than 57 offices around the world. Our services and renowned customer care are designed to enable businesses to provide an unparalleled Internet experience for their customers worldwide. Addresses, phone numbers, and contact information for all locations are listed on www.akamai.com/locations.

© Akamai Technologies, Inc. All Rights Reserved. No part of this publication may be reproduced, transmitted, transcribed, stored in a retrieval system or translated into any language in any form by any means without the written permission of Akamai Technologies, Inc. While precaution has been taken in the preparation of this document, Akamai Technologies, Inc. assumes no responsibility for errors, omissions, or for damages resulting from the use of the information herein. The information in this document is subject to change without notice. Without limitation of the foregoing, if this document discusses a product or feature in beta or limited availability, such information is provided with no representation or guarantee as to the matters discussed, as such products/features may have bugs or other issues.

Akamai and the Akamai wave logo are registered trademarks or service marks in the United States (Reg. U.S. Pat. & Tm. Off). Products or corporate names may be trademarks or registered trademarks of other companies and are used only for explanation and to the owner's benefit, without intent to infringe.

Akamai Confidential: NDA Required for Release to Customers

Contents

Overview

The Akamai Subscriber Services Manager (SSM) mediates between a subscriber portal (using the Akamai Secure Internet Access (SIA) Consumer or SIA Small and Medium Business (SMB) applications) and the Postgres and Levant provisioners. Customers who prefer not to use the Akamai DNSi/SIA Applications Portal can use the SSM Application Programming Interface (API) to configure their own custom subscriber portals.

This document provides SSM administrators with step-by-step instructions and use case examples for performing the following common tasks using the SSM API:

- Creating and activating subscribers
- Upgrading a subscriber account from single to multiple profile service
- Troubleshooting the SSM
- Managing Akamai SIA Remote client support for subscribers

Each use case describes the APIs required to perform each task.

Typographical Conventions

The following typographical conventions are used in this document.

Table 1: Typographical Conventions

Element	Representation	Examples
Root prompt.	Hash tag.	#
Command prompt.	Fixed-width font followed by >.	ssm> or nom-config-hub>
Literal input, most often these are commands with options or arguments a user types.	Bold fixed-width font; always show prompt.	[root@apps0 centos]# **nom-tell cas-agent**
Inline references to commands and API calls.	Bold text.	Use the **POST /control** API to add a new subscriber to your SSM**.**
Variables representing site-specific information you must fill in as you type.	Bold italic within angle brackets if used in text. Bold italic fixed-width within angle brackets; always show prompt if any.	Replace the ***<deploymentFileName>*** argument with the name of the desired deployment. nom-config-hub> **deployments.deploy name=<*deploymentFileName*>**

Element	Representation	Examples
Inline references to objects, fields, and and options.	Plain fixed-width font.	Verify the `cert-fingerprints` field of the `client-config` object remains identical to its previous state.
Names of files, paths, commands, tools, processes.	Italic, if used in text. If used as part of a command, follows command format.	Open the *design.yaml* file. Access the following path: */var/nom/secrets/spson/proxy/* # **cd */var/nom/secrets/spson/proxy/***
URLs	Italic, include protocol (in other words, include "http://" if HTTP) if used in text. If used as part of command, follow command format.	For more information, see *https://kafka.apache.org/documentation/* .

Chapter 1: Accessing the SSM API

There are three ways to access the SSM APIs:

- Using curl commands on the UNIX shell.
- Invoking the **nom-http -s localhost -p 9090** command from the UNIX shell.
- From the AMC-X jump host tool.

How you format SSM APIs depends on whether you are using curl commands or a nom-http client (through the nom-http command or the AMC-X jump host tool). The nom-http client interacts with the SSM and allows a user to send API requests in batches which is useful for retrieving configuration information quickly. Using the nom-http client is typically more convenient than performing the same task using curl. The nom-http client is installed in the following location:

/usr/local/nom/sbin

By default, the SSM listens on port **9090** for REST API calls over HTTP. The examples in this document use the nom-http client; however the same information is conveyed regardless of the client you are using.

The examples that follow demonstrate the differences in formatting an API for the curl and nom-http clients. These examples show how to perform the same task (creating a new SSM subscriber account using the **POST /control** API) on each client.

NOTE: In the examples that follow and throughout this document, the SSM API prompt is **SSM>**; note that your prompt will be the IP address of your SSM host.

Curl Command Example

You enter curl commands on the UNIX shell. The following example shows how to enter the **POST /control** API formatted for curl:

```
# curl -i --header "Content-Type:\ application/json"  --request POST \
--data '{"id":"joe.coffee", "time-zone":"America/Los_Angeles"}' \
http://localhost:9090/control
```

nom-http Command Example

There are two ways to access the nom-http client:

- Use the AMC-X jump host tool (as described in the "Access the SSM API nom-http Client From Your AMC-X Jump Host Tool" section).
- Invoke the following command from the UNIX shell:

```
# /usr/local/nom/sbin/nom-http -s localhost#9090 -u <username> -p <password>
```

For more details on using the **nom-http** command, see the "Using the nom-http client" appendix in one of the following guides, depending on which service you are configuring:

- *SIA SMB SSM API Reference,* Release 19.2.1
- *SIA Consumer SSM API Reference,* Release 19.2.1

The example that follows shows how to enter the **POST /control** API formatted for the `nom-http` client:

```
SSM> POST /control
{"id":"sub123",
"time-zone":"America/Los_Angeles"}
```

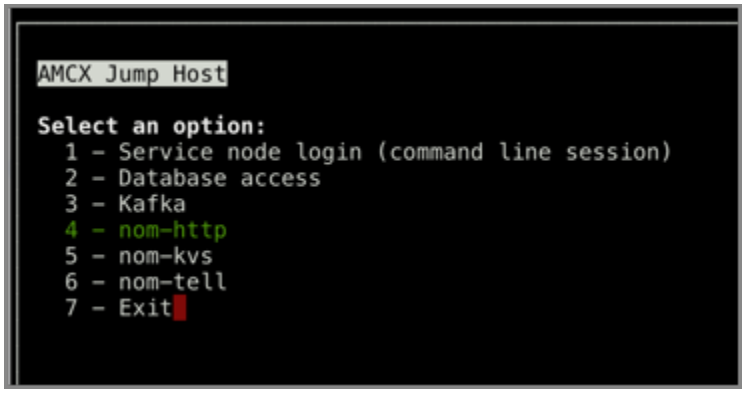
Access the SSM API nom-http Client from your AMC-X Jump Host Tool

Use the following steps to access the SSM API `nom-http` client from your AMC-X jump host tool:

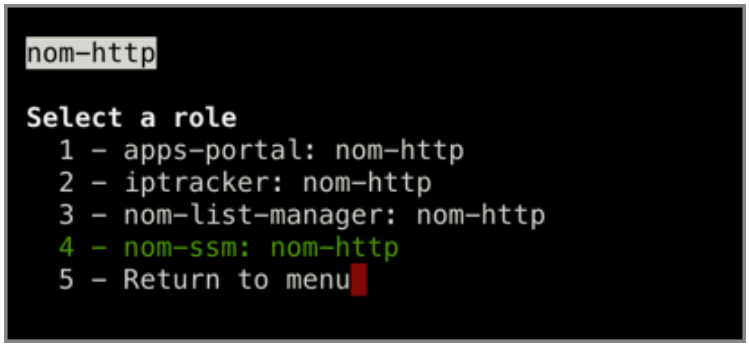
1. To open the AMC-X jump host tool, enter the following command on the node running `nom-config-hub`:

   ```
   # /usr/local/nom/sbin/amcx-jump-host
   ```

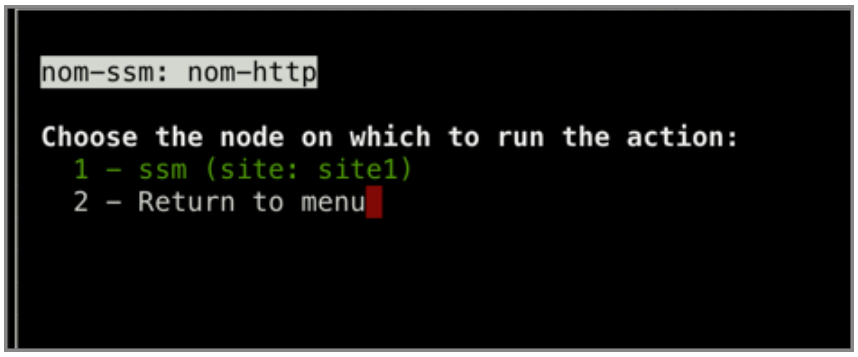
2. In your AMC-X jump host tool, select the **nom-http** option as demonstrated in the following example:



3. Select the **nom-ssm: nom-http** option as demonstrated in the following example:



4. Select the **ssm (site: site1)** option as demonstrated in the following example:



5. When prompted, enter your username and password to access the SSM API prompt, where you can start entering APIs:

NOTE: In this example and throughout this document, the SSM API prompt is `SSM>`; however, your own prompt is the IP address of your SSM host. For example, if your SSM host has the IP address 192.0.2.20, the SSM API prompt is `192.0.2.20>`.

Supported HTTP Methods

The SSM API uses the following HTTP methods:

- **GET**—Retrieves and displays information about a resource object. The **GET** method does not perform state change operations on the SSM.
- **POST**—Creates a new resource object or modifies one or more subordinate fields of a resource. You cannot update objects using the **POST** method.

- **PUT**—Modifies the entire content (all subordinate fields) of an existing resource object. Because this method replaces the entire object, you need to submit a complete representation of that object in the API call; in other words, you need to include all of the object's modifiable fields in the API call. Consider the following rules when entering a **PUT** call:
    - If you omit a modifiable field from the **PUT** request body, that field reverts back to the default setting; if no default setting exists for that field, the API clears any existing configuration from the field.
    - If an unmodifiable field is using the default setting, you do not need to include that field in the **PUT** request body.
- **PATCH**—Modifies only the specified fields for a resource object.
- **DELETE**—Removes a resource object from the system.

Chapter 2: Managing Subscriber Accounts and Services

This section provides use case examples for the following tasks:

- Adding a subscriber to the Subscriber Services Manager (SSM)
- Activating a service for a subscriber

The subscriber account needs to exist in your Akamai Secure Internet Access (SIA) deployment before you can enable a service for that subscriber. See the "Verify Whether a Subscriber Exists in your System" section to verify whether a subscriber account exists in your deployment.

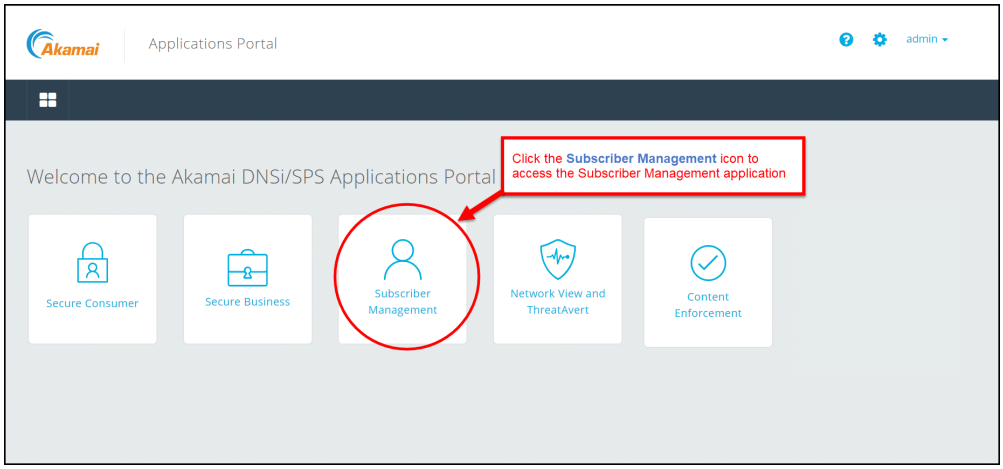
After verifying whether the subscriber exists in your system, perform one of the following tasks:

- If the subscriber does not exist in your SIA system, create a new subscriber account as described in the "Adding a Subscriber to Your System" section.
- If the subscriber already exists in your SIA system, enable services for that subscriber as described in the "Activating a Service for a Subscriber" section.

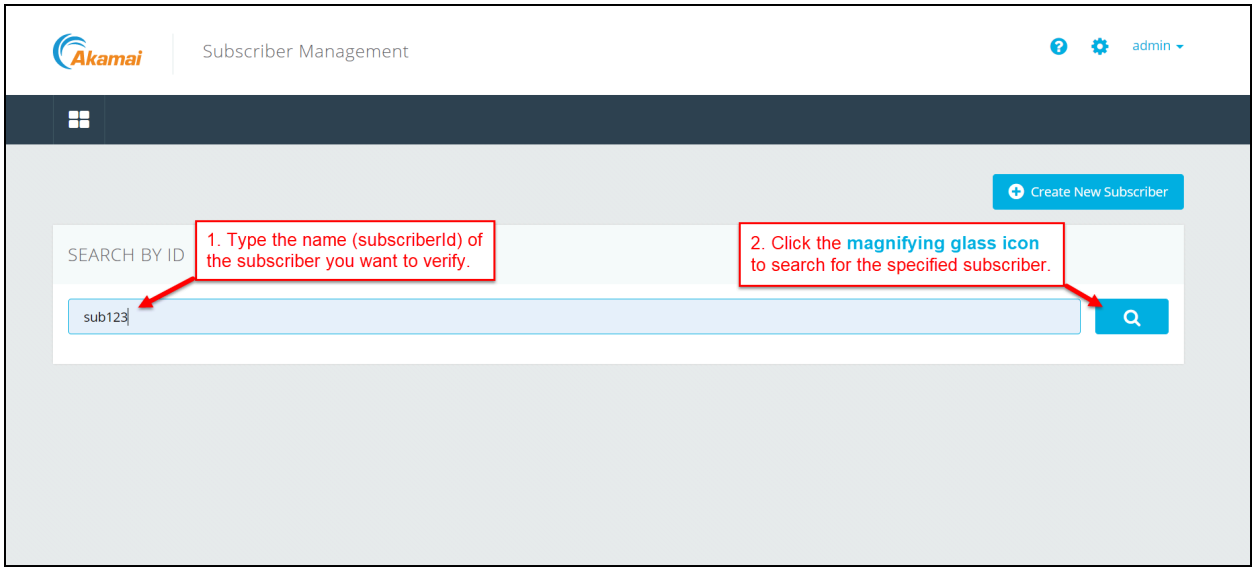
Verify Whether a Subscriber Exists in your System

To verify whether a subscriber exists in your SIA system, use the Subscriber Management application in the Akamai DNSi/SIA Applications Portal:

1. In your Akamai DNSi/SIA Applications Portal, open the Subscriber Management application as shown in the following example:



2. In the Subscriber Management application, search for the subscriber account you want to verify:



3. If the subscriber exists in the system, the Subscriber Management application returns information about the requested subscriber account. If the subscriber does not exist in the system, create the subscriber.

Adding a New Subscriber Account to your Akamai DNSi/SIA System

Use the following steps to create a subscriber account and activate services on that account:

- Use the SSM Application Programming Interface (API) to create a new subscriber account as described in the "Create a New Subscriber Using the SSM API" section.
- Activate one or more services for the subscriber account as described in the "Activating a Service for a Subscriber" section.

Create a New Subscriber Using the SSM API

Use the following API to add a new subscriber to your SSM:

```
SSM> POST /control {"id":"<accountId>", "time-zone":"<zone-id>"}
```

For more details about this API, see the "Create a Subscriber Account" section in one of the following guides, depending on which service you are configuring:

- *SIA SMB SSM API Reference,* Release 19.2.1
- *SIA Consumer SSM API Reference,* Release 19.2.1

Activate a Service for a Subscriber

You can activate the following services on a subscriber:

- SIA Consumer (also known as Personal Internet in the SSM API)
- SIA Small and Medium Business (SMB) (also known as Secure Business in the SSM API)
- Subscriber Safety (valid for SIA SMB and SIA Consume subscribers licensed for the Subscriber Safety service.)
- Security and Personalization Services Off Network (SPSON); this service is commonly known as the SIA Remote service and is valid for SIA SMB and SIA Consumer subscribers licensed for the SIA Remote service.

To activate a service, use the following **POST** API:

```
SSM> POST /control/<accountId>/service {"service": "<serviceId>",
"service-profile": "<profileId>"}
```

Replace the <*accountId*> argument with the name of the subscriber account for which to activate the desired service.

Replace the <*serviceId*> argument with one of the following service keywords:

- **personal-internet** (to specify the SIA Consumer service)
- **subscriber-safety**
- **secure-business** (to specify the SIA SMB service)
- **spson** (to specify the SIA Remote service)

  NOTE: Before setting the **spson** (SIA Remote) service, you need to configure the SIA Remote service and create the SPSON object on your SSM as described in Chapter 5, "Akamai SIA Remote client SSM API Use Cases."

Replace the <*profileId*> argument with the name of the profile you want to associate with the subscriber account; the profile you specify depends on the service you are configuring; possible profiles are:

- For the SIA Consumer (**personal-internet**) service, specify **household** or **perdevice**.
- For the SIA SMB (**secure-business**) service, specify **single-profile** or **multiple-profile**.
- For the Subscriber Safety service, replace the <*profileId*> argument with one of the following keywords, depending on the service you are configuring:
  - For SIA SMB, specify **sb-single** or **sb-multiple**.
  - For SIA Consumer, specify **standard**.
- For the SIA Remote service, specify **spson**.

Activating a service provisions the subscriber with the specified service and all the policy bindings in the service's Policy Binding Group. After you activate a service for a subscriber, an administrator or the subscriber can configure (enable or disable) the enforcement of that service on the subscriber's account. For more information on activating, enabling, and disabling a service, see the "Differences between activating and enabling a service" section in one of the following guides, depending on which service you are configuring:

- *SIA SMB SSM API Reference,* Release 19.2.1
- *SIA Consumer SSM API Reference,* Release 19.2.1

The examples that follow demonstrate how to use the SSM API to activate SIA SMB and SIA Consumer services on an existing subscriber account.

Examples

The examples that follow demonstrate how to perform the following tasks:

- Configure a New SIA SMB Subscriber Account
- Configure a New SIA Consumer Subscriber Account
- Activate the SIA Remote Service on an SIA SMB or SIA Consumer Subscriber Account

Example: Configure a New SIA SMB Subscriber Account

The example that follows uses the SSM API to perform the following tasks:

1. Adds a subscriber called **sub123** to the SSM.
2. Enables the SIA SMB and Subscriber Safety services in the subscriber.

In this example, the **"service":"secure-business"** construct specifies the SIA SMB service, and the **"service-profile":"multiple-profile"** construct indicates the service supports multiple profiles:

```
SSM> POST /control {"id": "sub123", "time-zone": "America/New_York"}
SSM> POST /control/sub123/service {"service":"secure-business",
"service-profile":"multiple-profile"}
```

Example: Configure a New SIA Consumer Subscriber Account

The example that follows activates the SIA Consumer service with multiple profiles for a subscriber called **paul.perdevice**. In this example, the **"service":"personal-internet"** construct specifies the SIA Consumer service, and the **"service-profile":"perdevice"** construct indicates the service supports multiple profiles:

```
SSM> POST /control {"id": "paul.perdevice", "time-zone":
"Europe/Berlin"}
SSM> POST /control/paul.perdevice/service
{"service":"personal-internet","service-profile":"perdevice"}
```

Example: Activate the SIA Remote Service on an SIA SMB or SIA Consumer Subscriber Account

The following **POST** API enables the SIA Remote (SPSON) service for a subscriber account:

```
SSM> POST /control/sub123/service {"service": "spson"}
```

NOTE: Before activating the SIA Remote service, you need to configure SIA Remote and create the SPSON object on your SSM as described in Chapter 5, “Akamai SIA Remote Client SSM API Use Cases.”

Using the SSM API to Remove a Service from a Subscriber Account

The following steps described how to use the SSM API to remove a service from subscriber account:

1. Use the following **DELETE** API to remove services from the subscriber account:

   ```
   SSM> DELETE /control/<accountId>/service/<service-name>
   ```

   Replace the ***<accountId>*** argument with the name of the subscriber account for which to remove the specified service. Replace the ***<service-name>*** argument with the name of the service you want to remove; possible services are:

   - **subscriber-safety**
   - **personal-internet** (to remove the SIA Consumer service)
   - **secure-business** (to remove the SIA SMB service)
   - **https-termination**
   - **spson** (to remove the SIA Remote service)

   NOTE: You can remove one service in an individual API call; if the subscriber has more than one service enabled, send an individual API call for each service until all services are removed from the subscriber account.

   For more details about using the SSM API to remove services from the subscriber account, see the “Deactivate a service for a subscriber” section in one of the following guides, depending on which service you are configuring:

   - *SIA SMB SSM API Reference,* Release 19.2.1
   - *SIA Consumer SSM API Reference,* Release 19.2.1

Using the SSM API to Remove a Subscriber Account from your Akamai DNSi/SIA System

Use the following **DELETE** API to remove a subscriber account from the SSM:

```
SSM> DELETE /account/<accountId>
```

Replace the ***<accountId>*** argument with the name of the subscriber account you want to remove from your Akamai DNSi/SIA system.

For more details about using the SSM API to remove a subscriber account from the SSM, see the “Delete a subscriber account” section in one of the following guides, depending on which service you are configuring:

- *SIA SMB SSM API Reference*, Release 19.2.1
- *SIA Consumer SSM API Reference,* Release 19.2.1

Adding Lines to a Subscriber Account

The SSM API uses "lines" to identify network traffic (through RADIUS or DHCP) and relate the traffic to a subscriber in the network. You associate lines with specific subscribers so that network traffic is sent to the appropriate subscriber. A subscriber account can have the following types of lines:

- Fixed lines—Fixed lines represent the customer premises equipment (CPE) that is connected to the Internet through a physical cable (for example, DSL, fiber, or cable). The connection between a subscriber's house and a fixed network is known as a line.
- Mobile lines—Mobile lines represent the connection between a mobile device and the mobile network is known as a line. Mobile devices are connected to the Internet through cellular or, in some instances, Wi-Fi connections.

Each line is associated with a line ID. Line lists map the line IDs to subscriber account IDs. For a mobile line, the line ID is the MSISDN associated with the subscriber device. For a fixed line, the line ID can be any name that clearly identifies the subscriber or CPE (for example, the login ID for the subscriber or the MAC address assigned to the device). The SSM and IP tracker send the line lists to Kafka for provisioning. When you add subscriber lines to a line list, the SSM provisions those lines for Kafka.

NOTE: Over time, you add and remove individual mobile lines to and from a subscriber account as devices are added and removed from that account. Removing a subscriber line from a line-list de-provisions those lines for Kafka

Prerequisites

Before adding fixed and mobile lines to a subscriber account, ensure the following prerequisites are met:

1. IP address tracking and subscriber mapping is set up and functioning properly as described in the *IP Tracker Configuration Manual*.
2. The appropriate mobile and fixed line lists exist in your deployment. If a line list does not already exist, use the following SSM API to add a line list to your deployment:

```
SSM> POST /line-list
{
  "has-devices": <boolean>,
  "name":"<list-id>"
}
```

   Replace the **<*boolean*>** argument with a boolean value indicating whether the specified list contains line nodes that need to be treated as devices. A value of **true** indicates that the line node for this list will be treated as devices. A value of **false** indicates that the line node for this list will not be treated as devices.

   NOTE: For fixed line lists, the **"has-devices"** field is always **false**. Only mobile networks can have lines that are devices.

   Replace the **<*list-id*>** argument with the name of a line list that exists in your Levant subscriber mapping configuration.

Configuration Steps

The process for adding lines to a subscriber account involves the following tasks:

1. If the subscriber account does not yet exist, create the account as described in the "Adding a New Subscriber Account to your Akamai DNSi/SIA System" section in Chapter 2, "Managing Subscriber Accounts and Services." If the subscriber account already exists, proceed to Step 2.
2. Use the following **POST** API to add devices (lines) to a subscriber account:

```
SSM> POST /account/<accountId>/line
{
  "add":[
  {
    "name":"<lineId>",
    "list":"<listId>" }, ...]
  }
}
```

NOTE: If you are adding mobile devices to a subscriber account, you need to add a line for each device the subscriber wants the account to manage.

The table that follows describes how to specify the constructs in the API:

Field	Description
<accountId>	Required. String that identifies the subscriber account for which to add subscriber lines.
"name"	Required. String that identifies the subscriber line you want to add to the specified line list. For a mobile line, use the MSISDN associated with the subscriber device. For a fixed line, use a name which clearly identifies the subscriber, such as the login ID for the subscriber.
"list"	Required. String that identifies the line list for which you are adding subscriber lines. The specified list needs to exist before you can add lines.

Examples

This section provides step-by-step examples for performing the following tasks:

- Add a Fixed Line (CPE) to a Subscriber Account
- Add Mobile Devices (Lines) to a Subscriber Account
- Add Converged (Fixed and Mobile) Lines to a Subscriber Account

Example: Add a Fixed Line to a Subscriber Account

1. Create the subscriber account as demonstrated in the example that follows; this example creates a new account called **paul.perdevice**, where the account gets set to the **America/Los_Angeles** time zone:

```
SSM> POST /control {"id":"paul.perdevice",
"time-zone":"America/Los_Angeles"}
```

2. Add a fixed line to the subscriber account CPE as demonstrated in the example that follows; this example adds a fixed line to the CPE associated with the **paul.perdevice** subscriber account created in Step 1. In addition, this API adds the fixed line to an existing line list called **fixed-DC**:

```
SSM> POST /account/<paul.perdivice/line { "add":[
{"name":"0000123456", "list":"fixed-DC" }]}}
```

Example: Add Mobile Devices (Lines) to a Subscriber Account

1. Create the subscriber account as demonstrated in the example that follows; this example creates a new account called **paul.perdevice**, where the account gets set to the **America/Los_Angeles** time zone:

```
SSM> POST /control {"id":"paul.perdevice",
"time-zone":"America/Los_Angeles"}
```

2. Add mobile lines for each device associated with the subscriber account as demonstrated in the example that follows; this example adds two mobile lines to the **paul.perdevice** subscriber account created in Step 1 and to an existing line list called **mobile-DC**:

```
SSM> POST /account/paul.perdivice/line { "add":[{
"name":"18005550100", "list":"mobile-DC"},
{"name":"18005550551", "list":"mobile-DC" }]}}
```

Example: Add Converged (Fixed and Mobile) Lines to a Subscriber Account

1. Create the subscriber account as demonstrated in the example that follows; this example creates a new account called **paul.perdevice**, where the account gets set to the **America/Los_Angeles** time zone:

```
SSM> POST /control {"id":"paul.perdevice",
"time-zone":"America/Los_Angeles"}
```

2. Add a fixed line to the subscriber account CPE as demonstrated in the example that follows; this example adds a fixed line to the CPE associated with the **paul.perdevice** subscriber account created in Step 1 and to an existing line list called **fixed-DC**:

```
SSM> POST /account/<paul.perdivice/line { "add":[
{"name":"0000123456", "list":"fixed-DC" }]}}
```

3. Add mobile lines for each device associated with the subscriber account as demonstrated in the example that follows; this example adds two mobile lines to the **paul.perdevice** subscriber account created in in Step 1 and to an existing line list called **mobile-DC**:

```
SSM> POST /account/paul.perdivice/line { "add":[{
"name":"18005550100", "list":"mobile-DC"},
{"name":"18005550551", "list":"mobile-DC" }]}}
```

Enable Multiple Scheduled Report Recipients in an SIA SMB Service

By default, SIA SMB subscribers can schedule block statistics reports to be sent to a single recipient only. To enable SIA SMB subscribers to send block statistics reports to multiple recipients, use the **portal.update** nom-tell command as demonstrated in the following example:

```
# nom-tell sportal
sportal> portal.update multiple-emails-enabled=true
response:
{
    type => 'portal.update'
}
```

To return the SIA SMB service to the default setting, where a subscriber can send block statistics reports to a single recipient only, use the **portal.update** nom-tell command to set the **multiple-emails-enabled** value to **false** as demonstrated in the following example:

```
sportal> portal.update multiple-emails-enabled=false
response:
{
    type => 'portal.update'
}
```

Chapter 3: Troubleshooting the SSM

This chapter describes how to perform the following troubleshooting tasks:

- Determine the services in which a subscriber is enrolled
- Determine when an opted-in subscriber sent information to the cloud
- Disable and reenable activated subscriber services

Determine the Services in Which a Subscriber is Enrolled

The information acquired in this task is valuable for creating a troubleshooting script based on subscriber account activation and web filter service status.

Use the following Subscriber Services Manager (SSM) Application Programming Interface (API) to display service enrollment information for a specific subscriber:

```
SSM> GET /account/<accountId>
```

The examples that follow highlight the sections of the SSM API output containing the desired enrollment information. The output indicates which services a subscriber is enrolled in, whether each service is currently enabled (enforced), and the date and time at which the service was enabled for the subscriber.

The following example shows sample output for a subscriber called **joe.coffee** who is enrolled in the Subscriber Safety and SIA Small and Medium Business (SMB) services, with multiple-profiles (device groups) enabled for the SIA SMB service:

```
SSM> GET /account/joe.coffee
{
    "id": "joe.coffee",
    "time-zone": "America/New_York",
    "subscriber-safety": {
        "enabled": true,
        "activated": true,
        "time": "2023-01-05T00:57:54.283+00:00",
        "bypass-pin-enabled": false
    },
    "personal-internet": {
        "enabled": false,
        "activated": false
    },
    "secure-business": {
```

```
        "enabled": true,
        "activated": true,
        "time": "2023-01-05T00:57:54.090+00:00",
        "type": "multiple-profile",
        "bypass-pin-enabled": false,
        "profile-count": 8,
        "profile-limit": 8,
        "spson-enabled": false,
        "bypass-private-dns": {
            "apple-private-relay": false
        }
    },
"https-termination": {
"enabled": false,
"activated": false
}
}
```

The following example shows sample output for a subscriber called **ursula.perdevice** who is enrolled in the Subscriber Safety and SIA Consumer (Personal Internet) services with the per-device service type:

```
SSM> GET /account/ursula.perdevice
{
   "id": "ursula.perdevice",
    "time-zone": "America/New_York",
    "subscriber-safety": {
        "enabled": true,
        "activated": true,
        "time": "2023-01-13T00:55:44.681+00:00",
        "bypass-pin-enabled": true,
        "bypass-pin-value": "11111"
    },
    "personal-internet": {
        "enabled": true,
        "activated": true,
        "time": "2023-01-13T00:55:44.642+00:00",
        "type": "perdevice",
        "bypass-pin-enabled": true,
        "bypass-pin-value": "11111",
        "profile-count": 2,
        "profile-limit": 4,
```

```
        "spson-enabled": false,
        "bypass-private-dns": {
            "apple-private-relay": false
        }
    },
    "secure-business": {
        "enabled": false,
        "activated": false
    },
    "https-termination": {
        "enabled": false,
        "activated": false
    }
}
```

NOTE: If the **"activated"** field is set to **true**, the subscriber is enrolled in the service and bound to that service's policy binding group. After a service is activated for a subscriber, an administrator or the subscriber can configure (enable or disable) the enforcement of that service on the subscriber's account; the **"enabled"** field indicates whether an activated service is currently being enforced for the subscriber.

For more details about this API, see the "Retrieve subscriber account service details" section in one of the following manuals, depending on your service:

- *SIA SMB SSM API Reference*, Release 19.2.1
- *SIA Consumer SSM API Reference*, Release 19.2.1

Determine When an Opted-in Subscriber Sent Information to the Cloud

This section describes how to retrieve information that is valuable for parsing device timestamps and quickly indicating a subscriber's status.

The following **GET** API displays the last time a subscriber account's devices accessed the network:

```
SSM> GET /series/device/last-seen?view=<viewId>
```

Replace the ***<viewId>*** argument with the name of the subscriber account for which to display blocked domain requests.

The example that follows retrieves last-seen data for three devices associated with the subscriber account called **sub123**. The last seen time is expressed in the UNIX format and highlighted in yellow in the example:

```
SSM> GET /series/device/last-seen?view=sub123
{
"00:16:5c:d9:3e:93":1524258000000,
"00:16:6d:84:1f:1f":1525129200000,
"00:16:c0:c1:a4:23":1524783600000
}
```

Disable and Re-enable Activated Subscriber Services

In the event of an emergency, an administrator or subscriber can disable the enforcement of an active service as a troubleshooting step.

The following **PATCH** API toggles service activation status for a specific subscriber account:

```
SSM> PATCH /control/<accountId>/service/<serviceId>
{
"enabled": <boolean>
}
```

Replace the **<*accountId*>** argument with the desired subscriber identifier. Replace the **<*service-name*>** argument with the name of the service you want to disable or reenable; possible service names are:

- **subscriber-safety**
- **personal-internet** (to remove the SIA Consumer service)
- **secure-business** (to remove the SIA SMB service)
- **https-termination**
- **spson** (to remove the SIA Remote service)

Replace the **<*boolean*>** argument with **false** to disable the specified service or **true** to reenable a disabled service.

See the "Update service activation status for a subscriber" section for more details about using the SSM API to remove services from the subscriber account for more details on this PATCH API.

Chapter 4: Upgrading a Subscriber Account from Single to Multiple Profile Service

To upgrade an Secure Internet Access (SIA) Small and Medium Business (SMB) or SIA Consumer subscriber account from single to multiple profile service, you need perform the following tasks:

1. Retrieve target account data and save that data on a local machine as described in the "Retrieve and Save Target Account Data" section.

   NOTE: This step is important because you lose single-profile data after you delete single profile services as described in Step 2.

2. Remove all single profile services from the subscriber account as described in the "Using the SSM API to Remove a Service from a Subscriber Account" section.
3. Activate multiple profile services for the subscriber account as described in the "Activate Multiple Profile Services for the Subscriber Account" section.
4. Configure the target account data you saved in Step 1 as described in the "Retrieve and Save Target Account Data" section.

These steps are described in detail in the sections that follow.

Step 1: Retrieve and Save Target Account Data

Use the following APIs to retrieve target account data; save this data on a local machine because you need to re-configure this data on the subscriber account after upgrading the account to use multiple profile services:

- **GET /<*service*>/<*accountId*>/profile**
- **GET /<*service*>/<*accountId*>/profile/<*profileId*>/protection**
- **GET /<*service*>/<*accountId*>/blacklist**
- **GET /<*service*>/<*accountId*>/whitelist**
- **GET /account/<*accountId*>/resource/<*resourceId*>**
- **GET /ss/<*accountId*>**

The table that follows describes how to replace the variable arguments in the **GET** APIs you use to retrieve account data:

Table 4-1: How to Set GET API Arguments When Retrieving Subscriber Account Data

Argument	Description
<*service*>	Specifies the service the subscriber you are upgrading is enrolled in. Replace the **<*service*>** argument with **sb** (for the SIA SMB service) or **pi** (for the SIA Consumer service).
<*accountId*>	Identifies the subscriber account you are upgrading. Replace the **<*accountId*>** argument with the identifier associated with the subscriber for which to display blacklist details.
<*profileId*>	Identifies the profile for which to display configuration details; replace the **<*profileId*>** argument with the name of a profile. You need to retrieve information from all profiles in your system.
<*resource-id*>	Required. String that identifies the resource object for which to retrieve configuration information. Replace the **<*resourceId*>** argument with one of the following resource object keywords: • **business-sec-template**—Retrieves malware and phishing filtering block page configuration details. • **business-web-template**—Retrieves web filtering block page configuration details. • **residential-sec-template**—Retrieves malware and phishing block page configuration details. • **residential-web-template**—Retrieves web filtering block page configuration details. You need to retrieve information from all resources in your system.

The examples that follow provide sample output from the SSM APIs used to retrieve and save target information required for upgrading a subscriber account from a single to multiple profile service.

NOTE: While the examples in this section show output for an SIA SMB subscriber account, the output for an SIA Consumer subscriber account is similar.

- Use the following **GET** API to retrieve profile information for a subscriber account:

```
SSM> GET /sb/sub123/profile
{
 "total": 1,
 "limit": 1,
 "content": [
  {
   "id": "07e75a70-0ca5-45af-b5ca-ed17838a5eaf",
   "name": "sb_group_visitor",
   "protection": "custom",
   "devices": [],
   "schedules": [
    {
     "name": "internet-off0",
     "type": "internet-off",
     "day": [ "MO" ],
     "enabled": true,
     "timeStart": "00:00",
     "timeEnd": "24:00",
     "categories": []
    },
    {
     "name": "homework0",
     "type": "homework",
     "day": [ "TU" ],
     "enabled": true,
     "timeStart": "00:00",
     "timeEnd": "24:00",
     "categories":["sb-chat", "sb-sports", "sb-travel"]
     }
    ],
   "default": true,
```

```
    "protection-enabled": true,
    "inspect-stream": false,
    "safe-search": true,
    "http-termination": false,
    "safe-search-services": {
    "youtube": true,
    "google": true,
    "bing": true
    }
   }
 ]
}
```

- If the subscriber has custom protections configured, use the following **GET** API to retrieve custom protection categories for a subscriber account:

```
SSM> GET /sb/sub123/profile/sb_group_visitor/protection
{
 "total": 3,
  "limit": 3,
  "content": [
   {
    "name": "sb-arts",
    "description": "Arts"
   },
   {
    "name": "sb-games",
    "description": "Online gaming sites"
   },
   {
    "name": "sb-news",
    "description": "News"
   }
  ]
}
```

- If the subscriber has blacklist nodes configured, use the following **GET** API request to retrieve a list of nodes on the blacklist belonging to a specific subscriber:

```
SSM> GET /sb/sub123/blacklist
{
"total": 1,
"limit": 10,
"content": [ "yandex.com" ]
}
```

- If the subscriber has whitelist nodes configured, use the following **GET** API request to retrieve whitelist nodes:

```
SSM> GET /sb/sub123/whitelist
{
"total": 1,
"limit": 10,
"content": [ "google.com" ]
}
```

NOTE: If the subscriber does not have nodes in a whitelist, skip this step.

- If the subscriber account had resource properties configured, use the following **GET** API to retrieve resource properties for a particular subscriber resource; you need to retrieve resource property data for all resources configured for the subscriber account:

```
SSM> GET /account/sub123/resource/business-web-template
{
 "name": "business-web-template",
 "type": "text/html",
 "fields": {
  "color": "#1BE300",
  "web-page-show-name": "",
  "web-page-show-code": "",
  "web-page-admin-phone": "11-22-33-44-55",
  "web-page-message": "website is blocked",
  "tagline": "test",
  "web-page-show-message": "true",
  "web-page-show-admin": "true",
  "web-page-admin-email": "blocked@sportal.com"
 }
}
```

For details about the output of this API, see the “Retrieve a preview of a specific subscriber resource” section in one of the following guides, depending on which service you are configuring:

  - *SIA SMB SSM API Reference,* Release 19.2.1
  - *SIA Consumer SSM API Reference,* Release 19.2.1

- Use the following **GET** API to determine whether subscriber safety is enabled for an SIA SMB subscriber account:

```
SSM> GET /ss/sub123

Response:
 {
  "profiles": [
    {
     "name": "sb_group_visitor",
     "enabled": false
    }
   ]
 }
```

NOTE: The **GET /ss/<*accountId*>** API is valid for SIA SMB subscribers only.

Step 2: Remove all Current Single-profile Services from the Subscriber Account

Use the following API to remove the SIA SMB service from the subscriber account:

```
SSM> DELETE /control/<accountId>/service/<service-name>
```

Replace the **<*accountId*>** argument with the subscriber account identifier for which to remove the single profile SIA SMB or SIA Consumer service. Replace the **<*service-name*>** argument with one of the following keywords, depending on the service you are removing:

- **secure-business** (to remove the SIA SMB service from a subscriber account).
- **subscriber-safety** (to remove the Subscriber Safety service from an SIA SMB subscriber account).
- **personal-internet** (to remove the SIA Consumer service from a subscriber account).

For more details about using the SSM API to remove services from an SIA SMB or SIA Consumer subscriber account, see the “Deactivate a service for a subscriber” section in one of the following manuals, depending on your service:

- *SIA SMB SSM API Reference*, Release 19.2.1
- *SIA Consumer SSM API Reference*, Release 19.2.1

Remove Single-profile SIA SMB Services from a Subscriber Account

1. Use the following **DELETE** API to remove the SIA SMB service from the subscriber account:

   ```
   SSM> DELETE /control/<accountId>/service/secure-business
   ```

   Replace the ***<accountId>*** argument with the desired subscriber identifier.

2. If the Subscriber Safety service is enabled on the account, use the following **DELETE** API to remove the Subscriber Safety service from the subscriber account:

   ```
   SSM> DELETE /control/<accountId>/service/subscriber-safety
   ```

   Replace the ***<accountId>*** argument with the desired subscriber identifier.

Remove the Single-profile SIA Consumer Service from a Subscriber Account

1. Use the following **DELETE** API to remove the SIA SMB service from the subscriber account:

   ```
   SSM> DELETE /control/<accountId>/service/personal-internet
   ```

   Replace the ***<accountId>*** argument with the desired subscriber identifier.

2. If the Subscriber Safety service is enabled on the account, use the following **DELETE** API to remove the Subscriber Safety service from the subscriber account:

   ```
   SSM> DELETE /control/<accountId>/service/subscriber-safety
   ```

   Replace the ***<accountId>*** argument with the desired subscriber identifier.

Step 3: Activate Multiple Profile Services for the Subscriber Account

After removing the single profile service from an SIA SMB or SIA Consumer subscriber account as described in “Step 2: Remove all Current Single-profile Services from the Subscriber Account,” you can enable multiple profile service on that account.

Use the following API to enable multiple-profile service on an SIA SMB or SIA Consumer subscriber account:

```
SSM> POST /control/<accountId>/service
{"service":"<service-name>","service-profile":"<profileId>"}
```

The table that follows describes the variables in the **POST /control/<*accountId*>/service** API:

Argument	Description
<*accountId*>	Identifies the subscriber account identifier for which to enable a service. Replace the **<*accountId*>** argument with the name of the subscriber account on which to enable a service.
<*serviceId*>	Identifies the service to activate for the specified subscriber account. Replace the **<*serviceId*>** argument with one of the following keywords: • For an SIA Consumer subscriber: **personal-internet** • For an SIA SMB subscriber: **secure-business** • For an SIA SMB subscriber with the Subscriber Safety service: **subscriber-safety**
<*profileId*>	Identifies the profile you want to associate with the subscriber account; the profile you specify (for the) depends on the service you are configuring; possible service types are: • SIA Consumer: Specify **household** or **perdevice**. • SIA SMB: Specify **single-profile** or **multiple-profile**. • Subscriber Safety: Specify standard **sb-single** or **sb-multiple**.

For more details on using the SSM API to enable multiple profile service for a subscriber, see the “Activate a service for a subscriber” section in one of the following guides, depending on which service you are configuring:

- *SIA SMB SSM API Reference,* Release 19.2.1
- *SIA SMB Consumer API Reference,* Release 19.2.1

Example: Enable Multiple Profile SIA SMB Services on a Subscriber Account

In this example, the **"service":"secure-business"** construct specifies the SIA SMB service, and the **"service-profile":"multiple-profile"** construct indicates the service supports multiple profiles:

```
SSM> POST /control/sub123/service
{"service":"secure-business","service-profile":"multiple-profile"}

SSM> POST /control/sub123/service {"service":"subscriber-safety",
"service-profile":"sb-multiple"}
```

Example: Enable the Multiple Profile SIA Consumer Service on a Subscriber Account

The example that follows activates the SIA Consumer service with multiple profiles for a subscriber called **paul.perdevice**. In this example, the **"service":"personal-internet"** construct specifies the SIA Consumer service, and the **"service-profile":"perdevice"** construct indicates the service supports multiple profiles:

```
SSM> POST /control {"id": "paul.perdevice"}

SSM> POST /control/paul.perdevice/service
{"service":"personal-internet","service-profile":"perdevice"}
```

Step 4: Configure Target Account

After enabling multiple-profile service on an SIA SMB or SIA Consumer subscriber account, you need to re-configure the account with the data you saved to a local machine in "Step 1: Retrieve and Save Target Account Data."

This section describes how to re-configure subscriber accounts for the following services:

- SIA SMB (see the "Configure an SIA SMB Account" section).
- SIA Consumer (see the "Configure an SIA Consumer Account" section).

Configure an SIA SMB Account

Use the following steps to re-configure an SIA SMB account with the data you saved in Step 1: Retrieve and Save Target Account Data."

1. Create subscriber profiles as demonstrated in the example that follows. This example creates three profiles for the SIA SMB subscriber account called **sub123**:

```
SSM> POST /sb/sub123/profile
[
 {
  "name": "sb_group_visitor",
  "protection": "custom",
  "safe-search-services": {
  "google": true,
  "bing": true,
  "youtube": true
 },
  "default": true,
  "internet-access": true,
  "safe-search": true
 },
 {
  "name": "sb_group_employee",
  "protection": "medium",
  "safe-search-services": {
  "google": true,
  "bing": true,
  "youtube": true
 },
  "internet-access": true,
  "safe-search": true,
  "default": false
 },
 {
  "name": "sb_group_blocked_all_hidden",
  "protection": "none",
  "safe-search-services": {
  "google": false,
```

```
  "bing": false,
  "youtube": false
 },
  "internet-access": false,
  "safe-search": true,
  "default": false
 }
]
```

2. If the subscriber had schedulers, add schedulers to the main profile as demonstrated in the example that follows. This example creates an internet-off schedule for the SIA SMB subscriber account called **sub123**:

```
SSM> POST /sb/sub123/profile/sb_group_visitor/schedule
Request (example for `internet-off`):
{
"name": "internet-off0",
"type": "internet-off",
"day": [ "MO" ],
"enabled": true,
"timeStart": "00:00",
"timeEnd": "24:00",
"categories": []
}
```

   If the subscriber has no schedulers, skip this step and go to Step 3.

3. If the subscriber account had custom protection categories configured, add the custom categories to the main profile as demonstrated in the example that follows. This example adds three protection categories (**sb-sports**, **sb-games**, and **sb-politics**) to a profile called **sb_group_visitor**:

```
SSM> POST /sb/sub123/profile/sb_group_visitor/protection
["sb-sports", "sb-games", "sb-politics"]
```

   If the subscriber has no custom protection categories configured, skip this step and go to Step 4.

4. If required, add nodes to the subscriber's **blacklist** as demonstrated in the example that follows. This example adds one node (the domain **violence.com**) to a blacklist belonging to the subscriber account called **sub123**:

```
SSM> POST /sb/sub123/blacklist
{
 "add":[
  {
   "node": "violence.com",
   "type": "core-domain"
  }
 ],
  "remove":[
 ]
}
```

   If the subscriber's blacklist did not contain nodes, skip this step and go to Step 5.

5. If required, add nodes to the subscriber's **whitelist** as demonstrated in the example that follows. This example adds one node (the domain **google.com**) to a whitelist belonging to the subscriber account called **sub123**:

```
SSM> POST /sb/sub123/whitelist
{
 "add":[
  {
   "node": "google.com",
   "type": "core-domain"
  }
 ],
  "remove":[
 ]
}
```

   If the subscriber whitelist did not contain nodes, skip this step and go to Step 6.

6. If the subscriber account had resource properties, use the following **PUT** API to add those resource properties to the subscriber account:

```
SSM> PUT /account/sub123/resource/business-sec-template
{
"sec-page-admin-email": "blocked@sportal.com",
"sec-page-message": "website is blocked",
"sec-page-show-message": "true",
"sec-page-show-admin": "true",
"sec-page-admin-phone": "11-22-33-44-55",
"rtl": ""
}
```

   If the subscriber account had no resource properties configured, skip this step.

Configure an SIA Consumer Account

Use the following steps to configure the target account data you saved in the "Retrieve and Save Target Account Data" section.

1. Create profiles as demonstrated in the example that follows. This example creates a profile (called **kids**) with **strict** protections and safe search enabled:

```
SSM> POST /pi/paul.perdevice/profile
[
 {
  "name": "kids",
  "protection": "strict",
  "safe-search": true
 }
...
]
```

2. If the subscriber had schedulers, add the schedulers to the main profile as demonstrated in the example that follows. This example creates an internet-off and homework-time schedule for devices attached to the **kids** profile:

```
SSM> POST /pi/paul.perdevice/profile/kids/schedule
{
"name": "internet-off0",
"type": "internet-off",
"day": [ "MO" ],
"enabled": true,
"timeStart": "00:00",
"timeEnd": "24:00",
"categories": []
}
```

Request (example for `homework`):

```
{
"name": "homework0",
"type": "homework",
"day": [ "TU" ],
"enabled": true,
"timeStart": "16:00",
"timeEnd": "24:00",
"categories": ["pi-chat", "pi-sports"]
}
```

If the subscriber account does not require schedulers, skip this step and go to Step 3.

3. If the subscriber account had custom protection categories configured, add the custom categories to the main profile as demonstrated in the following example. This example adds three custom protection categories to a profile called **visitor**:

```
SSM> POST /pi/paul.perdevice/profile/visitor/protection
["pi-gambling", "pi-weapons","pi-shopping"]
```

If the subscriber does not require custom protection categories, skip this step and go to Step 4.

4. If required, add nodes to the subscriber's blacklist as demonstrated in the example that follows. This example adds two nodes to a blacklist belonging to the SIA Consumer subscriber called **paul.perdevice**:

```
SSM> POST /pi/paul.perdevice/blacklist
{
 "add":[
  {
   "node": "violence.com",
   "type": "core-domain"
  }
 ],
  "remove":[
 ]
}
```

   If you do not need to add nodes to a subscriber's blacklist, skip this step and go to Step 5.

5. If required, add nodes to the subscriber's **whitelist** as demonstrated in the example that follows. This examples adds two nodes to a whitelist belonging to the SIA Consumer subscriber called **paul.perdevice**:

```
SSM> POST /pi/paul.perdevice/whitelist
{
 "add":[
  {
   "node": "google.com",
   "type": "core-domain"
  }
 ],
  "remove":[
 ]
}
```

   If you do not need to add nodes to a subscriber's whitelist, skip this step and go to Step 6.

6. If the subscriber account had resource properties, add those resource properties to the subscriber account as demonstrated in the following example:

```
SSM> PUT /account/paul.perdevice/resource/residential-sec-template
{
"sec-page-admin-email": "blocked@sportal.com",
"sec-page-message": "website is blocked",
"sec-page-show-message": "true",
"sec-page-show-admin": "true",
"sec-page-admin-phone": "11-22-33-44-55",
"rtl": ""
}
```

   If the subscriber account had no resource properties configured, skip this step.

Chapter 5: Akamai SIA Remote Client SSM API Use Cases

The Akamai SIA Remote client protects Secure Internet Access (SIA) Small and Medium Business (SMB) and SIA Consumer devices when those devices are off-network or roaming; in other words, devices that have the SIA Remote client enabled do not lose SIA SMB or SIA Consumer protections when not connected to SIA DNS servers. To configure your system to support Akamai's SIA Remote service, see the following documents:

- *SIA 19.2.1 Products: Installation Using AMC-X*
- *SIA 19.2.1 Products: AMC-X Overview and Deployment Design*

In release prior to Release 19.2.0, you used Subscriber Services Manager (SSM) APIs to configure Akamai SIA Remote in your system. With Release 19.2.0 and later releases, most of the Akamai SIA Remote configuration is built into the deployment design, using the design.yaml file. For deep link configuration and to manage client versions, you use nom-tell with nom-ssm and cas-agent.

In previous releases, Akamai SIA Remote supported time-bound (expiring) deep links only. With Release 19.2.1 and later releases SIA Remote supports the following types of deep links:

- Time-bound (expiring) deep links, which require a time-to-live (TTL) timestamp and no longer work after the TTL passes.
- Non-expiring (fixed) deep links that do not expire. We recommend using non-expiring deep links in MDM environments so that you do not need to regenerate the deep link and update the MDM configuration every time the link expires.

  Caution! Because non-expiring deep links contain secret data, do not use non-expiring deep links in MDM deployments where an end user can easily access that data. Because non-expiring deep links do not expire, the data embedded in the links is not encrypted.

The CAS agent uses entitlement codes to authenticate deep links. Each deep link type has its own entitlement code; in other words, time-bound deep links have entitlement codes that are separate from non-expiring deep link entitlement codes.

NOTE: Because time-bound and non-expiring deep links have different entitlement codes, rotate expiring and non-expiring entitlement codes independently from one another. In other words, rotate the non-expiring entitlement code independently of any existing expiring entitlement.

Consider the following information before configuring deep links:

- You need to implement client clean-up before deploying a new time-bound deep link.
- Do not store deep links in any configuration file or logs. If a client needs to determine whether a deep link has changed, use the hash value of the link instead of the link itself.
- Enable deep links during the setup phase when configuring your environment to support the off-network feature.

After completing the off-network feature setup phase, you can perform the following tasks:

- Retrieve the deep link and the associated entitlement code.
- Rotate the entitlement codes (revoke the existing deep links).

To ensure that your deployment design supports Akamai SIA Remote, see the "SIA Remote Considerations" section of *SIA 19.2.1 Products: AMC-X Overview and Deployment Design*.

To see the nom-tell commands for configuring your SSM with a deep link base URL, see the "Deep link setup" section of *SIA 19.2.1 Products: Installation Using AMC-X*.

After the Akamai SIA Remote service activates, your subscribers can log into your subscriber portal and activate Akamai SIA Remote service support on eligible devices.

This chapter describes how to perform the following tasks:

- Provision SIA Remote service support on a subscriber account
- Configure Akamai SIA Remote client device registration
- Generate a time-bound deep link from a third party portal
- Generate a non-expiring deep link from a third party portal
- Rotate the entitlement code for deep links
- Rename an Akamai SIA Remote client roaming device
- Set internal domain access for the devices that are associated with a subscriber account
- Retrieve SIA Remote service configuration details for a subscriber account
- Display the status of all SIA Remote client-enabled roaming devices for a specific subscriber
- Display generic Akamai SIA Remote client errors
- Delete a roaming Akamai SIA Remote client device
- Deprovision the Akamai SIA Remote client support for a subscriber account

Provisioning SIA Remote Service Support on a Subscriber Account

After signing up for the Akamai SIA Remote client, a CSP administrator or a back office application calls the SSM API to perform the following subscriber provisioning tasks:

1. Activate and enable the SIA Remote client for the subscriber account.
2. Create the SIA Remote service Security and Personalization Services Off Network (SPSON) object for the subscriber account.

Enable the SIA Remote Service for a Subscriber Account

The following **POST** API enables the SIA Remote service for a subscriber account:

```
SSM> POST /control/<accountId>/service {"service": "spson"}
```

Replace the ***<accountID>*** argument with the name of the subscriber account on which to enable SIA Remote service. The example that follows enables the SIA Remote service on a subscriber account called **joe.coffee**:

```
SSM> POST /control/joe.coffee/service {"service":"spson"}
```

Create the SIA Remote Service SPSON Object for the Subscriber Account

The following **POST** API demonstrates how to create the Akamai SIA Remote service (SPSON) object for a subscriber account:

```
SSM> POST /account/<accountId>/spson {"allowed-devices":
<number-of-devices>,"mdm-deeplink-enabled":<boolean>,
"internal-domains":["<domain>, …"]}
```

NOTE: Time-bound deep links are automatically supported on subscriber accounts that have the SIA Remote service enabled. To enable support for non-expiring deep links, you need to include the optional **"mdm-deeplink-enabled"** construct in the **POST /account/<*accountId*>/spson** API.

The table that follows describes how to set the variables in the **POST /account/<*accountId*>/spson** API:

Field	Description
<***accountId***>	Required. Specifies the subscriber for which to add SIA Remote service-enabled devices; replace the **<*accountId*>** argument with the desired subscriber.
"allowed-devices"	Required. Specifies the maximum number of SIA Remote service devices allowed for the specified subscriber account. Replace the **<*number-of-devices*>** argument with an integer in the range assigned to your ISP. **NOTE:** This field is valuable for ISPs offering tiered service support (for example, one price tier for 5 devices, a higher price tier for 10 devices and so forth).
"mdm-deeplink-enabled"	Optional. Enables and disables non-expiring deep link support on the subscriber account. Replace the **<*boolean*>** argument with one of the following values: ● **true**—Enables non-expiring deep link support on the specified subscriber account and generates an entitlement code for the deep links. The CAS agent uses this entitlement code to authenticate the non-expiring deep link. ● **false**—Disables non-expiring deep link support on the specified subscriber account. **NOTE:** We recommend enabling non-expiring deep links for subscribers in MDM environments so that you do not need to regenerate the deep link and update the MDM configuration every time the link expires. The default value is **false**.

"internal-domains"	Optional. Allow devices belonging to the specified subscriber account to access to your corporation's internal domains. **NOTE:** While local domains apply to all subscribers in an ISP, internal domains apply to a specific subscriber only. Replace the ***<domain>*** argument with an internal domain name; you can specify multiple domains, separating each domain with a comma. Be aware this field has a maximum limit of 999 characters. **NOTE:** Before enabling internal domain access for an individual subscriber account, use the **GET /account/<*accountId*>/spson** API to determine whether internal domains are already set for the subscriber account. In the API response, the "internal-domains" field lists all internal domains the subscriber can access. Use the **GET /account/<*accountId*>/spson** API to verify internal domain access for a subscriber; see the "Retrieving SIA Remote Service Configuration Details for a Subscriber Account" section details.

Example: Create a SPSON object for a Subscriber Account that Supports Time-bound Deep Links

The example that follows demonstrates how to create an SIA Remote service SPSON object for a subscriber account that supports time-bound deep links. The API in the example performs the following configuration for a subscriber accounts called **joe.coffee**:

- Generates an entitlement code for the time-bound deep links.
- Enables support for up to **10** SIA Remote service devices.
- Configures access to the internal domain **akamai.com** (ensuring the SIA Remote devices that are associated with the account can access the **akamai.com** domain)..

```
SSM> POST /account/joe.coffee/spson {"allowed-devices": 10
"internal-domains":["akamai.com"]}
```

NOTE: Because the API omits the **"mdm-deeplink-enabled": true** construct, the **joe.coffee** account supports time-bound deep links only.

Example: Create a SPSON object for a Subscriber Account that Supports Non-expiring Deep Links

The example that follows demonstrates how to enable non-expiring deep links when creating an SIA Remote service SPSON object for a subscriber account. The API in the example performs the following configuration for a subscriber accounts called **joe.coffee**:

- Enables non-expiring deep links and generates an entitlement code for authenticating those deep links.
- Enables support for up to four (**4**) SIA Remote service devices.

```
SSM> POST /account/art.gallery/spson {"allowed-devices": 4,
"mdm-deeplink-enabled": true}
```

Configuring Akamai SIA Remote Client Device Registration

You can configure your company's Akamai SIA Remote client to use deep link connections to register new subscriber devices for a subscriber account. The SIA Remote service supports the following types of deep links:

- Time-bound (expiring) deep links, which require a time-to-live (TTL) timestamp and no longer work after the TTL passes.
- Non-expiring (fixed) deep links that do not expire. We recommend using non-expiring deep links in MDM environments so that you do not need to regenerate the deep link and update the MDM configuration every time the link expires.

  Caution! Because non-expiring deep links contain secret data, do not use non-expiring deep links in MDM deployments where an end user can easily access that data. Because non-expiring deep links do not expire, the data embedded in the links is not encrypted.

Use the following steps to register new subscriber devices for the Akamai SIA Remote client:

1. Use one of the following **POST** API requests to generate a deep link connection for an SIA Remote service:
    - Use the following **POST** API to generate a time-bound (expiring) deep link:

```
SSM> POST /account/sub123/spson/deeplink

{
   "link": "<deepLink>",
   "deeplink-ttl": <ttl>
}
```

    - Use the following **POST** API to generate a non-expiring (fixed) deep link:

```
SSM> POST /account/<accountId>/spson/mdm_deeplink
{
   "link": "<deepLink>",
}
```

2. Use the desired communication channel (for example, an email or text message) to send the generated deep link to the subscriber.
3. When the subscriber receives the deep link on their device (the device the subscriber wants to register), clicking that link performs the following tasks:
    - If your company's Akamai SIA Remote client application is not already installed on the device, the deep link installs the client application on the device.
    - Prompts the subscriber to enter their first and last name to register the device.

Generating a Time-bound Deep Link from a Third Party Portal

After integrating deep link generation on your subscriber support (as described in the *SIA 19.2.1 Products: AMC-X Overview and Deployment Design* and *SIA 19.2.1 Products: Installation Using AMC-X* manuals), you can generate a deep link connection for your SIA Remote service subscribers from your support portal. The example that follows generates a deep link connection for a subscriber called **joe.coffee**:

```
SSM> POST /account/joe.coffee/spson/deeplink
{
   "link":
"https://cx2-static.spson-test.akamaiapis.net/apps?apidomain=custom
er1.spsoncas-preprod.akamaiap
is.net&regid=27egbjy5xcftd5cy7iyq-gi2nlj5skqvrpqidccuextl6ee-6rmczk
rvbm2q4lshnbtt7g25wi&entcode=2-a27-160359
9266-66zwnn6gl3t44broio2q-6kvf4cfnvf2zbdfaikus5gdai5ywewvayy4if3tka
e6rttl46opcm6tfh4-x7frfbbnekccira3g6aebqu
s3m",
   "deeplink-ttl": 86400
}
```

Generating a Non-expiring Deep Link from a Third Party Portal

After integrating deep link generation on your subscriber support (as described in the *SIA 19.2.1 Products: AMC-X Overview and Deployment Design* and *SIA 19.2.1 Products: Installation Using AMC-X* manuals), you can generate a deep link connection for your SIA Remote service subscribers from your support portal. The example that follows generates a deep link connection for a subscriber called **joe.coffee**:

```
SSM> POST /account/joe.coffee/spson/mdm_deeplink
{
"link":"https://akacloud.spsonstatic-preprod.akamaiapis.net/apps?ap
idomain=eng7.spsoncas-preprod.akamaiapis.net&regid=4f5d34ae-11ab-4e
45-9677-ced7357d9246&entcode=3-9053589c-3f2e-486f-aad0-9bb3325d0985
"
}
```

Rotating the Entitlement Code for Deep Links

Rotating the entitlement code for a deep link performs the following operations:

- Generates and saves a new entitlement code that gets sent to the CAS.
- Generates a new deep link using the new entitlement code.

Rotating entitlement codes (thereby generating new deep links) reduces the risk of a compromised or terminated subscriber account accessing your service. We recommend regularly rotating the entitlement codes for subscriber accounts to ensure your service data is safe.

NOTE: Because time-bound and non-expiring deep links have different entitlement codes, rotate expiring and non-expiring entitlement codes independently from one another.

When you rotate the entitlement code and generate a new deep link, the previous deep link value is no longer valid. Any attempt to register a remote device with the old deep link results in an "Activation information invalid" message in the SIA Remote application.

This section describes how to perform the following tasks:

- Rotate the entitlement code for time-bound deep links
- Rotate the entitlement code for non-expiring deep links

Rotate the Entitlement Code for Time-bound Deep Links

Use the following **PUT** API to rotate the time-bound deep link entitlement code for a subscriber account; rotating the entitlement code revokes any existing time-bound (type2) deep link:

```
SSM> PUT /account/<accountId>/spson/entitlement
```

Replace the ***<accountId>*** argument with the name of the subscriber account for which to rotate the entitlement code.

The example that follows rotates the entitlement code for a subscriber account called **joe.coffee**:

```
SSM> PUT /account/joe.coffee/spson/entitlement
```

Rotate the Entitlement Code for Non-expiring Deep Links

Use the following **PUT** API to rotate the non-expiring deep link entitlement code for a subscriber account; rotating the MDM entitlement code revokes any existing non-expiring (type3) deep link.

```
SSM> PUT /account/<accountId>/spson/mdm_entitlement
```

Replace the ***<accountId>*** argument with the name of the subscriber account for which to rotate the entitlement code.

The example that follows rotates the entitlement code and generates a new non-expiring deep link for a subscriber called **joe.coffee**:

```
SSM> PUT /account/joe.coffee/spson/mdm_entitlement
```

Renaming an Akamai SIA Remote Client Roaming Device

Use the following **PUT** API to modify the device name associated with a device that has the Akamai SIA Remote client enabled:

```
SSM> PUT /account/<accountId>/spson/devices/<device_uuid>
{
     "name": "Jane iphone"
}
```

Setting Internal Domain Access for the Devices that are Associated with a Subscriber Account

Before enabling internal domain access for an individual subscriber account, use the **GET /account/<*accountId*>/spson** API to determine whether internal domains are already set for a subscriber account. See the "Retrieving SIA Remote Service Configuration Details for a Subscriber Account" section for details on how to use the **GET /account/<*accountId*>/spson** API to retrieve a list of internal domains the devices associated with a subscriber account can access.

Use the following **PATCH** API to set or replace the internal domain access for the devices that are associated with a specific subscriber account:

```
SSM> PATCH /account/<accountId>/spson
{
    "internal-domains": ["<domain>, …"]
}
```

Replace the **<*accountId*>** argument with the name of the subscriber account for which you want to set internal domain access. Use the **"Internal-domains"** field to specify a list of internal or local domains devices can access, where the **<*domain*>** argument specifies a domain. To specify multiple domains, separate each domain with a comma as shown in the following example:

```
"internal-domains":[domain-1.com, domain-2.com]
```

Caution! When using the **PATCH /account/<*accountId*>/spson** API, be aware that setting the **"internal-domains"** construct replaces the enture list of internal domains the devices can access. To add access to additional domains, the **"internal-domains"** field needs to include all internal domains the subscriber needs to access.

For more information about using the **PATCH /account/<*accountId*>/spson** API to update internal domain access for a subscriber account, see the “Update SIA Remote Options for a Subscriber Account” section in one of the following manuals, depending on your service:

- *SIA SMB SSM API Reference*, Release 19.2.1
- *SIA Consumer SSM API Reference*, Release 19.2.1

The example that follows sets the internal domain access for a subscriber account called **joe.coffee**. This example enables access to one internal domain (**akamai.com**):

```
SSM> PATCH/account/joe.coffee/spson
{
    "internal-domains": ["akamai.com"]
}
```

Retrieving SIA Remote Service Configuration Details for a Subscriber Account

Use the **GET /account/<*accountId*>/spson** API to display SIA Remote service configuration details for a subscriber account. Replace the **<*accountId*>** argument with the name of the subscriber account for which to display configuration details.

- The maximum number of SIA Remote service devices the subscriber account can have.
- The global entitlement UUID associated with the subscriber account
- The entitlement UUID for time-bound the deep link.
- Whether non-expiring deep links are enabled for the specified subscriber account.
- If non-expiring deep links are enabled for the account, the output includes entitlement UUID for the non-expiring deep link.
- A list of the internal and local domains the associated subscriber devices can access.

  NOTE: Before enabling internal domain access for an individual subscriber account, use the **GET /account/<*accountId*>/spson** API to determine whether internal domains are already set for a subscriber account.

The example that follows shows sample output from the **GET /account/<*accountId*>/spson** API. This examples shows SIA Remote configuration information for a subscriber account called **joe.coffee**:

```
SSM> GET /account/joe.coffee/spson
{
    "allowed-devices": 3,
    "customer-id": "772a0c7b-99a3-40f0-b63f-c12c26bb6584",
    "entitlement": "c7866937-3699-4006-970a-0be462c47bc2",
    "internal-domains": [
        "first-domain.com",
        "second-domain.com"

    ]
}
```

Displaying the Status of All SIA Remote Client-enabled Roaming Devices for a Specific Subscriber

Use the following **GET** API to display detailed information for the roaming devices associated with a specific subscriber account:

```
SSM> GET /account/<accountId>/logical-device
```

Replace the

The command output displays the following details for each roaming device:

- The name of the device.
- The device UUID.
- Device Model, OS Type, and Version information.
- The subscriber's full name.
- Whether SPSON is currently enabled on the device.
- The application version installed on the device.
- The time at which the device was seen (active) on the network).

Displaying Generic Akamai SIA Remote Client Errors

Use the following **GET** command to display the status of an Akamai SIA Remote client and the client application:

```
SSM> GET /account/<accountId>/spson/devices/<device-uuid>
```

The table that follows describes the possible status types that can be included in the API output:

unenrolled	The subscriber has SPSON service but the device is not enrolled yet.
enrolled	The device is enrolled in the Akamai SIA Remote client service.
unregistered	The device is not registered in the SIA Remote service.
registered	The device is registered for Akamai SIA Remote client service.
enabled	The Akamai SIA Remote client is enabled on the device.
disabled	The Akamai SIA Remote client is disabled on the device.
on-net	The device is detected to be on the service provider network.
off-net	The device is detected to be off the service provider network.
stopped	The Akamai SIA Remote client application is stopped on the subscriber device.
resumed	The Akamai SIA Remote client application resumed after stopping on a device.
crashed	The Akamai SIA Remote client application crashed on the subscriber device.
uninstalled	The Akamai SIA Remote client application is uninstalled from the subscriber device.

Deleting a Roaming Akamai SIA Remote Client Device

Use the following **DELETE** API to unenroll a specific device from the Akamai SIA Remote client support:

```
SSM> DELETE /account/<accountId>/spson/devices/<device-uuid>
```

Deprovisioning the Akamai SIA Remote Client Support for a Subscriber Account

Deprovisioning the Akamai SIA Remote client support for a specific subscriber account involves:

1. Use the following **DELETE** API to deactivate the Off Network Protection service for the subscriber account:

```
SSM> DELETE /control/sub123/service/spson
```

2. Use the following **DELETE** API to remove the SPSON object from the subscriber account:

```
SSM> DELETE /account/sub123/spson
```

Chapter 6: Using the Service Check Application

The SIA SMB and SIA Consumer services support a Service Check application that enables non-privileged users (for example, as an ISP agent) to perform the following tasks:

- Determine whether an end-user device is pointing at the correct service.
- Access additional information useful for troubleshooting.

For example, you can use the Service Check application to determine which service tier a customer has access to and troubleshoot issues related to subscribers not pointing to the correct DNS infrastructure.

This appendix contains the following sections:

Prerequisites

Ensure your Akamai DNSi/SIA Applications Portal deployment meets following prerequisites before using the Service Check application:

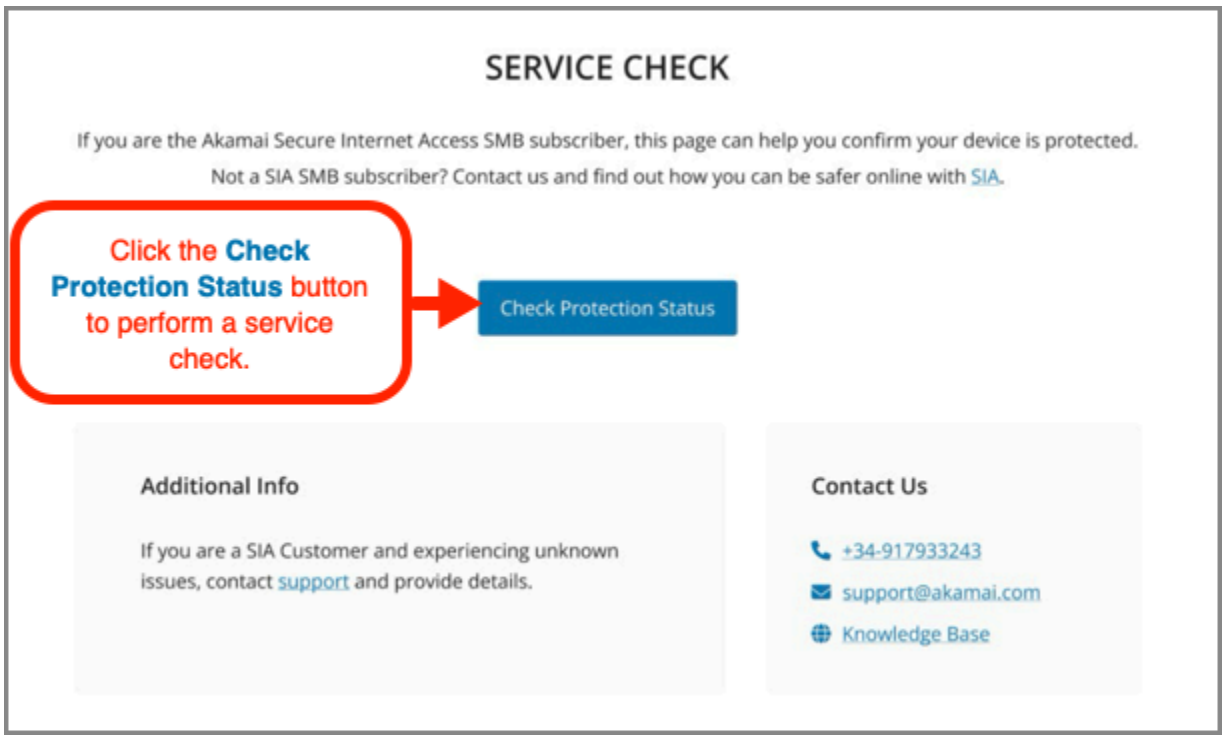
- Your service has a publicly accessible generic domain that is resolvable from anywhere (for example, *ServiceCheck.ServiceProvider.com*).
- The Service Check application is enabled on your Akamai DNSi/SIA Applications Portal deployment. See the "Enabling the Service Check Application in your Akamai DNSi/SIA Applications Portal Deployment" section for instructions on enabling the Service Check application.
- The landing pages a subscriber can encounter upon performing a service check are customized on your Akamai DNSi/SIA Applications Portal. The Service Check application supports the display of the following landing pages:
    - The device is protected
    - Device onboarding is incomplete
    - The device is not protected

See the "Customizing Service Check Landing Pages" section for details on customizing the service check landing pages.

Performing a Service Check

The following steps describe the process for performing a service check for an SIA SMB or SIA Consumer subscriber:

1. A user calls the ISP support with a question about a website being unexpectedly blocked or available despite Subscriber Portal account settings.
2. An ISP support administrator instructs the user to make a query against the publicly available Service Check URL (`https://{sportal}/service-check`) to verify service connectivity.
3. The user opens the URL and sees a landing page similar to the page in the example that follows. The user clicks the **Check Protection Status** button to perform a service check:



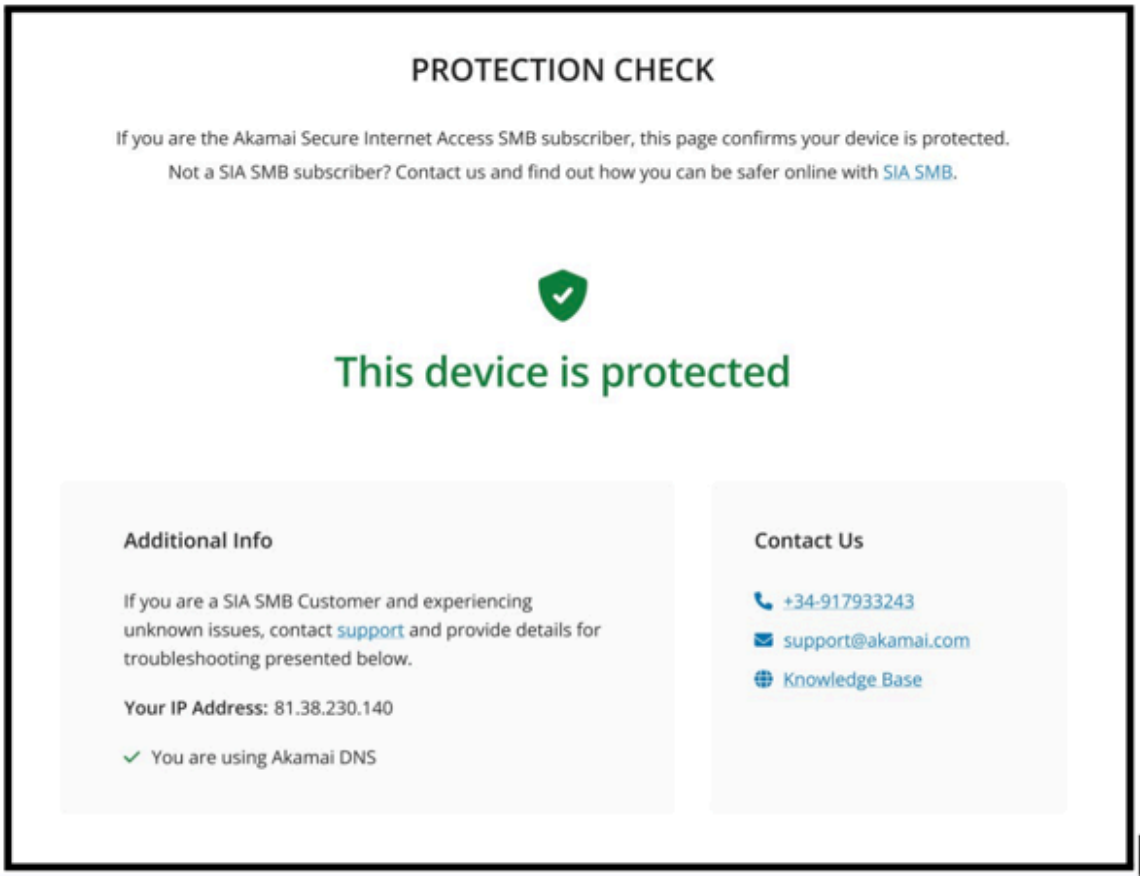
4. The service check responds with the user's IP address and configuration status (whether the service check is successful, incomplete, or fails). The user reports the results to the ISP support contact.
5. The ISP support contact uses the information to take further steps on the user or ISP side.

The sections that follow provide examples of each landing page and what the results mean.

Service Check is Successful

If the user's DNS is pointing to the designated SIA platform and the designated view, the service check succeeds and the device user is redirected to the following landing page:

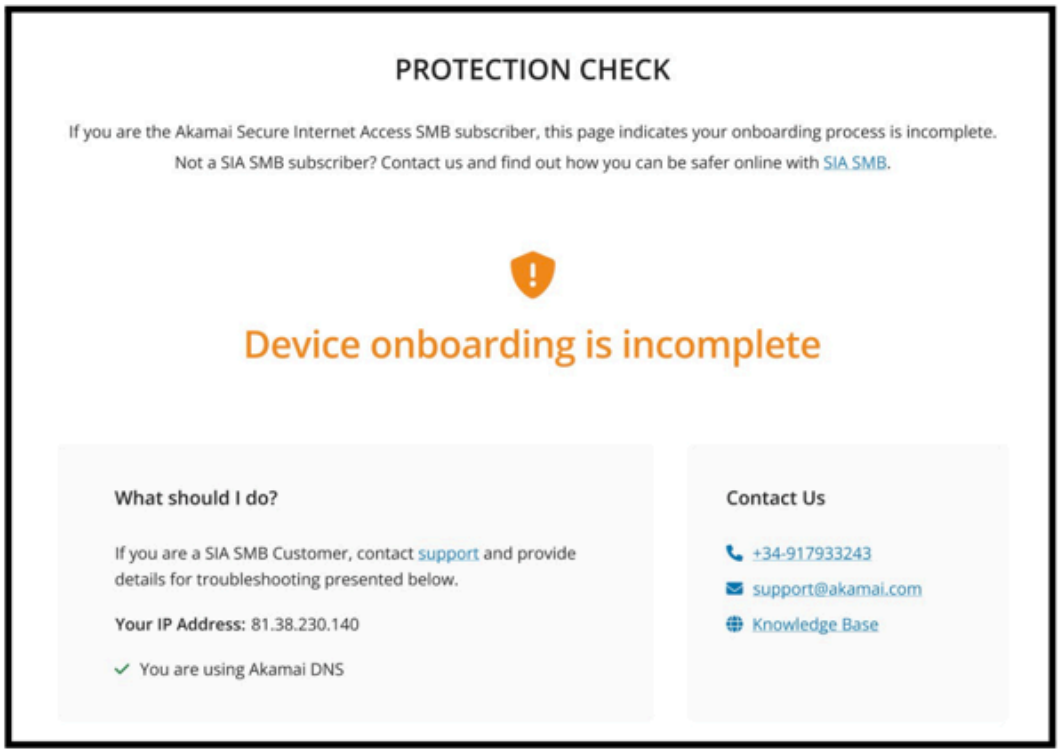
Figure 6-1: Example Landing Page When the Service Check is Successful



Service Check is Incomplete

If the device user's DNS points to the designated SIA platform but not to a designated view (for example, if the subscriber device is pointing to the world view), the service check cannot complete and the device user is redirected to the following landing page:

Figure 6-2: Example Landing Page When the Service Check is Incomplete



Service Check Failures

A service check fails when one of the following circumstances occurs:

- The Service Check redirect cannot resolve using the carrier's DNS.
- The device user's DNS does not point to the designated SIA platform.

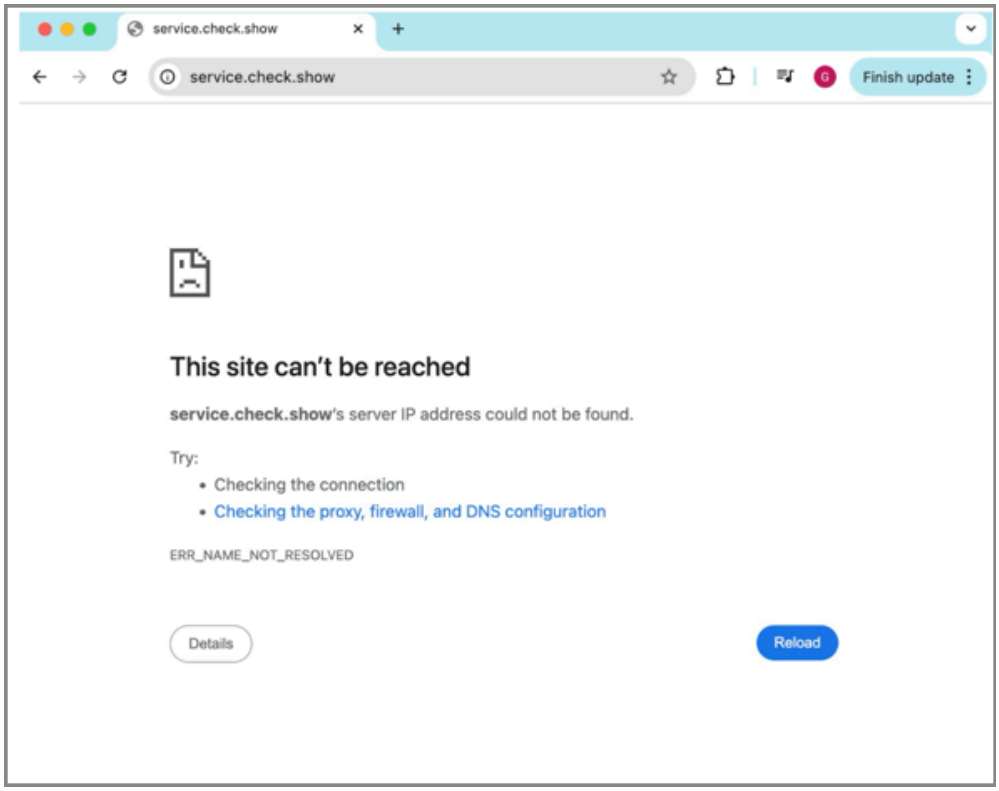
The user sees a different page, depending on the reason for the failure.

If the Service Check redirect cannot resolve using the carrier DNS, the request fails due to an NXDOMAIN error and the user sees a standard browser error page.

NOTE: The error page is different for each browser.

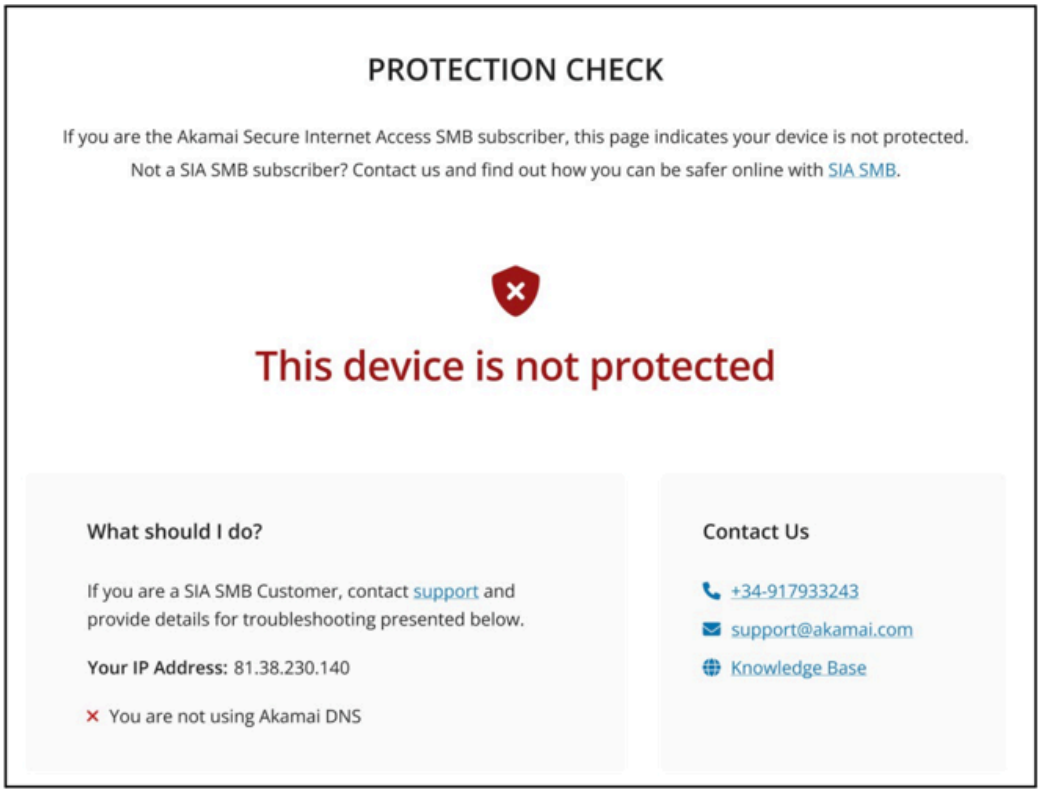
The following example shows the browser error page Chrome:

Figure 6-3: Example Chrome Browser Error When the Service Check Fails Due to an NXDOMAIN error



If the failure occurs because a device user's DNS does not point to the designated SIA platform, the device user is redirected to the landing page that indicates the device is not protected. The example that follows shows the message a user sees when the service check is unsuccessful:

Figure 6-4: Example Landing Page When the Service Check Fails Because the DNS Does Not Point to the Designated SIA Platform



Enabling and Disabling Service Check Application

At the initial setup, the `business.yaml` or `residential.yaml` configuration file (depending on which service you are supporting) determines whether the Service Check application is enabled or disabled for the Subscriber Portal. If the Service Check domain is not specified in the appropriate YAML configuration, the related policies are not provisioned and the Service Check application is not enabled.

To enable the Service Check application in your SIA SMB or SIA Consumer service, you need to add the Service Check domain to the `virtual-domain-service-check` field in your `business.yaml` or `residential.yaml` configuration file.

To turn the Service Check application off after it has been enabled, use one of the following methods to remove the Service Check domain from the settings database in PGaaS:

- To use a simple command to remove the Service Check domain from the settings database, access the Levant Command Line Interface (CLI) and enter the **del business:virtual-domain-service-check** command as demonstrated in the following example:

```
levant-cli> del business:virtual-domain-service-check
```

- To manually remove the Service Check domain from the the `business.yaml` or `residential.yaml` configuration file, perform the following steps:

  a. In the `business.yaml` or `residential.yaml`configuration file under */var/nom/nom-config-hub/design/application-settings*, locate the `virtual-domain-service-check` field and replace the value with **~** as demonstrated in the following example:

```
global:
  business:
       ...
       virtual-domain-service-check: ~
```

  b. In the nom-config-hub nom-tell command channel, enter the **deployments.deploy** command as demonstrated in the following example to trigger a redeployment:

```
# /usr/local/nom/sbin/nom-tell nom-config-hub deployments.deploy
name=<deploymentFileName>
```

  Replace the **<*deploymentFileName*>** argument with the name of your deployment.

Customizing Service Check

There are two HTML pages that can be customized.

- The Service Check landing page in the Subscriber Portal.
- The Service Check response template on the Proxy Server.

You need to customize the Service Check landing page because the landing page includes several dummy links, placeholders for contact details and an Akamai logo. Use an HTML editor to customize the Service Check landing page, which is an HTML file. In other words, customize the Service Check landing page in the same way as the rest of the Subscriber Portal.

Keep the following recommendations in mind when customizing Service Check:

- Place the customized Service Check landing page in the same resource location as other customized resources to ensure those resources are not overwritten during upgrades. For example, if SIA Business customizations are located in the path */usr/local/nom/share/sportal/secure-business*, place your Service Check customizations in the following path: */usr/local/nom/share/sportal/service-check*.
- To ease the customization process, do not minimize the Service Check page size.
- Because the Service Check application is small, only fonts and favicons load as separate assets. The rest of the icons, along with styles and a small JS snippet, are embedded in the index.html file.

Response Template for the Proxy Server

The `service-check-template.yaml` file is the response template for the Proxy server and is loaded into the settings database upon deployment (like the other configurations in the `/var/nom/nom-config-hub/design/application-settings` path). In the `service-check-template.yaml` file, the content field contains all the HTML, styles and JS of the response body that an ISP can change as desired.

When modifying the `service-check-template.yaml` file, pay attention to the substituted values for the {{policy}} and {{client-address}} variables. The Proxy server replaces those variables with actual values when preparing an HTTP response.

After making changes to the page, run the following command to commit the changes:

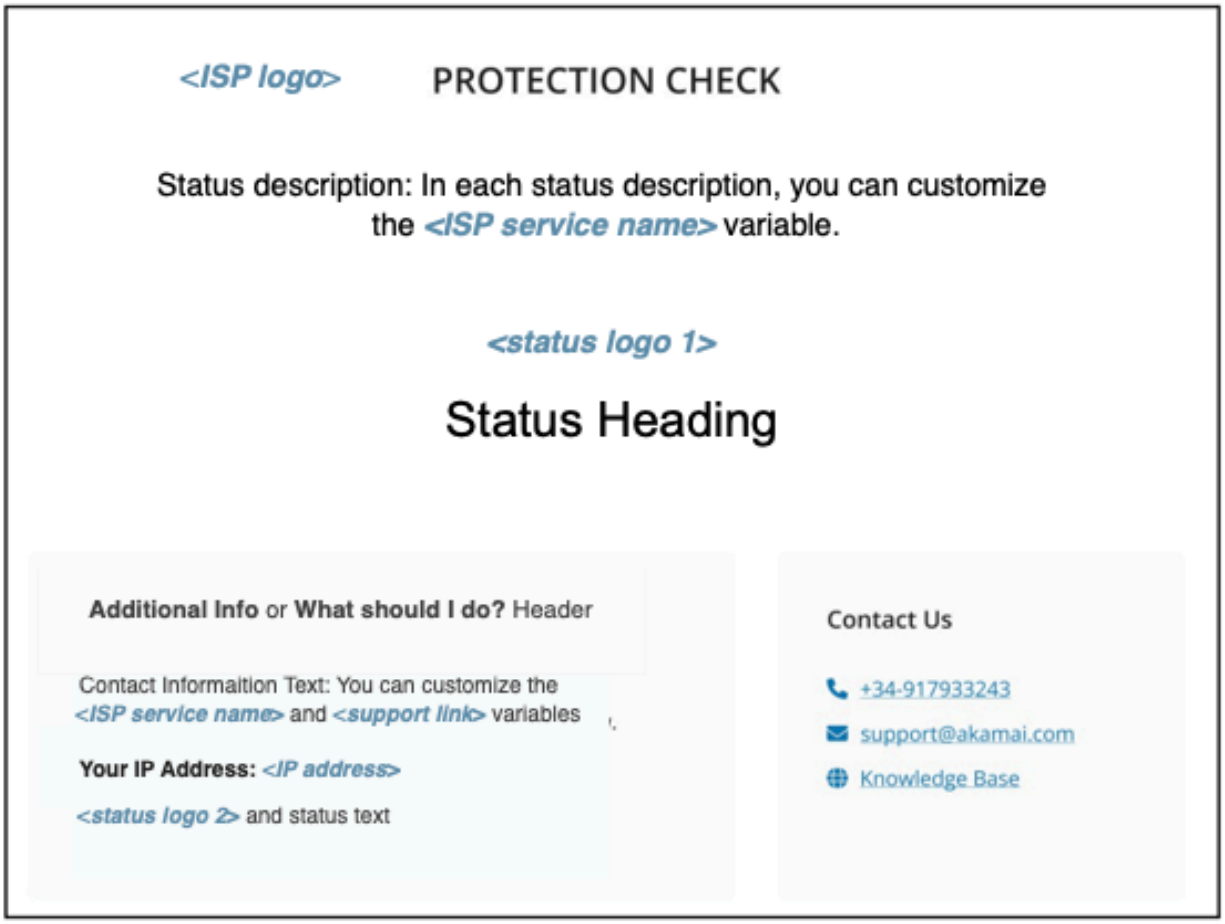
```
# /usr/local/nom/sbin/nom-tell nom-config-hub deployments.deploy
name=<deploymentFileName>
```

Replace the **<*deploymentFileName*>** argument with the name of your deployment

Customizing Service Check Landing Pages

Before using the Service Check application, you need to customize the landing pages the subscribers can encounter when a service check is successful or incomplete.

While each status has its own landing page, the general format of the landing page is similar and contains customizable variables. The figure that follows shows the basic format of a landing page, where the customizable variables are called out as blue arguments in brackets (for example *<ISP logo>*):



The table that follows describes the customizable variables you can set for each landing page:

Table 6-1: Landing Page Variable Descriptions

Variable	Description
<ISP Logo>	If desired, you can add your ISP Logo to the landing page header.
<ISP service name>	Specifies the name of your ISP service. You can include your ISP service name in the following locations: ● In the status description text that appears above the *<status logo 1>* variable. The text includes your ISP service name any place the *<ISP service name>* variable appears. ● In the contact information text that appears in the following locations, depending on the landing page you are configuring: ○ On the "device is protected" landing page, includes the specified ISP service name in the contact information text that appears under the "Additional Info" header. ○ On the "device onboarding is incomplete" landing page, includes the specified ISP service name in the contact information text that appears under the "What should I do?" header. **NOTE:** You can hot-link any of the variables to point to a web page that provides more information about the service you are providing.
<status logo 1>	Includes the specified status logo above the status heading. Each landing page has its own status logo.
<support link>	Specifies a hot link to your ISP support web page in the following locations, depending on the landing page you are configuring: ○ On the "device is protected" landing page, the support link is located in the contact information text that appears under the "Additional Info" header. ○ On the "device onboarding is incomplete" landing page, the support link is located in the contact information text that appears under the "What should I do?" header.
<IP address>	Specifies the subscriber's public facing IP address. The address appears in the following locations, depending on the landing page you are configuring: ● On the "device is protected" landing page, the public facing IP address appears under the "Additional Info" header. ● On the "device onboarding is incomplete" landing page, the public facing IP address appears under the "What should I do?" header.

<status logo 2>	Includes a status logo in the following locations, depending on the landing page you are configuring: • On the “device is protected” landing page, the desired “protected” logo exists next to the status text that appears under the “Additional Info” header. • On the “device onboarding is incomplete” landing page, the desired “incomplete” logo exists next to the status text that appears under the “What should I do?” header.

Enabling the Service Check Application in your Akamai DNSi/SIA Applications Portal Deployment

In your `business.yaml` or `residential.yaml` file, locate the `virtual-domain-service-check:` field and add your service check virtual domain as demonstrated in the example that follows.

NOTE: The virtual domain needs to be a subdomain of the carrier owned domain.

The following example shows partial output from the `business.yaml` file, where the `virtual-domain-service-check:` field is highlighted:

```
[root@levant conf]# cat business.yaml
global:
  business:
    # The 'sites-allowed' object is used to define a global allow
list-group.
    # Specify an array of list-group names defined in
"nom-list-manager" to enable this feature.
    # Local List and Local List Group object names are subject to
the provisioned name transformation
    # executed by "nom-list-manager". See the "nom-list-manager"
documentation for its provisioning behavior.
   # Example:
   # sites-allowed:
   #   - gisp_sites_allowed
   virtual-domain-beacon-event: bb.demo.delivery
   virtual-domain-device-check: dc.demo.delivery
   virtual-domain-device-register: dr.demo.delivery
   virtual-domain-service-check: <virtualDomain>
   virtual-domain-asset: asset.demo.delivery
   proxy-group-mapping:
   ...
```

After defining the virtual domain in the `business.yaml` or `residential.yaml` file, Levant provisions the the following objects to CacheServe and the Proxy server:

- The frontend Service Check application
- The Service Check endpoint
- The Service Check name lists
- The bus-gen-vir policy
- The response template for Service Check
- The SIA Remote connect-port-policy policy (in deployments supporting the SIA Remote service).

NOTE: If you do not specify the ***<virtualDomain>*** field in the `business.yaml` or `residential.yaml` file, none of the objects will be rendered.

Verifying Service Check Functionality

This section provides details on how to verify and troubleshoot your Service Check configuration.

After initially configuring and enabling the Service Check application in your deployment, we recommend performing the following tests:

- Ensure the device user is redirected to the "Service Check is successful" landing page user's DNS is pointing to the designated SIA platform and the designated view,,
- Ensure the device user is redirected to the "Service Check is incomplete" landing page when DNS resolution on the device is configured correctly but the request is caught by the world view.
- Ensure the device user is redirected to a standard browser error page when the Service Check redirect cannot resolve using the carrier's DNS.
- Ensure the device user is redirected to the "Device is not protected" landing page when the device user's DNS does not point to the designated SIA platform.

The sections that follow describe additional verification and troubleshooting tasks for the following Service Check components:

- The fontend Service Check application
- The Service Check endpoint
- The Service Check name lists on CacheServe
- The Service Check name lists on the Proxy Server
- The Subscriber Portal refers to the Service Check domain
- The bus-gen-vir policy
- The Proxy Server
- The SIA Remote connect-port-policy policy (for deployments that have the SIA Remote service enabled).

Verify Frontend Service Check Application Functionality

Perform the following tasks to verify the Service Check frontend application is properly configured:

- Verify the Service Check frontend application exists with its own /service-check route on your Subscriber Portal.
- Verify the Service Check home page explains the goal of the Service Check to the user and provides a button or link that initiates a service check and determines a user's protection status. The complete flow with mockups is described in the "Performing a Service Check" section.
- Verify the Service Check frontend application has an HTML web page that contains all the necessary static resources (styles, fonts, images, and so forth), along with the JS code that is responsible for retrieving the Service Check domain value from the Subscriber Portal backend (see the "Verify the Service Check Endpoint" section).

NOTE: If you are simultaneously supporting the SIA Business and SIA Consumer services, the Service Check domain values can be different for SIA Business and SIA Consumer.

If required, you can add a second Service Check page to the Subscriber Portal. See the "Customizing your Service Check Application" section for details.

Verify the Service Check Endpoint

Use the following **GET** API to verify the Service Check domain that is defined in the settings database.

```
curl https://<subscriberPortalDomain>/sportal/service-check
{
    "business": "service-check.nomdebug.com",
    "residential": "service-check.nomdebug.com"
}
```

Replace the **<*subscriberPortalDomain*>** argument with your Subscriber Portal domain name.

NOTE: If you are simultaneously supporting the SIA Business and SIA Consumer services, the API response includes the domain for both services.

Verify the Service Check Name Lists on CacheServe

On the CacheServe, verify the **bus-vir-sc** name list exists and is included in the **bus-vir-sc** list group, which contains the lists of virtual domains for the SIA Business service:

1. Access the nom-tell command channel for your CacheServe:

```
# /usr/local/nom/sbin/nom-tell cacheserve
nom-tell 19.2.1.0, interactive mode
cacheserve>
```

1. Use the **name-list.get** command to verify the **bus-vir-sc** name list exists as demonstrated in the following example:

```
cacheserve> name-list.get name=bus-vir-sc layer=*
response:
{
    type => 'name-list.get'
    name => 'bus-vir-sc'
    match-type => 'exact-or-www'
    count => '1'
}
```

2. Use the **name-group.get** command to verify the **bus-gen-vir** list group contains the **bus-vir-sc** name list as demonstrated in the following example; the **bus-vir-sc** name list is highlighted in the output

```
cacheserve> name-group.get name=bus-gen-vir layer=*
response:
{
    type => 'name-group.get'
    name => 'bus-gen-vir'
    lists => ('bus-vir-sc' 'bus-vir-be' 'bus-vir-dc'
    'bus-vir-dr' 'bus-vir-by' 'bus-vir-as')
}
```

Verify the Service Check Name Lists on the Proxy Server

On the Proxy Server, verify the **bus-vir-sc** name list exists and is included in the **bus-vir-sc** list group, which contains the lists of virtual domains for the SIA Business service:

2. Access the nom-tell command channel for the Proxy server:

```
# /usr/local/nom/sbin/nom-tell nom-proxy
nom-tell 19.2.1.0, interactive mode
nom-proxy>
```

3. Use the **name-list.get** command to verify the **bus-vir-sc** name list exists as demonstrated in the following example:

```
nom-proxy> list.get name=bus-vir-sc layer=*
response:
{
    type => 'list.get'
    name => 'bus-vir-sc'
    match-type => 'exact-or-www'
    count => '1'
}
```

4. Use the **name-group.get** command to verify the **bus-gen-vir** list group contains the **bus-vir-sc** name list as demonstrated in the following example; the **bus-vir-sc** name list is highlighted in the output:

```
nom-proxy> list-group.get name=bus-gen-vir layer=*
response:
{
    type => 'list-group.get'
    name => 'bus-gen-vir'
    lists => ('bus-vir-sc' 'bus-vir-be' 'bus-vir-dc'
'bus-vir-dr' 'bus-vir-by' 'bus-vir-as')
}
```

Verify the Subscriber Portal Refers to the Service Check Domain

The Subscriber Portal uses the Service Check domain from the `business.yaml` or `residential.yaml` file to ensure the Subscriber Portal redirects users to the correct URL. This is why Levant also provisions both of those values (if present) to sportal. The Subscriber Portal makes the values available in the SSM API (see the "Verify the Service Check Endpoint" section for more details).

```
sportal> portal.get layer=*
response:
{
    type => 'portal.get'
    multiple-emails-enabled => 'false'
    service-check-domain => {
        business => 'service-check.demo.delivery
        residential => 'service-check.demo.delivery
    }
}
```

Verify the bus-gen-vir Policy Returns the Correct Response When a Subscriber View is Matched and the Service Check is Successful

The `bus-gen-vir` policy includes all rules related to the handling of virtual domains and is bound to all SIA Business subscribers. Use the **`policy.get name=bus-gen-vir layer=*`** command to verify the policy contains the rule ensuring that when a subscriber view is matched, the redirect to the Service Check virtual domain returns the `service-check-template` response body (which indicates the Service Check is successful).

The example that follows demonstrates what the Service Check rule looks like in the **`policy.get name=bus-gen-vir layer=*`** command output. This example shows partial output and the service check rule is highlighted:

```
nom-proxy> policy.get name=bus-gen-vir layer=*
Response:
{
   type => 'policy.get'
    name => 'bus-gen-vir'
    rules => (
        {
            condition => (('and' (('list' ('bus-vir-sc')) ('not'
(('request-method' ('connect')))))))
            action => (
                (
                    'return-response'
                    {
                        status-code => '200'
                        body => 'service-check-template'
                    }
```

```
                    )
                )
            }
        {
            condition => (('list' ('bus-vir-be')))
            action => (
                (
                    'return-response'
                    {
                        status-code => '200'
                    }
                )
            )
        }
    ...
    )
}
```

NOTE: A condition about the connect method is included for deployments that have the SIA Remote service enabled (see the "Verify the SIA Remote connect-port-policy Policy Contains a Service Check Exception" section).

To learn more about the response body, see the "Understanding the Response Template for Service Check" section.

Verify the Proxy Server Returns the Correct Response When Service Check is Incomplete

When a service check is incomplete, a Proxy server binding returns a 200 status code and the `service-check-template` response body (which indicates the "Service Check is Incomplete"). The binding catches all requests to the Service Check virtual domain that are not caught by more specific selectors. The binding returns a response body that includes values for the client address and policy name. In the Proxy server nom-tell command channel, use the **server.get layer=*** command to verify the Proxy server contains this binding.

The example that follows shows output from the **server.get layer=*** command with the `service-check-template` binding highlighted:

```
nom-proxy> server.get layer=*
response:
{
    type => 'server.get'
    listen-on-matching => (
        {
            patterns => ('10.42.15.161/32')
            ports => (('80' 'http') ('443' 'https'))
        }
    )
    policy-bindings => (
```

```
         {
             priority => '10000'
             condition => (('list-group' ('res-gen-vir')))
            action => ('deny')
        }
        {
            priority => '9000'
            condition => (('list' ('bus-vir-sc')))
            action => (
                (
                    'return-response'
                    {
                        status-code => '200'
                        body => 'service-check-template'
                    }
                )
            )
        }
        {
            priority => '10000'
            condition => (('list-group' ('bus-gen-vir')))
            action => ('deny')
        }
        {
            priority => '9000'
            condition => (('list' ('res-vir-sc')))
            action => (
                (
                    'return-response'
                    {
                        status-code => '200'
                        body => 'service-check-template'
                    }
                )
            )
        }
    )
}
```

Verify the SIA Remote connect-port-policy Policy Contains a Service Check Exception

If you have SIA Remote configured in your system, you need to verify the `connect-port-policy` policy contains an exception that allows Service Check requests.

The `connect-port-policy` policy typically prevents a request from reaching the Proxy server if the requesting device does not have the SIA Remote application installed. With the Service Check application, the `connect-port-policy` policy needs to include an exception that allows any domain request matching the Service Check list. This exception prevents the Service Check application from breaking when Service Check does not receive a response. As long as the `connect-port-policy` allows the request.

Use the **policy.get name=connect-port-policy layer=*** command to verify the `connect-port-policy` conditions contain the exception allowing requests matching the Service Check list. The example that follows shows sample output from the **policy.get name=connect-port-policy layer=*** command:

```
nom-proxy> policy.get name=connect-port-policy layer=*
response:
{
    type => 'policy.get'
    name => 'connect-port-policy'
    rules => (
        {
            condition => (
('not' (('or' (('and' (('proxy-port' ('443')) ('request-method'
('connect'))))
('request-url' 'https://test3-gkasatki.demo.delivery/spsonapi/v1')
('list' ('bus-vir-sc''))
('proxy-port' ('80'))))))
)
            action => ('deny')
        }
    )
}
```

Troubleshooting

This section describes possible issues you can encounter when supporting the Service Check application and provides information on how to troubleshoot each issue.

Issue:	Although a device is protected, the Service Check response incorrectly indicates that device is not protected.
Recommendation:	Verify that the Service Check virtual domain is the same domain that is configured in the Levant YAML files.

Issue:	Although a device is not protected, the Service Check response incorrectly indicates that device is protected.
Recommendation:	Verify that the Service Check virtual domain is configured correctly and use nom-tell queries in CacheServe and the Proxy server to trace and determine why an unprotected device receives a 200 OK response when requesting the virtual domain.

Issue:	The device status displays as protected, even though the Service Check response displays an IP address that does not correspond to any view selector.
Recommendation:	A client is considered protected when the request from the client is matched to a non-catch-all view-selector. Be aware that this match may be based on the device ID or other aspects of the request. In this case the IP is useless for debugging.

Chapter 7: Configuring CAS Agent Client Version Changes

Use the following steps to update one or more client versions for a CAS agent:

1. On your Levant node, create a `global_engine.yaml` file that references your CAS agent client versions as demonstrated in the example that follows:

```
# vi global_engine.yaml
global:
  engine_direct:
    cas-agent-client-versions:
      name: Cas agent client versions
      engine_objects:
        cas-agent: >
          [
            {
              "type": "client-version",
              "name": "<CAS-version>",
              "client-type": "sps"
            }
            ...
          ]
```

   Replace the **<CAS-version>** argument with the client version to update to.

   The example that follows shows a `global_engine.yaml` file that `global_engine.yaml` file the references one client version :

```
# vi global_engine.yaml
global:
  engine_direct:
    cas-agent-client-versions:
      name: Cas agent client versions
      engine_objects:
        cas-agent: >
          [
            {
              "type": "client-version",
              "name": "ios-19.1.2.0",
              "client-type": "sps"
            }
          ]
```

The example that follows shows a `global_engine.yaml` file that references multiple client versions:

```
# vi global_engine.yaml
global:
  engine_direct:
    cas-agent-client-versions:
      name: Cas agent client versions
      engine_objects:
        cas-agent: >
          [
           {
             "type": "client-version",
             "name": "android-1.2.2",
             "client-type": "sps"
           },
           {
             "type": "client-version",
             "name": "ios-1.2.2",
             "client-type": "sps"
           },
           {
             "type": "client-version",
             "name": "macos-2.14",
             "client-type": "sps"
           },
           {
             "type": "client-version",
             "name": "windows-2.14",
             "client-type": "sps"
           }
          ]
```

2. Access the Kafka provisioning watch tool from your Jump Host Tools as described in the "How to Access the Kafka Provisioning Watch Tool" section. You need to use the Kafka Provisioning Watch tool after uploading the YAML file as described in Step 3.

3. Access the `config-hub` nom-tell command channel (CC) and use the **`deployments.deploy`** command to deploy the YAML file as demonstrated in the following example:

```
[root@nom-config-hub ~]# /usr/local/nom/sbin/nom-tell nom-config-hub
deployments.deploy name=<deploymentName>
```

   Replace the **<*deploymentName*>** argument with the name of your deployment.

4. In your Kafka Provisioning Watch tool, ensure the YAML file is uploaded as demonstrated in the following example; in this example, the Kafka Provisioning Watch tool displays a record confirming that Levant uploaded YAML file called `ios-19.1.2.0`:

```
2022-08-30 15:29:12.095000
nom-provisioning-levant-schema-casagent-p1 levant add 0
{'name': 'ios-19.1.2.0', 'type': 'client-version'}
```

5. Access the `cas-agent nom-tell` command channel for your SSM host server and enter the **client-version.mget layer=*** command to verify the CAS agent version you added appears in the command response as demonstrated in the example that follows:

```
cas-agent> client-version.mget layer=*
response:

{
    type => 'client-version.mget'
    client-type => 'sps'
    name => 'ios-19.1.2.0'
}
```

   NOTE: See the "How to Access the cas-agent nom-tell Command Channel for an SSM node" section for details on how to access the CAS agent nom-tell command channel for an SSM node.

6. Enter the **client-version.status** command and verify the client version `status` field says `'ready to send'` as demonstrated in the following example:

```
cas-agent> client-version.status
response:
{
    type => 'client-version.status'
    status => 'ready to send'
}
```

7. Enter the **client-version.send** command as demonstrated in the following example to send the client version configuration to CAS:

```
cas-agent> client-version.send
response:
{
    type => 'client-version.send'
}
```

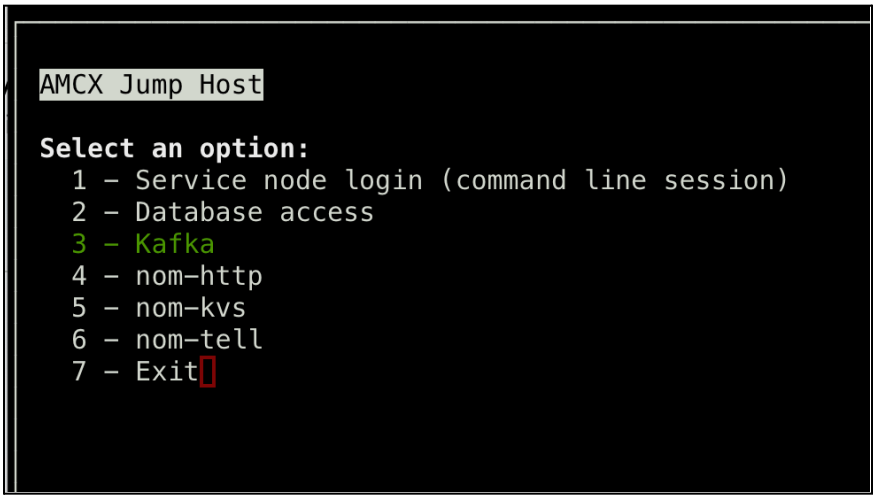
8. Enter the **client-version.status** command and verify the client version `status` field indicates the client version was `'successfully sent'` to CAS as demonstrated in the following example:

```
cas-agent> client-version.status
response:
{
    type => 'client-version.status'
    status => 'successfully sent'
}
```

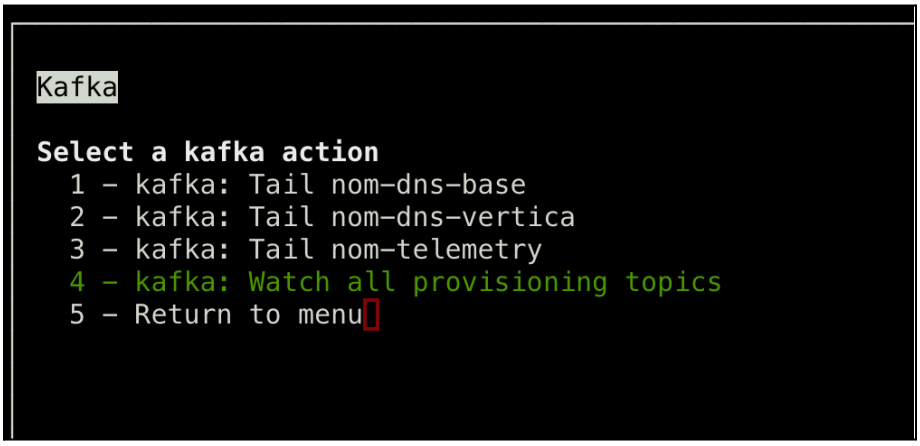
How to Access the Kafka Provisioning Watch Tool

Use the following steps to access the service command channel:

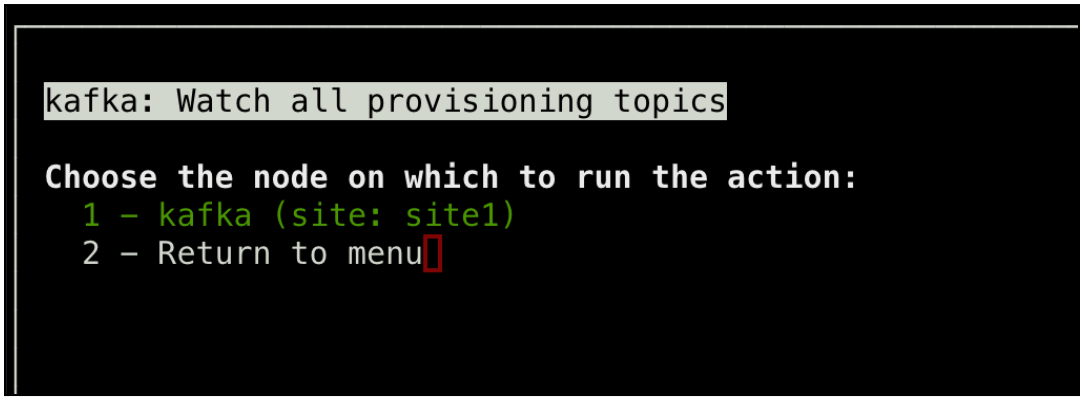
1. In your AMCX Jump Host tool, select the **Kafka** option (option **3**) to open the Kafka action selector, as demonstrated in the following example:



2. In the Kafka action selector, select the **kafka: Watch all provisioning topics** option (option **4**) to open the Kafka Provisioning Watch node selector as demonstration in the following example:



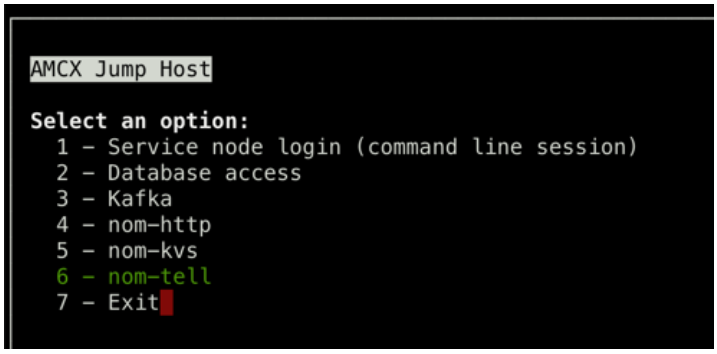
3. In the Kafka Provisioning Watch node selector, select the Kafka site for which to view provisioning topics; the example that follows contains a single site (**site1**):



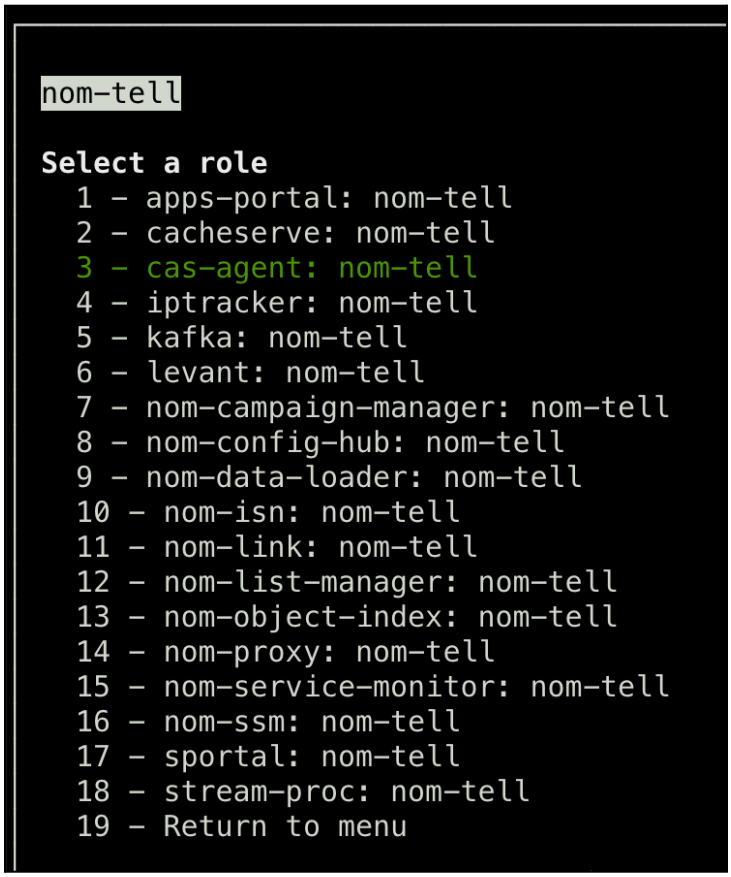
How to Access the cas-agent nom-tell Command Channel for an SSM node

Use the following steps to access the cas-agent nom-tell command channel for an SSM node:

1. In your AMCX Jump Host tool, select the **nom-tell** option (option **6**) to open the nom-tell role section screen, as demonstrated in the following example:



2. Select a the **cas-agent** role to open the node selection screen as demonstrated in the following example:



3. In the CAS agent node selector screen, select an SSM node for which to access the cas-agent nom-tell command prompt; the following example shows the CAS agent node selector screen for an environment that has one SSM:

4. At the nom-tell command prompt, you can start invoking nom-tell commands; the example that follows shows what the cas-agent nom-tell command prompt looks like:

```
cas-agent>
```