Avaya Device Enrollment Services REST API Programmer Guide

Release 3.1.18
Issue 35.0
June 3, 2021

Notice

While reasonable efforts were made to ensure that the information in this document was complete and accurate at the time of printing, Avaya Inc. can assume no liability for any errors. Changes and corrections to the information in this document might be incorporated in future releases.

Documentation disclaimer

Avaya Inc. is not responsible for any modifications, additions, or deletions to the original published version of this documentation unless such modifications, additions, or deletions were performed by Avaya. Customer and/or End User agree to indemnify and hold harmless Avaya, Avaya's agents, servants and employees against all claims, lawsuits, demands and judgments arising out of, or in connection with, subsequent modifications, additions or deletions to this documentation to the extent made by the Customer or End User.

Link disclaimer

Avaya Inc. is not responsible for the contents or reliability of any linked Web sites referenced elsewhere within this documentation, and Avaya does not necessarily endorse the products, services, or information described or offered within them. We cannot guarantee that these links will work all the time and we have no control over the availability of the linked pages.

Warranty

Avaya Inc. does not provide any warranty for the Device Enrollment Services.

Device Enrollment Services including this document is provides “AS IS” without warranty of any kind.

License

Copyright

Preventing toll fraud

"Toll fraud" is the unauthorized use of your telecommunications system by an unauthorized party (for example, a person who is not a corporate employee, agent, subcontractor, or is not working on your company's behalf). Be aware that there can be a risk of toll fraud associated with your system and that, if toll fraud occurs, it can result in substantial additional charges for your telecommunications services.

Avaya fraud intervention

If you suspect that you are being victimized by toll fraud and you need technical assistance or support, call Technical Services Center Toll Fraud Intervention Hotline at +1-800-643-2353 for the United States and Canada. For additional support telephone numbers, see the Avaya Support Web site: http://support.avaya.com

Trademarks

Avaya and the Avaya logo are either registered trademarks or trademarks of Avaya Inc. in the United States of America and/or other jurisdictions. All other trademarks are the property of their respective owners.

Downloading documents

For the most current versions of documentation, see the Avaya Support Web site: http://support.avaya.com

Contents

Chapter 1: Purpose of document

The Device Enrollment Services REST interface will support provisioning of various operations Device Enrollment Services provides.

The purpose of this document is to describe:

- The Device Enrollment Services REST interfaces supports managing Accounts, Customers, Customer Site, Provisioning URLs, Devices, Numeric Enrollment Codes.

What's New in DES 3.1.5 REST APIs

Device Enrollment Services introduces following new features

- Customer Site CRUD REST API - Site Management Interface for more details.
- Reseller Account linking REST API – Account Linking Interface for more details.

What's New in DES 3.1.6 REST APIs

Device Enrollment Services introduces following new features

- Only Serial number based device association support – Associate Devices to an Account.

What's New in DES 3.1.7 REST APIs

Device Enrollment Services introduces following new features

- A new API to get Device information – Get Device details.
- Update in GET Site API to get additional non mandatory information including NEC, PEC, ProvisionId, Associated Devices etc. - Get Site.

What’s New in DES 3.1.8 REST APIs

Device Enrollment Services introduces following new features

- Offline refresh token. – Refresh Token details
- Single Access Token per user. – Access Token details
- Disassociation new Device State. - New device state

What’s New in DES 3.1.9 REST APIs

Device Enrollment Services introduces following new features

- New V2 API for Profile CRUD operation. – Version V2 APIs for Profile Management
- New V2 API for account update actions. - Version V2 API for account update actions New V2 API for sending account linking request. – Version V2 API for sending account linking request.
- New V2 API for approving account linking request. – Version V2 API for approving account linking request

What’s New in DES 3.1.10 REST APIs

Device Enrollment Services introduces following new features

- Lightweight Create Account API – Create Account V2 API
- Support of new fields (**Reseller Order ID, Avaya Order ID, Shipment Carrier, Carrier Tracking ID**) in Device bulk operations. - Device Bulk operation API
- Additional optional field (Optional reference fields (**Reseller Order ID, Avaya Order ID, Shipment Carrier, Carrier Tracking ID**) in Get Device API – Get Device API

What’s New in DES 3.1.11 REST APIs

Device Enrollment Services introduces following new features

- Bulk basic provisioning file upload for sites – Bulk upload Basic Provisioning file
- Download Bulk upload Basic Provisioning ZIP – Download Bulk upload Basic Provisioning ZIP API
- Get Bulk Site Profile Update Job details – Get Bulk Site Profile Update Job details
- Download Device bulk job operation details – Export Device bulk operation details API

What’s New in DES 3.1.12 REST APIs

Device Enrollment Services introduces following new features

- Masquerading feature added for Service Provider role where, service provider can switch account / masquerade as any Reseller under its account hierarchy and manage any resource as Reseller.
- Access Request needs to raise to Switch as Reseller account.
- Masquerading support for all DES API - Masquerading API

What’s New in DES 3.1.14 REST APIs

Device Enrollment Services introduces following new features

- **DES-ACO integration**

  DES 3.1.14 supports automated syncing of devices from DES to ACO server using DES API. Access and use of this feature is restricted and requires a contractual authorization and training provided by Avaya.

  A new operation ‘sync’ is supported in devices API which will Claim, Associate, Active and upload supported Avaya device MAC address and device type to a designated ACO customer account

  DES already support providing additional information like **Reseller Order ID, Avaya Order ID, Shipment Carrier, Carrier Tracking ID** in devices bulk operation. (Device Bulk operation API) from DES 3.1.12 release.

What’s New in DES 3.1.18 REST APIs

Device Enrollment Services introduces following new features

- **Device lookup**

  DES 3.1.18 supports an API to device lookup across DES inventory irrespective of device claimed or unclaimed. - Lookup Device

- **Export Devices**

  An API to export devices details based on MAC address or export all devices. Devices will be exported from account bucket. – Export device data API

- **Get export job status**

  A API to get exported job status and details - Get export devices job status

- **Download Export job details**

  Download exported job. – Download exported device job API

Chapter 2: Must Read

1. **<u>DES API Access</u>**

   Not all DES users are permitted to user API, a DES API access request needs to be raised from DES GUI interface.

2. **<u>Use FQDN while gaining access to Device Enrollment Services</u>**

   Device Enrollment Services REST API should be access with FQDN des.avaya.com. The base URL for all the REST API is https://des.avaya.com

3. **<u>User Authentication</u>**

   All Device Enrollment Services REST APIs are secured, and only Authenticated users have access to respective Resource in Device Enrollment Services.

   User authentication is supported based on Basic Authentication (username + Password).

   A valid username password needs to be sent in Get Token REST API to retrieve a Token for authenticated user.

4. **<u>User Authorization</u>**

   The client application will receive an *access token* from Device Enrollment Services on success Authentication. The *access token* is digitally signed by the application realm and contains access information (like user role mappings) that the application can use to determine what resources the user can access on the application.

   So, Access token which is received after successful login should be set in REST API request Header along Authorization attribute.

   Example:

```
Authorization : bearer
eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICItaDlrQV9tTTNNZ1pXbzJ1UEx0b29
nOTcyZFZLUHRYb3lHSnJpb2hiYnpnIn0.eyJqdGkiOiIzZDU3NjJiZC04Njg3LTQxOGQtYTZiYy
0zY2FiMGVlN2Q1OGUiLCJleHAiOjE1MzI3NzcwMTQsIm5iZiI6MCwiaWF0IjoxNTMyNzc2
NzE0LCJpc3MiOiJodHRwczovL2Rlcy5hdmF5YS5jb20vYXV0aC9yZWFsbXMvZGVzIiwiYXV
kIjoiZGVzLXVpIiwic3ViIjoiOTRjNGFhMmQtMTQ1NC00ZTEwLWEwNWMtNjQ4ODlmYmZ
iMzgxIiwidHlwIjoiQmVhcmVyIiwiYXpwIjoiZGVzLXVpIiwiYXV0aF90aW1lIjowLCJzZXNzaW
9uX3N0YXRlIjoiODJmYTcwM2QtMDJmYi00NjQ5LWExYmEtMzE5M2Y3MDVlNDQ2IiwiY
WNyIjoiMSIsImFsbG93ZWQtb3JpZ2lucyI6WyJodHRwczovL2Rlcy5hdmF5YS5jb20iXSwic
mVhbG1fYWNjZXNzIjp7InJvbGVzIjpbInVtYV9hdXRob3JpemF0aW9uIl19LCJyZXNvdXJjZV
9hY2Nlc3MiOnsiZGVzLXVpIjp7InJvbGVzIjpbInVtYV9wcm90ZWN0aW9uIl19LCJhY2NvdW
```

50Ijp7InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50IiwibWFuYWdlLWFjY291bnQtbGlua3MiL
CJ2aWV3LXByb2ZpbGUiXX19LCJjbGllbnRIb3N0IjoiMTM1LjEyMy4xNDguMTY4IiwiY2xpZ
W50SWQiOiJkZXMtdWkiLCJwcmVmZXJyZWRfdXNlcm5hbWUiOiJzZXJ2aWNlLWFjY291b
nQtZGVzLXVpIiwiY2xpZW50QWRkcmVzcyI6IjEzNS4xMjMuMTQ4LjE2OCIsImVtYWlsIjoic
2VydmljZS1hY2NvdW50LWRlcy11aUBwbGFjZWhvbGRlci5vcmcifQ.agsoRoJ6sD-
M7e6x_V28-
ym3Oic8Umr8H2LfYZWPJR7tqlgjs_VTg7yQ8DPGB7ax9tjuTnKVtnGadsyB5cIqn-
ECGby2dqa1oVe9JY7QveyW9-zAwN-
XXZWGOKnphcj7xbMjsK4X8qaIHgGoq1VU6ZrA5ZeoxIzYNtfY-rG0NUAlWJNnnairknn-
XFwlo2oj61FjGSyydgXMX45XVTArEuhaXXxAm6In0JEKGJw5Ww4Yd1VRq0YVu0HcOVvB
9eaH2hCe_eMC92Jai1LMhJXQ9N1P-aurXChWEhtFvwuZcdawYyOGW-H-
lKy4AVZ35k6rHDu6sTNRbxX4sRjnEDKH-g

Chapter 3: Writing a client application

Authentication and Acquire Access Token

1. Get token

To access Device Enrollment Services REST API interface, an Authorization Bearer access token should be provided with the request. A valid Bearer access token only permit user to execute REST API.

At any given time one user could obtain single Access Token and another Access Token can be obtained only after current Access Token is expired.

Current active Access Token will be revoked on using refresh token before Access Token expiry and new Access Token will be provided.

Title	Get access token
URL	**/v1/resources/token**
Method	**POST**
URL Params	*NA*
Request Content Type	**application/x-www-form-urlencoded**
Response Content Type	**application/json**

Body	`username=[desLoginName]`
	`password=[password]`

Success Response, example	**Code**: 200 **Content**: { "access_token": "eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICItaDlrQV9tTTNNZ1pXbzJ1UEx0b29nOTcyZFZLUHRYb3lHSnJpb2hiYnpnIn0.eyJqdGkiOiIyZGVjYzY4NC1mOTZlLTQyMzAtYTAwNi0wMjc0YmQ3NmNmMGYiLCJleHAiOjE1Njc3NDY0MDgsIm5iZiI6MCwiaWF0IjoxNTY3NzQ2MTA4LCJpcYzhmLTZmYWEzOTM1NTI2ZSIsInR5cCI6IkJlYXJlciIsImF6cCI6ImRlcy11aSIsImF1dGhfdGltZSI6MCwic2Vzc2lvbl9zdGF0ZSI6ImE5

	NWIzZGRmLTBkODctNDk3OC04ODE1LTBkYTIzNDVhZTBlOSIsImFjciI 6IjEiLCJhbGxvd2VkLW9yaWdpbnMiOlsiaHR0cHM6Ly9kZXMuYXZhe WEuY29tIl0sInJlYWxtX2FjY2VzcyI6eyJyb2xlcyI6WyJvZmZsaW5lX2Fj Y2VzcyIsInVtYV9hdXRob3JpemF0aW9uIl19LCJyZXNvdXJjZV9hY2Nlc 3MiOnsiZGVzLXVpIjp7InJvbGVzIjpbInVtYV9wcm90ZWN0aW9uIiwic mVzZWxsZXIiLCJhdWRpdG9yIiwic2VydmljZS1wcm92aWRlciIsImF2Y XlhLWFkbWluIiwibWFudWZhY3R1cmVyIl19LCJhY2NvdW50Ijp7InJv bGVzIjpbIm1hbmFnZS1hY2NvdW50IiwibWFuYWdlLWFjY291bnQtb Glua3MiLCJ2aWV3LXByb2ZpbGUiXX19LCJzY29wZSI6IiIsInByZWZlcn JlZF91c2VybmFtZSI6ImRlc2NlcnR1c2VyIiwiZW1haWwiOiJkZXNjZXJ0 dXNlckBhZi5jb20ifQ.WKMEIHF_kBUEC_C5PuXwlLbeG1lYtDrwWXeS my8DTltPP619n7SZbInbjKC80sAqKAKwn7rCdXzJPES1Ey_iV5kFhjuc N00AieDr0b5yGbcBb41vxCDqKOAGkrN8lrVCWlXlOQM2P5bZnlROO g6DejeAqrYVEqPuxK7msWjKwvzP1y6BqCkLvpVAujUh0q5BYel3dDT kVtg-QBeXp2h2Ze4JXuB- l9hWcI40pQOD6Zbv9amB4g3LYtXGlfPn8bTYOFYvzoJC9- YxKKuw2Mzga4XLx_13QV_fH0xglLeiA-qIN4Lb9Wp4Qa2f6UU- Di44WJGWLS9xTOyRzIIvM4xEHw", "expires_in": 300, "refresh_expires_in": 600, "refresh_token": "eyJhbGciOiJIUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICJmOTZiNmY2 MC02NmZmLTRmZmItOWFlNC0yYzUxZGIxNjRiZmYifQ.eyJqdGkiOiI yYWQ2ZDgyZS00NzNiLTQwNjgtOTYzMS02ZTZkYzhlMmIwZmQiLCJl eHAiOjE1Njc3NDY3MDgsIm5iZiI6MCwiaWF0IjoxNTY3NzQ2MTA4LC Jpc3MiOiJodHRwczovL2Rlcy5hdmF5YS5jb20vYXV0aC9yZWFsbXMv ZGVzIiwiYXVkIjoiaHR0cHM6Ly9kZXMuYXZheWEuY29tL2F1dGgvcm VhbG1zL2RlcyIsInN1YiI6IjZlZDg0NTkzLTZkN2EtNDlhZC1iYzhmLTZm YWEzOTM1NTI2ZSIsInR5cCI6IlJlZnJlc2giLCJhenAiOiJkZXMtdWkiLCJh dXRoX3RpbWUiOjAsInNlc3Npb25fc3RhdGUiOiJhOTViM2RkZi0wZD g3LTQ5NzgtODgxNS0wZGEyMzQ1YWUwZTkiLCJyZWFsbV9hY2Nlc3 MiOnsicm9sZXMiOlsib2ZmbGluZV9hY2Nlc3MiLCJ1bWFfYXV0aG9ya XphdGlvbiJdfSwicmVzb3VyY2VfYWNjZXNzIjp7ImRlcy11aSI6eyJyb2x lcyI6WyJ1bWFfcHJvdGVjdGlvbiIsInJlc2VsbGVyIiwiYXVkaXRvciIsInNl cnZpY2UtcHJvdmlkZXIiLCJhdmF5YS1hZG1pbiIsIm1hbnVmYWN0dXJ lciJdfSwiYWNjb3VudCI6eyJyb2xlcyI6WyJtYW5hZ2UtYWNjb3VudCIsI m1hbmFnZS1hY2NvdW50LWxpbmtzIiwidmlldy1wcm9maWxlIl19fS wic2NvcGUiOiIifQ.DGAIb23EaAYAnKFybAEOvZWIZzns42PTujD_HBs dP0I", "token_type": "bearer", "not-before-policy": 1523881304, "session_state": "a95b3ddf-0d87-4978-8815-0da2345ae0e9", "scope": "" }
Error Response	**Example:** **Code:** 400 Bad Request

	Content: **{** **"error": "invalid_grant",** **"error_description": "Invalid user credentials"** **}**

2. Refresh token

Common use cases include getting new access tokens after old ones have expired or getting access to a new resource for the first time. Refresh tokens can also expire but are rather long-lived. Refresh tokens are usually subject to strict storage requirements to ensure they are not leaked. They can also be blacklisted by the authorization server.

Refresh Token is of type offline and expiry time will 30 days from the Token is issued.

Refresh Token will be revoked by DES on single usage.

Title	**Get access token using refresh token**
URL	**/v1/resources/refreshtoken**
Method	**POST**
URL Params	*NA*
Request Content Type	**application/x-www-form-urlencoded**
Response Content Type	**application/json**
Body	`refresh_token=[refresh_token]`
Success Response, example	**Code**: 200 **Content**: { "access_token": "eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICItaDlrQV9tTTNNZ1pXbzJ1UEx0b29nOTcyZFZLUHRYb3lHSnJpb2hiYnpnIn0.eyJqdGkiOiIyZGVjYzY4NC1mOTZlLTQyMzAtYTAwNi0wMjc0YmQ3NmNmMGYiLCJleHAiOjE1Njc3NDY0MDgsIm5iZiI6MCwiaWF0IjoxNTY3NzQ2MTA4LCJpc3MiOiJodHRwczovL2Rlcy5hdmF5YS5jb20vYXV0aC9yZW

	FsbXMvZGVzIiwiYXVkIjoiYWNjb3VudCIsInN1YiI6IjZlZDg0NTkzLTZkN 2EtNDlhZC1iYzhmLTZmYWEzOTM1NTI2ZSIsInR5cCI6IkJlYXJlciIsImF6 cCI6ImRlcy11aSIsImF1dGhfdGltZSI6MCwic2Vzc2lvbl9zdGF0ZSI6ImE 5NWIzZGRmLTjp7InJvbGVzIjpbInVtYV9wcm90ZWN0aW9uIiwicmVz ZWxsZXIiLCJhdWRpdG9yIiwic2VydmljZS1wcm92aWRlciIsImF2YXlhL WFkbWluIiwibWFudWZhY3R1cmVyIl19LCJhY2NvdW50Ijp7InJvbGV zIjpbIm1hbmFByb2ZpbGUiXX19LCJzY29wZSI6IiIsInByZWZlcnJlZF91c 2VybmFtZSI6ImRlc2NlcnR1c2VyIiwiZW1haWwiOiJkZXNjZXJ0dXNlck BhZi5jb20ifQ.WKMEIHF_kBUEC_C5PuXwlLbeG1lYtDrwWXeSmy8DT ltPP619n7SZbInbjKC80sAqKAKwn7rCdXzJPES1Ey_iV5kFhjucN00Aie Dr0b5yGbcBb41vxCDqKOAGkrN8lrVCWlXlOQM2P5bZnlROOg6Deje AqrYVEqPuxK7msWjKwvzP1y6BqCkLvpVAujUh0q5BYel3dDTkVtg- QBeXp2h2Ze4JXuB- l9hWcI40pQOD6Zbv9amB4g3LYtXGlfPn8bTYOFYvzoJC9- YxKKuw2Mzga4XLx_13QV_fH0xglLeiA-qIN4Lb9Wp4Qa2f6UU- Di44WJGWLS9xTOyRzIIvM4xEHw", "expires_in": 300, "refresh_expires_in": 0, "refresh_token": "eyJhbGciOiJIUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICJmOTZiNmY2 MC02NmZmLTRmZmItOWFlNC0yYzUxZGIxNjRiZmYifQ.eyJqdGkiOiI yYWQ2ZDgyZS00NzNiLTQwNjgtOTYzMS02ZTZkYzhlMmIwZmQiLCJl eHAiOjE1Njc3NDY3MDgsIm5iZiI6MCwiaWF0IjoxNTY3NzQ2MTA4LC Jpc3MiOiJodHRwZSIsInR5cCI6IlJlZnJlc2giLCJhenAiOiJkZXMtdWkiLCJ hdXRoX3RpbWUiOjAsInNlc3Npb25fc3RhdGUiOiJhOTViM2RkZi0wZ Dg3LTQ5NzgtODgxNS0wZGEyMzQ1YWUwZTkiLCJyZWFsbV9hY2Nlc 3MiOnsicm9sZXMiOlsib2ZmbGluZV9hY2Nlc3MiLCJ1bWFfYXV0aG9y aXphdGlvbiJdfSwicmVzb3VyY2VfYWNjZXNzIjp7ImRlcy11aSI6eyJyb2 xlcyI6WyJ1bWFfcHJvdGVjdGlvbiIsInJlc2VsbGVyIiwiYXVkaXRvciIsInN lcnZpY2UtcHJvdmlkZXIiLCJhdmF5YS1hZG1pbiIsIm1hbnVmYWN0dX JlciJdfSwiYWNjb3VudCI6eyJyb2xlcyI6WyJtYW5hZ2UtYWNjb3VudCI sIm1hbmFnZS1hY2NvdW50LWxpbmtzIiwidmlldy1wcm9maWxlIl19f Swic2NvcGUiOiIifQ.DGAIb23EaAYAnKFybAEOvZWIZzns42PTujD_HB sdP0I", "token_type": "bearer", "not-before-policy": 1523881304, "session_state": "a95b3ddf-0d87-4978-8815-0da2345ae0e9", "scope": "" }
Error Response	**Example:** **Code:** 400 UNAUTHORIZED **Content:** **{** **"error": "invalid_grant",** **"error_description": "Invalid refresh token"** **}**

Account Management (REST interface)

Account Type ID Details

Id	Account Type	Comments
3	Reseller	
5	Customer	Only SP & Reseller can create Customer account. Customer having access to DES API cannot create another Customer account. bpLinkId field for Customer account maps to SAP Sold to Number/Functional Location (FL)

1. Create account (V1 API)

Description	• API to create an account in Device Enrollment Services. Account could be of type Reseller or Customer. **CustomerSite** attribute is mandatory only if It’s a Customer type of Account. • On successful account creation DES account Id returned as response. • Firmware version for all the device models will be latest by default upon creation of a new account if firmwareUpgradeRequire value is true. It means firmware device being enrolled will be upgraded to the latest firmware version (listed in device family table). The firmware settings can be edited later by the account.
URL	**/v1/resources/account**
Method	**POST**
URL Params	*NA*
Request Content Type	**multipart/form-data**
Response Content Type	**application/json**

Request Data

- **Request data for Service Provider/Reseller type of Account. (Multipart Key: “accountJson”)**

```
{
"accountName":"[alphanumeric]",
"firstName":"[alphanumeric]",
"lastName":"[alphanumeric]",
"businessEmailId":"[alphanumeric]",
"businessPhoneNumber":"[alphanumeric]",
"addressl1":"[alphanumeric]",
"addressl2":"[alphanumeric]",
"addressCity":"[alphanumeric]",
"addressState":"[alphanumeric]",
"addressCountry":"[alphanumeric]",
"addressPostalCode":"[alphanumeric]",
"firmwareUpgradeRequire”:"[true / false]",
"bpLinkId":"[numeric]",
"desAccountType":
  {
    "id":"[numeric]",
   "accountType":"[alphabetic]"
  }
}
```

e.g.

```
{
   "accountName":"StarFish_Arrow3",
   "businessEmailAddress":"arrow3admin@arrow3.
com",
   "firstName":"Arrow3FN",
   "businessPhoneNumber":"02041011113",
   "isenabled":"true",
   "lastName":"Arrow3LN",
   "addressl1":"Arrow3Addr",
   "addressl2":"addrl2",
   "addressCity":"addressCity",
   "addressState":"addressState",
   "addressCountry":"addressCountry",
   "addressPostalCode":"411028",
   "desAccountType":{
      "id":"3",
      "accountType":"Reseller"
   },
   "bpLinkId":"123"
}
```

- **Request data for Customer type of Account.**

```
{
  "accountName":"[alphanumeric]",
  "firstName":"[ alphanumeric]",
  "lastName":"[ alphanumeric]",
  "businessEmailAddress":"[ emailaddress]",
  "businessPhoneNumber":"[ alphanumeric]",
  "addressl1":"[ alphanumeric]",
  "addressCity":"[ alphanumeric]",
  "addressState":"[ alphanumeric]",
  "addressCountry":"[ alphanumeric]",
  "addressPostalCode":"[ alphanumeric]",
  "firmwareUpgradeRequire":[ true / false],
  "bpLinkId":"[numeric]",
  "desAccountType":{
    "id":5,
    "accountType":"Customer"
  },
  "desCustomerSites":[
    {
      "siteLocation":"[ alphanumeric]",
      "desProvisionUrlCerts":[
        {
          "provisionUrl":"[URL]",
          "phoneGroup":[numeric],
          "serverType":"[HTTP(S)/DES]",
          "desProfile":{
            "profileName":"[alphanumeric]"
          }
        }
      ]
    }
  ]
}
```

e.g.

```
{
   "accountName":"Cust2",
   "firstName":"Cust2FN",
   "lastName":"Cust2LN",
   "businessEmailAddress":"cust2@customer.com"
,
   "businessPhoneNumber":"789987456",
```

	"**addressl1**":"Cust2Addressl1",
	"**addressCity**":"addressCity",
	"**addressState**":"addressState",
	"**addressCountry**":"addressCountry",
	"**addressPostalCode**":"123654",
	"**isenabled**":true,
	"**firmwareUpgradeRequire**":false,
	"**bpLinkId**":"12345",
	"**desAccountType**":{
	  "**id**":5,
	  "**accountType**":"Customer"
	},
	"**desCustomerSites**":[
	  {
	    "**siteLocation**":"Mumbai",
	    "**desProvisionUrlCerts**":[
	      {
	        "**provisionUrl**":"https://provision.com/asia",
	        "**phoneGroup**":0,
	        "**serverType**":"HTTP(S)",
	        "**desProfile**":{
	          "**profileName**":"Profile1"
	        }
	      }
	    ]
	  }
	]
	}
	
	URL: URL can be HTTP, HTTPS URL with IP/FQDN.

Success Response, example	**{**
	accountId : “12345678”
	}

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 400 Bad Request **Content:**

	Please refer Appendix for all event Codes and Error Messages OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages.**

2. Create account (V2 API)

Description	• API to create an Account in Device Enrollment Services. • Account can be created with ONLY with the Basic details. • On successful account creation DES account Id returned as response. • BPLinkId / FunctionalLocation field is mandatory based on account type needs to be created. ○ BPLinkId field is mandatory only for ServiceProvider / Reseller account type. ○ FunctionalLocation field is mendatory for Customer type account.
URL	**/v2/resources/account**
Method	**POST**
URL Params	*NA*
Request Content Type	**multipart/form-data**
Response Content Type	**application/json**
Request Data	{ "accountName":"*[alphanumeric]*", "firstName":"*[alphanumeric]*", "lastName":"*[alphanumeric]*", "businessEmailId":"*[alphanumeric]*", "businessPhoneNumber":"*[alphanumeric]*", "addressl1":"*[alphanumeric]*", "addressl2":"*[alphanumeric]*", "addressCity":"*[alphanumeric]*", "addressState":"*[alphanumeric]*",

```
"addressCountry":"[alphanumeric]",
"addressPostalCode":"[alphanumeric]",
"bpLinkId":"[numeric]",
"functionalLocation":"[numeric]",
"desAccountType":
  {
    "id":"[numeric]",
   "accountType":"[alphabetic]"
  }
}
```

e.g.

```
{
   "accountName":"StarFish_Arrow3",
   "businessEmailAddress":"arrow3admin@arrow3.com",
   "firstName":"Arrow3FN",
   "businessPhoneNumber":"02041011113",
   "isenabled":"true",
   "lastName":"Arrow3LN",
   "addressl1":"Arrow3Addr",
   "addressl2":"addrl2",
   "addressCity":"addressCity",
   "addressState":"addressState",
   "addressCountry":"addressCountry",
   "addressPostalCode":"411028",
   "desAccountType":{
      "id":"3",
      "accountType":"Reseller"
   },
   "bpLinkId":"123"
}
```

e.g.

```
{
   "accountName":"StarFish_Arrow3",
   "businessEmailAddress":"arrow3admin@arrow3.com",
   "firstName":"Arrow3FN",
   "businessPhoneNumber":"02041011113",
   "isenabled":"true",
   "lastName":"Arrow3LN",
   "addressl1":"Arrow3Addr",
   "addressl2":"addrl2",
```

	"addressCity":"addressCity",
	"addressState":"addressState",
	"addressCountry":"addressCountry",
	"addressPostalCode":"411028",
	"desAccountType":{
	"id":"3",
	"accountType":"Customer"
	},
	"functionalLocation":"123"
	}

Success Response, example	**{**
	accountId : “12345678”
	}

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 400 Bad Request **Content:** **Please refer Appendix for all event Codes and Error Messages** OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

3. Get account details

Description	• API to get an Account details. Response JSON return complete Account hierarchy including Account Users, Site and Profile Associated with Site. • Account Users attribute will be non-empty for Reseller and Service Provider.

	● Site and Associated Profiles is applicable for Customer type of Account.
URL	**/v1/resources/account/{accountId}**
Method	**GET**

URL Params	**Required:**
	`accountId``=[numeric]`
	
	example:
	`https://des.avaya.com/v1/resources/account/12345678`

Request Content Type	*NA*
Response Content Type	**application/json**
Request Params	*NA*

Success Response, example	**Code**: 200
	Content:
	
	`{`
	`"accountId":78,`
	`"accountName":"Devices-SV-BDDMS-cust",`
	`"desUniqueId":64862163,`
	`"firstName":"Devices-SV-BDDMS-custFN",`
	`"lastName":"Devices-SV-BDDMS-custLN",`
	`"addressl1":"Avaya Inc.,",`
	`"addressl2":"4655 Great America Parkway,",`
	`"addressCity":"Santa Clara,",`
	`"addressState":"CA",`
	`"addressCountry":"USA",`
	`"addressPostalCode":"95054-1233",`
	`"businessPhoneNumber":"1234567890",`
	`"businessEmailAddress":"dasp@avaya.com",`
	`"isenabled":true,`
	`"firmwareUpgradeRequire":false,`
	`"generateAvayaCert":true,`
	`"bpLinkId":null,`

```
   "profilePublicKey":null,
   "desAccountType":{
      "id":5,
      "accountType":"Customer"
   },
   "desAccountUsers":[

   ],
   "desCustomerSites":[
      {
         "id":34,
         "siteLocation":"BSDMS",
         "provisionUrlEnrollmentCode":null,
         "desProvisionUrlCerts":[
            {
               "id":355,
               "provisionUrl":"https://xsp1.iop1.broadworks.net/dms/Avaya_J129_DM",
               "phoneGroup":0,
               "serverType":"HTTP(S)",
               "desProfile":{
                  "id":33,
                  "profileName":"Devices-SV-BSDMS"
               }
            }
         ]
      },
      {
         "id":53,
         "siteLocation":"new",
         "provisionUrlEnrollmentCode":null,
         "desProvisionUrlCerts":[
            {
               "id":491,
               "provisionUrl":"http://XSP-bwlabs.devices.avaya.com/dms/Pune_SV",
               "phoneGroup":0,
               "serverType":"HTTP(S)",
               "desProfile":{
                  "id":55,
                  "profileName":"dasp"
               }
            }
         ]
```

	`}`
	`]`
	`}`

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

4. Update account details

Description	• API update Account basic details.
URL	**/v1/resources/account/{accountId}**
Method	**PUT**
URL Params	**Required:** **`accountId`**`=[numeric]` = Account Id of account to be updated example: `https://des.avaya.com/v1/resources/account/12345678`
Request Content Type	**multipart/form-data**
Response Content Type	**application/json**

Request Data

- **Update Account basic details JSON . (Multipart Key: “accountJson”)**
- **bpLinkId field maps to SAP Sold to Number/Functional Location (FL) for Customer account.**

```
{
  "accountName":"[alphanumeric]",
  "firstName":"[alphanumeric]",
  "lastName":"[ alphanumeric]",
  "businessEmailAddress":"[ emailaddress]",
  "businessPhoneNumber":"[ alphanumeric]",
  "addressl1":"[ alphanumeric ]",
  "addressl2":"[ alphanumeric ]",
  "addressCity":"[ alphanumeric ]",
  "addressState":"[ alphanumeric ]",
  "addressCountry":"[ alphanumeric ]",
  "addressPostalCode":"[ alphanumeric ]",
  "isenabled":[ true / false ],
  "bpLinkId":"[ numeric ]",
}
```

e.g. for Reseller / SP account update

```
{
   "accountName":"Devices-SV",
   "firstName":"Devices-SVFN",
   "lastName":"Devices-SVLN",
   "businessEmailAddress":"mehra9@avaya.com",
   "businessPhoneNumber":"912066886367",
   "addressl1":"Avaya Inc.,",
   "addressl2":"4655 Great America Parkway,",
   "addressCity":"Santa Clara,",
   "addressState":"CA",
   "addressCountry":"USA",
   "addressPostalCode":"95054-1233",
   "isenabled":true,
   "bpLinkId":"1233",

}
```

e.g. for Customer account update

```
{
   "accountName":"Cust2",
   "firstName":"Cust2FN",
   "lastName":"Cust2LN",
```

	"businessEmailAddress":"cust2@customer.com",
	"businessPhoneNumber":"789987456",
	"addressl1":"Cust2Addressl1",
	"addressCity":"addressCity",
	"addressState":"addressState",
	"addressCountry":"addressCountry",
	"addressPostalCode":"123654",
	"bpLinkId":"12345",
	"isenabled":true,
	"firmwareUpgradeRequire":false,
	"desAccountType":{
	"id":5,
	"accountType":"Customer"
	},
	"desCustomerSites":[
	{
	"siteLocation":"Mumbai",
	"desProvisionUrlCerts":[
	{
	"provisionUrl":"https://provision.com/asia",
	"phoneGroup":0,
	"serverType":"HTTP(S)",
	"desProfile":{
	"profileName":"Profile1"
	}
	}
	]
	}
	]
	}

Success Response, example	**Code**: 200 **{** **"code":"DES_152",** **"message":"Account has been updated successfully."** **}**
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support

	OR **Code:** 400 Bad Request **Content:** **Please refer Appendix for all event Codes and Error Messages** OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

5. Delete account

Description	• API delete account basic details along with Account User, Customer Site. • On account delectation all the profiles and devices associated with respective account site gets released. • On reseller account deletion, profiles, NECs and Customer accounts created by that reseller will get deleted. • Own account cannot be deleted, please contact Avaya Administrator.
URL	**/v1/resources/account/{accountId}**
Method	**DELETE**
URL Params	**Required:** **`accountId`**`=[numeric]` = Account Id of account to be deleted example: `https://des.avaya.com/v1/resources/account/12345678`
Request Content-Type	*NA*
Response	**application/json**

Content-Type	
Request Params	*NA*

Success Response, example	**{**
	"code":"DES_153",
	"message":"Account has been deleted successfully."
	}

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

6. Account actions update (V1 API)

Description	• Various flag at account level will get be updated • Action needs to mention to update the respective operation / feature for an Account • Action could be one of below. • V1 account actions update API's response content type is text/plain. `[enable | disable | autoFirmwareUpgradeEnable | autoFirmwareUpgradeDisable | twoFactorAuthEnable | twoFactorAuthDisable]`
URL	**/v1/resources/account/{`accountId`}/{action}**
Method	**PUT**
URL Params	**Required:** **`accountId`**`=[numeric]`

	`action = [enable | disable | autoFirmwareUpgradeEnable | autoFirmwareUpgradeDisable | twoFactorAuthEnable | twoFactorAuthDisable]`
	
	example:
	`https://des.avaya.com/v1/resources/account/12345678/twoFactorAuthEnable`

Request Content-Type	**application/json**
Response Content-Type	**text/plain**
Request Params	*NA*

Success Response, example	**{**
	"code":"DES_152",
	"message":"Account has been updated successfully."
	}

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

7. Account actions update (V2 API)

Description	- Various flag at account level will get be updated - Action needs to mention to update the respective operation / feature for an Account - Action could be one of below. - **V2 account actions update API's response content type is application/json unlike V1 API.** `[enable | disable | autoFirmwareUpgradeEnable | autoFirmwareUpgradeDisable | twoFactorAuthEnable | twoFactorAuthDisable]`
URL	**/v2/resources/account/{accountId}/{action}**
Method	**PUT**
URL Params	**Required:** **accountId**=[numeric] **action** = [enable | disable | autoFirmwareUpgradeEnable | autoFirmwareUpgradeDisable | twoFactorAuthEnable | twoFactorAuthDisable] example: https://des.avaya.com/v1/resources/account/12345678/twoFactorAuthEnable
Request Content-Type	**application/json**
Response Content-Type	**application/json**
Request Params	*NA*

Success Response, example	**{
	"code":"DES_152",
	"message":"Account has been updated successfully."
	}**

Error Response	**Example:** **Code:** 401 UNAUTHORIZED

	Content: Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

8. Account hierarchy access request

Description	- **This API is applicable for Service Provider accounts.** - **Service Provider can request to get access to their reseller accounts.** - **Service provider can masquerade their reseller accounts after the request is approved by Avaya Admin.** - **Service Provider will be notified on approval of the request.**
URL	**/v1/resources/hierarchyaccess**
Method	**POST**
URL Params	*NA*
Request Content-Type	*NA*
Response Content-Type	**application/json**
Request Params	*NA*
Success Response, example	**{"code":"DES_310","message":"Hierarchy access request has been created successfully."}**

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

Account Linking (REST Interface)

9. Reseller account linking request (V1 API)

Description	• API used to send Account Linking request to Reseller. • Account linking request can be sent only Global Reseller available on DES • Response content-type is text/plain
URL	**/v1/resources/account/{bplinkId}/linkReseller**
Method	**PUT**

URL Params	**Required:**
	`bplinkId=[numeric]`
	bpLinkId of reseller account to link with
	
	example:
	`https://des.avaya.com/v1/resources/account/12345/linkReseller`

Request Content-Type	**application/json**
Response Content-Type	**text/plain**
Request Params	*NA*

Success Response, example	**{**
	"code": "DES_555",
	"message": "Account linking request has been successfully sent."
	}

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR

	Code: 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

10. Approving account linking request (V1 API)

Description	• API used to approve Account Linking request sent by Service Provider. • Resellers are only allowed to either Approve / Deny account linking request • BPLinkId of requestor needs to be sent in approve account linking. • Response content-type is text/plain
URL	**/v1/resources/account/{bplinkId}/approveLinkRequest**
Method	**PUT**
URL Params	**Required:** **bplinkId** `=[numeric]` = bpLinkId of service provider account who has sent linking request example: `https://des.avaya.com/v1/resources/account/54321/approveLinkRequest`
Request Content-Type	**application/json**
Response Content-Type	**text/plain**
Request Params	*NA*

Success Response, example	**{**
	"code": "DES_558",
	"message": "Account has been successfully linked."
	}

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

11. Reseller account linking request (V2 API)

Description	• API used to send Account Linking request to Reseller. • Account linking request can be sent only Global Reseller available on DES • **Response content-type is application/json**
URL	**/v2/resources/account/{bplinkId}/linkReseller**
Method	**PUT**
URL Params	**Required:** **`bplinkId`**`=[numeric]` bpLinkId of reseller account to link with example: `https://des.avaya.com/v1/resources/account/12345/linkReseller`
Request Content-Type	**application/json**
Response Content-Type	**application/json**
Request Params	*NA*

Success Response, example	**{**
	"code": "DES_555",
	"message": "Account linking request has been successfully sent."
	}

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

12. Approving account linking request (V2 API)

Description	• API used to approve Account Linking request sent by Service Provider. • Resellers are only allowed to either Approve / Deny account linking request • BPLinkId of requestor needs to be sent in approve account linking. • **Response content-type is application/json**
URL	**/v2/resources/account/{bplinkId}/approveLinkRequest**
Method	**PUT**
URL Params	**Required:** **bplinkId** `=[numeric]` = bpLinkId of service provider account who has sent linking request example: `https://des.avaya.com/v1/resources/account/54321/approveLinkRequest`
Request Content-Type	**application/json**

Response Content-Type	**application/json**
Request Params	*NA*

Success Response, example	**{**
	"code": "DES_558",
	"message”: “Account has been successfully linked."
	}

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

Administrator User Management (REST interface)

Locale Details

Language	**Locale Value**
German	de
English (United States)	en-us
Russian (Russia)	ru
Korean	ko
Japanese	ja
Latin (Spain)	la_es
Italian	it
Portuguese (Brazil)	pt_br
French	fr
Chinese (China)	zh

1. Create administrative user

Description	• Create Administrative User for an Account.
URL	**/v1/resources/administrator/**
Method	**POST**
URL Params	*NA*
Request Content Type	**application/json**
Response Content Type	**application/json**

Request Params	Create Administrator request
	{
	"emailAddress":"[*emailaddress*]",
	"firstName":"[*alphanumeric*]",
	"lastName":"[*alphanumeric*]",
	"loginName":"[*alphanumeric*]",
	"desAccount":{

	"desUniqueId":[*numeric*]
	},
	"locale":"en-us",
	"isenabled":true,
	"password":"[*alphanumeric*]"
	}
	
	e.g.
	{
	"emailAddress":"testuser@avayades.com",
	"firstName":"Test",
	"lastName":"User",
	"loginName":"testuser@avayades.com",
	"desAccount":{
	"desUniqueId":90830727
	},
	"locale":"en-us",
	"isenabled":true,
	"password":"Avaya123$"
	}

Success Response, example	**{**
	"code":"DES_169",
	"message":"User has been created successfully."
	}

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

2. Get administrator user details

Description	- Get Administrative User details.
URL	**/v1/resources/administrator/{loginname}**
Method	**GET**

URL Params	**Required:**
	`loginname=[alphanumeric]`
	
	example:
	`https://des.avaya.com/v1/resources/administrator/alphadesadmin@avaya.com`

Request Content Type	*NA*
Response Content Type	**application/json**
Request Params	*NA*

Success Response, example	`{`
	`"emailAddress":"testuser@avayades.com",`
	`"firstName":"Test",`
	`"lastName":"User",`
	`"loginName":"testuser@avayades.com",`
	`"desAccount":{`
	`"desUniqueId":90830727`
	`},`
	`"id":1234,`
	`"locale":"en-us",`
	`"isenabled":true`
	`}`

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR

	Code: 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

3. Update administrator user details

Description	- Update Administrative User details.
URL	**/v1/resources/administrator/**
Method	**PUT**
URL Params	*NA*
Request Content Type	**application/json**
Response Content Type	**application/json**

Request Params	Create Administrator request
	{
	"emailAddress":"[*emailaddress*]",
	"firstName":"[*alphanumeric*]",
	"lastName":"[*alphanumeric*]",
	"loginName":"[*alphanumeric*]",
	"id":"[*numeric*]",
	"desAccount":{
	"desUniqueId":[*numeric*]
	},
	"locale":"en-us",
	"isenabled":true,
	"password":"[*alphanumeric*]"
	}
	
	`e.g.`
	`{`
	`"emailAddress":"testuser@avayades.com",`

	"firstName":"Test",
	"lastName":"User",
	"loginName":"testuser@avayades.com",
	"id": 1234,
	"desAccount":{
	"desUniqueId":90830727
	},
	"locale":"en-us",
	"isenabled":true,
	"password":"Avaya123$"
	}

Success Response, example	**{**
	"code":"DES_170",
	"message":"User has been updated successfully."
	}

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

4. Delete administrator user

Description	- Delete Administrative User of an Account.
URL	**/v1/resources/administrator/{loginName}**
Method	**DELETE**
URL Params	**Required:** **loginname**=[alphanumeric] example:

	`https://des.avaya.com/v1/resources/administrator/alphadesadmin@avaya.com`
Request Content Type	*NA*
Response Content Type	**application/json**
Request Params	*NA*

Success Response, example	**{**
	"code":"DES_171",
	"message":"User(s) have been deleted successfully."
	}

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

Site Management (REST interface)

1. Create site for Customer account

Description	• API to create Site for Customer account. • Site can be created with or without associating profile, provisioning URL • If Provision URL is DES then a basic Provisioning ZIP also needs to be provided while adding Site.
URL	**/v1/resources/site/{accountId}**
Method	**POST**
URL Params	**Required:** **accountId**= Customer DES account Id for which Site needs to be added. example: https://des.avaya.com/v1/resources/site/12345678
Request Content Type	**Multipart/form-data**
Response Content Type	**application/json**

Request Data	• **Multipart Form data key**: siteJson
	• **value:** {
	"siteLocation":"Boston",
	"desProvisionUrlCerts":[
	{
	"provisionUrl":"https://provisionurl.new.starfish.com/europe",
	"phoneGroup":"0",
	"serverType":"HTTP(S)",
	"desProfile":{
	"profileName":"Alpha-Profile"
	}
	}
	]
	}
	Add more form data key values pairs for uploading basic

	provisioning zip file if associating with DES type provisioning URL with **key as <site>** (in this case "Boston") and **value as the file**. - **Multipart Form data key**: <site> - **value:** <Basic Provisioning ZIP file> **Example:** - **Multipart Form data key**: Boston - **value:** basicProvFile1.zip - **Multipart Form data key**: London - **value:** basicProvFile2.zip

Success Response, example	**Code**: 200
	Content:
	
	`{`
	`"code":"DES_1011",`
	`"message":"Site has been created successfully."`
	`}`

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

2. Get site details

Description	- API to get Customer Site details like Profile, Prov URL, Phone Group, PEC code. - Devices attribute with list of Devices with MAC, Serial number will be included in response If there are any devices associated against Site. - Enrollment Code related information like Numeric Enrollment Code, Provision Enrollment Code, Provisioning Id will be included in response if available.

URL	**/v1/resources/site/{accountId }/{siteName}**
Method	**GET**
URL Params	**Required:** **accountId** = Customer DES account Id for which Site details needs to be fetched. **siteName** =Name of Site for which details needs to be fetched. example: https://des.avaya.com/v1/resources/site/12345678/alpha-site
Request Content Type	*NA*
Response Content Type	**application/json**
Request Data	*NA*

Success Response, example	**Code**: 200
	Content:
	
	{
	"**siteLocation**":"alpha-site",
	"**nec**":"12345678",
	"**provisionId**" : "9012",
	"**pec**":"123456789012",
	"**provisionUrlCerts**":[
	{
	"**provisionUrlCert**":{
	"**provisionUrl**":"https://des.avaya.com",
	"**phoneGroup**":0,
	"**serverType**":"DES",
	"**desProfile**": {
	"**profileName**":"alphaProfile"
	}
	}
	}
	],
	"**devices**":[
	{
	"**MAC**":"010000000002",

	`"Serial":"E0W8M24E2C30"`
	`},`
	`{`
	`"MAC":"010000000001",`
	`"Serial":"F9M0V3K9UC4W"`
	`}`
	`]`
	`}`

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

3. Download Provisioning settings file

Description	- API to download basic provisioning settings file if exists.
URL	**/v1/resources/account/download/{`accountName`}/{siteLocation}**
Method	**GET**

URL Params	**Required:**
	`accountName` `= Customer DES account Name.`
	
	`siteName` `= Name of Site for which basic provisioning file to be downloaded.`
	example:
	`https://des.avaya.com/v1/resources/account/download/alpha_account/alpha-site`

Request Content Type	*NA*
Response	**application/octet-stream**

Content Type	
Request Data	*NA*
Success Response, example	**Code**: 200 **Content**: **`Actual basic provisioning settings file`**
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

4. Update Site details

Description	• API to update Site details. • New Profile can be associated, disassociated with site. • If Provision URL is DES, then a basic Provisioning ZIP also needs to be provided. • Existing Basic Provisioning ZIP also can be updated against Site.
URL	**/v1/resources/site/{`accountId`}**
Method	**PUT**

URL Params	**Required:**
	`accountId``= Customer DES account Id for which Site details needs to be fetched.`
	example:
	`https://des.avaya.com/v1/resources/site/12345678`

Request Content Type	**Multipart/form-data**
Response Content Type	**application/json**

Request Data	- **Multipart Form data key**: siteJson
	- **value:** {
	"siteLocation":"Boston",
	"desProvisionUrlCerts":[
	{
	"provisionUrl":"https://provisionurl.new.starfish.com/europe",
	"phoneGroup":"0",
	"serverType":"HTTP(S)",
	"desProfile":{
	"profileName":"Alpha-ProfileTest"
	
	}
	}
	]
	}
	Add more form data key values pairs for uploading basic provisioning zip file if associating with DES type provisioning URL with **key as <site>** (in this case "Boston") and **value as the file**.
	- **Multipart Form data key**: <site>
	- **value:** <Basic Provisioning ZIP file>
	Example:
	- **Multipart Form data key**: Boston
	- **value:** basicProvFile1.zip
	- **Multipart Form data key**: London
	- **value:** basicProvFile2.zip

Success Response, example	**Code**: 200 **Content**: **{** **"code":"DES_1011",** **"message":"Site has been updated successfully."** **}**
Error	**Example:**

Response	**Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

5. Delete Site

Description	• API to delete site.
URL	**/v1/resources/site/{accountId}/{siteName}**
Method	**DELETE**
URL Params	**Required:** **accountId**= Customer DES account Id for which Site details needs to be fetched. **siteName** =Name of Site for which details needs to be fetched. example: https://des.avaya.com/v1/resources/site/12345678/alpha-site
Request Content Type	*NA*
Response Content Type	**application/json**
Request Data	*NA*
Success Response,	**Code**: 200 **Content**:

example	`{`
	`"code":"DES_1011",`
	`"message":"Site has been deleted successfully."`
	`}`

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

6. Bulk upload Provisioning file

Description	• API to associate multiple existing sites across different customer accounts with provisioning URL and to upload a single Basic Provisioning file for all if Provision URL is DES.
URL	**/v1/resources/site/bulk**
Method	**PUT**
URL Params	**No URL Path Params** example: `https://des.avaya.com/v1/resources/site/bulk`
Request Content Type	**Multipart/form-data**
Response Content Type	**application/json**

Request Data	**Example:**
	• **Multipart Form data key**: “desAccounts”
	• **value:** [{
	"desUniqueId": "12345678",
	"desCustomerSites": [
	{
	"siteLocation": "Ohio"
	}, {
	"siteLocation": "New Jersey"
	}
	]
	}, {
	"desUniqueId": "22445566",
	"desCustomerSites": [
	{
	"siteLocation": "Pune"
	}, {
	"siteLocation": "Mumbai"
	}
	]
	}]
	• **Multipart Form data key**: “desProvisionUrlCert”
	value: {
	"provisionUrl":"https://des.avaya.com",
	"phoneGroup":"0",
	"serverType":"DES",
	"desProfile":{
	"profileName":"Alpha-ProfileTest"
	}
	}
	• **Multipart Form data key**: “ “basicProvisioningZip”
	• **value:** basicProvFile1.zip

Success Response, example	**Code**: 200 **Content**: { **"code":"DES_1011",** **"message":"Site has been updated successfully."**

	}
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

7. Download bulk Provisioning file

Description	• API to download Basic Provisioning file uploaded against Bulk upload basic provisioning zip for multiple Site update.
URL	**/v1/resources/site/download/{jobName}**
Method	**GET**
URL Params	**Required:** `jobName` = Name of the job returned by bulk site profile association operation. example: `https://des.avaya.com/v1/resources/site/download/sampleJob`
Request Content Type	*NA*
Response Content Type	**application/octet-stream**
Request Data	*NA*
Success	**Code**: 200

Response, example	**Content**: **`Actual basic provisioning settings file`**
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

8. Get bulk Site profile association job details

Description	- API to get bulk site profile association job details
URL	**/v1/resources/site/bulk/{jobName}**
Method	**GET**

URL Params	**Required:**
	`jobName` `= Name of the job returned by bulk site profile association operation.`
	example:
	`https://des.avaya.com/v1/resources/site/bulk/sampleJob`

Request Content Type	*NA*
Response Content Type	**application/json**
Request Data	*NA*

Success Response, example	**Code**: 200
	Content:
	
	`{`
	`   "id":54,`
	`   "jobName":"Avaya Administrator_Update_Bulk_Site_1598270801426",`
	`   "profileName":"AvayaCustProfile",`
	`   "provisioningURL":"https://avayacust.avaya.com",`
	`   "phoneGroup":0,`
	`   "isBasicProvFileExists":true,`
	`   "totalSitesCount":2,`
	`   "successSitesCount":0,`
	`   "failedSitesCount":2,`
	`   "percentage":100`
	`}`

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

Device Management (REST interface)

1. Claim, Disassociate, Release, Activate, Deactivate Device, Sync

Description	• API to Claim/Disassociate/Release/Activate/Deactivate device. • All operations can be perform using either MAC or Serial or both. • Optional reference fields (**Reseller Order ID, Avaya Order ID, Shipment Carrier, Carrier Tracking ID**) are applicable only for **Claim, Associate, Active operation**. • A device can be … o activated/deactivated, only if it is currently owned by that account or by any account created by it. o released (by Service Provider or Reseller), only if it is currently owned by that account. o released (by Direct Customer), only if it is currently owned by that account and not associated to any of its sites. o claimed (by Service Provider or Direct Customer) only if it is currently owned by the Avaya Administrator. o claimed (by Reseller) only if it is currently owned by Avaya Administrator or by its parent Service Provider. o disassociated (by Service Provider or Reseller) only if it is currently owned by any account created by it. o disassociated (by Direct Customer) only if it is currently associated to one of its customer sites.
URL	**/v1/resources/inventory/{action}**
Method	**POST**

URL Params	**Required:**
	`action =[ activate / deactivate / claim / disassociate / release ]`
	
	example:
	`https://des.avaya.com/v1/resources/inventory/claim`

Request Content Type	**application/json**
Response Content Type	**application/json**

Request Data

- **Single Device operation**

```
[{
"MAC":"[HEXDECIAL]”,
"Serial":"[ ALPANUMERIC]”
 "resellerOrderId": "[ ALPANUMERIC]",
  "avayaOrderId": "[ ALPANUMERIC]",
  "carrierTrackingId": "[ ALPANUMERIC]",
  "shipmentCarrier": "[ ALPANUMERIC]"
}]
```

e.g.

```
[{
   "MAC":"010000000001",
   "Serial":"E0W8M24E2C31",
    "resellerOrderId": "StanleyCarpets",
    "avayaOrderId": "A1S-123429834",
    "shipmentCarrier": "FedEx",
    "carrierTrackingId":"10987123kasdf"
}]
```

- **Multiple Device operation**

```
[{
"MAC":"[HEXDECIAL]”,
"Serial":"[ ALPANUMERIC]”
 "resellerOrderId": "[ ALPANUMERIC]",
  "avayaOrderId": "[ ALPANUMERIC]",
  "carrierTrackingId": "[ ALPANUMERIC]",
  "shipmentCarrier": "[ ALPANUMERIC]"
},
{
"MAC":"[HEXDECIAL]”,
"Serial":"[alphanumeric]”
}]
```

e.g.

```
[{
   "MAC":"010000000001",
   "Serial":"E0W8M24E2C31"
    "resellerOrderId": "StanleyCarpets",
    "avayaOrderId": "A1S-123429834",
    "shipmentCarrier": "FedEx",
    "carrierTrackingId":"10987123kasdf"
},
{
```

	"MAC":"010000000002", "Serial":"E0W8M24E2C32" }, { "MAC":"010000000003", "Serial":"E0W8M24E2C33" }]
Success Response, example	**Code**: 200 **Content**: **{** **"code":"DES_1011",** **"message":"Device has been activated successfully."** **}** If more than one devices are used in request data for respective action, DES schedule a job the bulk operation. A Job name will be return for the bulk operation request e.g. **Code**: 200 **Content**: **{** **"jobName":"bulk activation_1577692090568"** **}**
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

2. ACO Sync API

Description	• **ACO-SYNC** ○ Sync operation access is restricted to Avaya Authorized Distributors (requires agreement, training and access to a ACO enabled DES account managed by Avaya to access). ○ Sync operation will Claim, Associate, Active devices based on data provided in sync operation. ○ Sync operation will override prior Claims by all other accounts for ACO. ○ Sync operation will upload the MAC address and device information necessary to upload the device to a specific ACO customer account (aka ACO UserID) provided in the resellerOrderId field. ○ Sync operation requires the ACO customer account to have an available 'existing device' without an assigned device for DES to successfully upload to ACO. ○ PhoneNumber is optional field to sync devices. If provided devices will be synced against specified phone number.
URL	**/v1/resources/inventory/sync**
Method	**POST**
URL Params	example: `https://des.avaya.com/v1/resources/inventory/sync`
Request Content Type	**application/json**
Response Content Type	**application/json**
Request Data	• **Single Device operation** [{ "**MAC**":"*[HEXDECIAL]",* "**Serial**":"*[ALPANUMERIC]"* "**resellerOrderId**": "[*ALPANUMERIC*]", "**avayaOrderId**": "[*ALPANUMERIC*]", "**carrierTrackingId**": "[*ALPANUMERIC*]", "**shipmentCarrier**": "[*ALPANUMERIC*]", "**accountName**": "[*ALPANUMERIC*]",

```
 "siteName": "[ ALPANUMERIC]",
 "phoneNumber": "[ E.164 format]"
}]
```

e.g.

```
        [{
           "MAC":"010000000001",
           "Serial":"E0W8M24E2C31",
            "resellerOrderId": "StanleyCarpets",
            "avayaOrderId": "A1S-123429834",
            "shipmentCarrier": "FedEx",
            "carrierTrackingId":"10987123kasdf",
            "accountName": "ACO Customer",
            "siteName":"Pune",
            "phoneNumber":"+912041019991"
        }]
```

- **Multiple Device operation**

```
[{
"MAC":"[HEXDECIAL]”,
 "resellerOrderId": "[ ALPANUMERIC]",
  "avayaOrderId": "[ ALPANUMERIC]",
  "carrierTrackingId": "[ ALPANUMERIC]",
  "shipmentCarrier": "[ ALPANUMERIC]"
  "accountName": "[ ALPANUMERIC]",
  "siteName": "[ ALPANUMERIC]",
  "phoneNumber": "[ E.164 format]"

},
{
 "resellerOrderId": "[ ALPANUMERIC]",
  "avayaOrderId": "[ ALPANUMERIC]",
  "carrierTrackingId": "[ ALPANUMERIC]",
  "shipmentCarrier": "[ ALPANUMERIC]"
  "accountName": "[ ALPANUMERIC]",
  "siteName": "[ ALPANUMERIC]",
  "phoneNumber": "[ E.164 format]"

}]
```

e.g.

```
        [{
           "MAC":"010000000001",
            "resellerOrderId": "StanleyCarpets",
```

	"**avayaOrderId**": "A1S-123429834", "**shipmentCarrier**": "FedEx", "**carrierTrackingId**":"10987123kasdf", "**accountName**": "ACO Customer1", "**siteName**":"Pune1", "**phoneNumber**":"+912041019991" }, { "**Serial**":"E0W8M24E2C32" "**resellerOrderId**": "StanleyCarpets", "**avayaOrderId**": "A1S-123429834", "**shipmentCarrier**": "FedEx", "**carrierTrackingId**":"10987123kasdf", "**accountName**": "ACO Customer2", "**siteName**":"Pune2", "**phoneNumber**":"+912041019992" }]
Success Response, example	**Code**: 200 **Content**: **{** **"code":"DES_364",** **"message":"Device synced successfully."** **}** If more than one devices are used in request data for respective action, DES schedule a job the bulk operation. A Job name will be return for the bulk operation request e.g. **Code**: 200 **Content**: **{** **"jobName":"ACO_SP_1577692090568"** **}**
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR

	Code: 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

3. Validate device

Description	- API checks is a valid Device found in Device Enrollment Service against attached Self Signed Certificate.
URL	**/v1/resources/validatedevice**
Method	**POST**
URL PARAM's	*NA*
Request Content Type	**multipart/form-data**
Response Content Type	**text/plain**
Request Data	-----------------------------821439392029 Content-Disposition: form-data; name="certificate"; filename="deviceCert.crt" Content-Type: application/x-x509-ca-cert -----BEGIN CERTIFICATE----- MIIGxTCCBK2gAwIBAgITJQAAAAJVT7deqWLWCwAAAAAAAjANBgkq hkiG9w0BAQsF ADA2MQ4wDAYDVQQKEwVBdmF5YTELMAkGA1UECxMCSVQxFzAV BgNVBAMTDkF2YXlh SVRyb290Q0EyMB4XDTE1MDQyOTE3MTcyMVoXDTQwMDQyOTE3 MjcyMVowXzETMBEG -----END CERTIFICATE----- -----------------------------821439392029—
Success Response, example	**Code**: 200 **Content**: `A valid device certificate.`
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:**

	Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

4. Get device details

Description	• Get Device details using either MAC or Serial or Both. • Either of the value is mandatory to get the Device details. • If fields (**Reseller Order ID, Avaya Order ID, Shipment Carrier, Carrier Tracking ID**) are populated against device during Bulk operation then respective fields and values will be part of get device response.
URL	**/v1/resources/inventory;MAC={MAC};Serial={Serial}**
Method	**GET**
URL Params	**Required:** **MAC**= MAC address of the Device wants to get details. **Serial**= Serial number of the Device wants to get details. example: Using only MAC https://des.avaya.com/v1/resources/inventory;MAC=010000000001 Using only Serial Number https://des.avaya.com/v1/resources/inventory;Serial=F9M0V3K9UC4W Using both MAC & Serial Number https://des.avaya.com/v1/resources/inventory;MAC=010000000001;Serial=F9M0V3K9UC4W
Request	**NA**

Content Type	
Response Content Type	**application/json**
Request Data	NA

Success Response, example	**Code**: 200
	Content:
	
	{
	"**MAC**": "[*HEX*]",
	"**Serial**": "[*ALPANUMERIC*]",
	"**Product**": "[*ALPANUMERIC*]",
	"**Model**": "[*ALPANUMERIC*]",
	"**Hardware**": "[*ALPANUMERIC*]",
	"**ResellerOrderId**": "[*ALPANUMERIC*]",
	"**AvayaOrderId**": "[*ALPANUMERIC*]",
	"**CarrierTrackingId**": "[*ALPANUMERIC*]",
	"**ShipmentCarrier**": "[*ALPANUMERIC*]",
	"**ownerAccount** ": "[*ALPANUMERIC*]",
	"**site**" : "[*ALPANUMERIC*]",
	"**provisioningURL**": "[*URL*]",
	"**isActive**": [*TRUE/FALSE*],
	"**isLocked**" : [*TRUE/FALSE*],
	"**firmwareVersion**":"[*ALPANUMERIC*]",
	"**labels**": "[*ALPANUMERIC*]"
	}
	
	`e.g.`
	`{`
	`"MAC": "C81FEA799A75",`
	`"Serial": "18WZ03300TRG1",`
	`"Product": "700512392",`
	`"Model": "J129",`
	`"Hardware": "0",`
	`"ResellerOrderId": "Stanley Carpets",`
	`"AvayaOrderId": "A1S-123429834",`
	`"CarrierTrackingId": "FedEx",`
	`"ShipmentCarrier": "123098712323kasdf",`
	`"ownerAccount": "StarFish_Arrow3",`
	`"site": "ssa",`

	"provisioningURL": "https://des.avaddya.com", "isActive": false, "isLocked": false, "firmwareVersion": null, "labels": [] }
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

5. Device V2 API to get device details along with ACO Product ID

Description	• Get Device details using either MAC or Serial or Both. • Either of the value is mandatory to get the Device details. • If fields (**Reseller Order ID, Avaya Order ID, Shipment Carrier, Carrier Tracking ID**) are populated against device during Bulk operation then respective fields and values will be part of get device response. • This v2 API will return ACO Product Id mapped with device Product ID
URL	**/v2/resources/inventory;MAC={MAC};Serial={Serial}**
Method	**GET**
URL Params	**Required:** **MAC**= MAC address of the Device wants to get details. **Serial**= Serial number of the Device wants to get details. example:

	Using only MAC https://des.avaya.com/v1/resources/inventory;MAC=010000000001 Using only Serial Number https://des.avaya.com/v1/resources/inventory;Serial=F9M0V3K9UC4W Using both MAC & Serial Number https://des.avaya.com/v1/resources/inventory;MAC=010000000001;Serial=F9M0V3K9UC4W
Request Content Type	**NA**
Response Content Type	**application/json**
Request Data	NA

Success Response, example	**Code**: 200
	Content:
	
	{
	"**MAC**": "[*HEX*]",
	"**Serial**": "[*ALPANUMERIC*]",
	"**Product**": "[*ALPANUMERIC*]",
	"**Model**": "[*ALPANUMERIC*]",
	"**Hardware**": "[*ALPANUMERIC*]",
	"**ACOProduct**": "[*ALPANUMERIC*]",
	"**ResellerOrderId**": "[*ALPANUMERIC*]",
	"**AvayaOrderId**": "[*ALPANUMERIC*]",
	"**CarrierTrackingId**": "[*ALPANUMERIC*]",
	"**ShipmentCarrier**": "[*ALPANUMERIC*]",
	"**ownerAccount** ": "[*ALPANUMERIC*]",
	"**site**" : "[*ALPANUMERIC*]",
	"**provisioningURL**": "[*URL*]",
	"**isActive**": [*TRUE/FALSE*],
	"**isLocked**" : [*TRUE/FALSE*],
	"**firmwareVersion**":"[*ALPANUMERIC*]",
	"**labels**": "[*ALPANUMERIC*]"
	}

	e.g.
	{
	"MAC": "C81FEA799A75",
	"Serial": "18WZ03300TRG1",
	"Product": "700512392",
	"Model": "J129",
	"Hardware": "0",
	"ACOProduct": "408966",
	"ResellerOrderId": "Stanley Carpets",
	"AvayaOrderId": "A1S-123429834",
	"CarrierTrackingId": "FedEx",
	"ShipmentCarrier": "123098712323kasdf",
	"ownerAccount": "StarFish_Arrow3",
	"site": "ssa",
	"provisioningURL": "https://des.avaddya.com",
	"isActive": false,
	"isLocked": false,
	"firmwareVersion": null,
	"labels": []
	}

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

6. Lookup device

Description	• Device lookup across DES inventory irrespective of device claimed or unclaimed. • Device support multiple device lookup using Json having a list of objects containing either MAC or Serial or Both • Response of the API would be a json with a list of objects containing MAC, Serial, Model Number and Product id.
URL	**/v1/resources/inventory/getdevices**

Method	POST
URL Params	**NA**
Request Content Type	**application/json**
Response Content Type	**application/json**

Request Data	[
	{
	"MAC": "C81FEA704D95",
	"Serial": "18WZ04400K6K"
	},
	{
	"MAC": "C81FEA704D97"
	},
	{
	"Serial": "18WZ04400K6N"
	},
	{
	"MAC”: "020202020300"
	}
	]

Success Response, example	**Code**: 200 **Content**: { “phones”:[{ "MAC": "[*HEX*]", "Serial": "[*ALPANUMERIC*]", "Product": "[*ALPANUMERIC*]", "Model": "[*ALPANUMERIC*]", }, { "MAC": "[*HEX*]", "Serial": "[*ALPANUMERIC*]", "ErrorCode": "[*ALPANUMERIC*]", "ErrorMessage": "[*ALPANUMERIC*]" }] }

	e.g
	{
	"phones": [
	{
	"MAC": “C81FEA704D92”,
	"Serial": "18WZ04400K6K",
	"ErrorCode": "DES_1090",
	"ErrorMessage": "The Device does not exist in DES inventory. "
	},
	{
	"MAC": "C81FEA704D97",
	"ErrorCode": "DES_1090",
	"ErrorMessage": "The Device does not exist in DES inventory. "
	},
	{
	"Serial": "18WZ04400K6N",
	"ErrorCode": "DES_1090",
	"ErrorMessage": "The Device does not exist in DES inventory. "
	},
	{
	"MAC": "020202020300",
	"Serial": "7DBRFSPFOUBS",
	"Model": "J129",
	"Product": "700512392"
	},
	{
	"MAC": "010000000409",
	"Serial": "JKL5QSXPDHU0",
	"Model": "G430V3",
	"Product": "700512392"
	},
	{
	"MAC": "010000000406",
	"Serial": "QSMTJNJ51GDU",
	"Model": "G430",
	"Product": "700512392"
	}
	]
	}

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for

	support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

7. Associate device with account

Description	• API update Account details with associating Devices with the Account. • Only Claimed Devices can be associated with Account. • Device can be associated using either only MAC or only Serial number or MAC & Serial number both. • If more than one devices are used in request data for associating with Account, DES schedule a job the bulk operation. A Job name will be return for the bulk operation request.
URL	**/v1/resources/account/{accountId}/associateDevices**
Method	**PUT**

URL Params	**Required:**
	`accountId=[numeric]`
	example:
	`https://des.avaya.com/v1/resources/account/12345678/associateDevices`

Request Content Type	**application/json**
Response Content Type	**text/plain**

Request Data	• **Associate Devices to Reseller.**
	{
	    "desDevices":[

```
          {
            "MAC":"[HEXDECIAL]”,
            "Serial":"[ ALPANUMERIC]”
             "ResellerOrderId": "[ ALPANUMERIC]",
              "AvayaOrderId": "[ ALPANUMERIC]",
              "CarrierTrackingId": "[ ALPANUMERIC]",
              "ShipmentCarrier": "[ ALPANUMERIC]"
          }
        ]
}
```

e.g

```
{
   "desDevices":[
        {
           "MAC":"010000000002",
           "Serial":"E0W8M24E2C30",
          "resellerOrderId": "StanleyCarpet",
           "avayaOrderId": "A1S-123429834",
           "shipmentCarrier": "FedEx",
           "carrierTrackingId":"87123kasdf"
        }
    ]
}
```

Associate Multiple Devices.

```
{
   "desDevices":[
        {
           "MAC":"010000000001",
           "Serial":"E0W8M24E2C31",
          "resellerOrderId":"StanleyCarpet",
           "avayaOrderId":"A1S-123429834",
           "shipmentCarrier":"FedEx",
           "carrierTrackingId":"87123kasdf"
        },
        {
           "MAC":"010000000002",
           "Serial":"E0W8M24E2C32"
        }
    ]
}
```

- **Associate Devices to customer / site.**

```
{
  "desCustomerSites":[
    {
      "siteLocation":"[ ALPANUMERIC]",
      "desDevices":[
        {
          "MAC":"[HEXDECIAL]”,
          "Serial":"[ ALPANUMERIC]”
           "ResellerOrderId": "[ ALPANUMERIC]",
            "AvayaOrderId": "[ ALPANUMERIC]",
            "CarrierTrackingId": "[ ALPANUMERIC]",
            "ShipmentCarrier": "[ ALPANUMERIC]"
        }
      ]
    }
  ]
}
```

e.g

```
{
   "desCustomerSites":[
      {
         "siteLocation":"devices-sv-site",
         "desDevices":[
            {
               "MAC":"010000000002",
               "Serial":"E0W8M24E2C30",
            "resellerOrderId":"StanleyCarpet",
            "avayaOrderId":"A1S-123429834",
            "shipmentCarrier":"FedEx",
            "carrierTrackingId":"87123kasdf"
            }
         ]
      }
   ]
}

Associate Multiple devices
{
   "desCustomerSites":[
      {
         "siteLocation":"devices-sv-site",
```

	"desDevices":[
	{
	"MAC":"010000000001",
	"Serial":"E0W8M24E2C31"
	},
	{
	"MAC":"010000000002",
	"Serial":"E0W8M24E2C32"
	}
	]
	}
	]
	}

Success Response, example	**Code**: 200 **Content**: **{** **"code":"DES_152",** **"message":"Account has been updated successfully."** **}** If more than one devices are used in request data for respective action, DES schedule a job the bulk operation. A Job name will be return for the bulk operation request e.g. **Code**: 200 **Content**: **{** **"jobName":"bulk associate_1577692090568"** **}**
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code**: 406 Not Acceptable **Please refer Appendix for all event Codes and Error Messages** OR

	Code: 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

8. Get bulk operation job status

Description	- API to get bulk operation job status and details.
URL	**/v1/resources/inventory/{jobName}**
Method	**GET**
URL Params	**Required:** **jobName** = Name of bulk operation job for which you want to get details example: https://des.avaya.com/v1/resources/inventory/job-activate-devices
Request Content Type	**NA**
Response Content Type	**application/json**
Request Data	NA
Success Response, example	**Code**: 200 **Content**: {"totalCount":1,"list":[{"id":2035,"successDevicesCount":"3","failedDevicesCount":"0","duplicateDevicesCount":"0","totalDevicesCount":"3", "percentage":"100%"}]}
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

9. Download / Export device bulk operation job details.

Description	- **API to export device details which are provided in device bulk operation job.** - **Export can be down in different file as JSON / CSV**
URL	**/v1/resources/inventory/download/{jobNAme}/{DownloadType}**
Method	**GET**
URL Params	**URL Path Params** **DownloadType: json/csv** **example:** `https://des.avaya.com/v1/resources/inventory/download/Devices-SV_associatelabels_1597334640782/csv`
Request Content Type	**NA**
Response Content Type	**application/octet-stream**
Request Data	**NA**
Success Response, example	**Code: 200** **Content:** **Actual Devices details export File in JSON / CSV format.**
Error Response	**Example:** **Code: 401 UNAUTHORIZED** **Content:** **Access Denied to resource. Please contact Avaya Administrator for support** **OR** **Code: 500 Internal Server Error** **Content** **Please refer Appendix for all event Codes and Error Messages**

10. Export selected devices

Description	- **API to export selected devices.** - **Export can be done in two file formats as json / csv** - **Once device export API is called, user can download exported devices file.**
URL	**/v1/resources/export/{fileType}**
Method	**POST**
URL Params	**URL Path Params** **File Type: json/csv** **example:** `https://des.avaya.com/v1/resources/export/json`
Request Content Type	**application/json**
Response Content Type	**application/json**
Request Data	**List of Device MAC** `[“010000000001”,” 010000000002”]`
Success Response, example	**Code: 200** **Content:** `{"jobName":"avayaadmin_1622453061104"}`
Error Response	**Example:** **Code: 401 UNAUTHORIZED** **Content:** **Access Denied to resource. Please contact Avaya Administrator for support** **OR** **Code: 500 Internal Server Error** **Content** **Please refer Appendix for all event Codes and Error Messages**

11. Export All devices

Description	• **API to export all devices** • **Export can be done in two file formats as json / csv** • **Once device export API is called, user can download exported devices file**
URL	**/v1/resources/export/all/{fileType}**
Method	**POST**
URL Params	**URL Path Params** **File Type: json/csv** **example:** `https://des.avaya.com/v1/resources/export/all/json`
Request Content Type	**application/json**
Response Content Type	**application/json**
Request Data	**Device Filter Properties – The following json object can be modified to apply filter on devices. Remove keys which are not required or keep values blank or null if not required.** **The key “filterForDevice” is mandatory and should be always set to true.** All values are in string format except following: • **filterForDevice – always true** • **active – true/false** • **synced – true/false** **{"filterForDevice":true, "macAddress": "macAddress",** **"modelNumber": "modelNumber", "serialNumber": "serialNumber",** **"accountName": "accountName", "hardware": "hardware", "associatedLabels": "associatedLabels",** **"labelName": "labelName", "shipmentCarrier": "shipmentCarrier", "carrierTrackingId": "carrierTrackingId",** **"avayaOrderId": "avayaOrderId","resellerOrderId": "resellerOrderId","manufactureDateTime": "manufactureDateTime",** **"desCustomerSite": "desCustomerSite","firmwareVersion":**

	**"firmwareVersion",
	"active": "active","synced": "synced"}**
	
	For example:
	
	{"filterForDevice":true, "macAddress": "0100000000C1", "modelNumber": "J129", "active": true}

Success Response, example	**Code: 200** **Content:** `{"jobName":"avayaadmin_1622453061104"}`
Error Response	**Example:** **Code: 401 UNAUTHORIZED** **Content:** **Access Denied to resource. Please contact Avaya Administrator for support** **OR** **Code: 500 Internal Server Error** **Content** **Please refer Appendix for all event Codes and Error Messages**

12. Get exported device job status

Description	• API to return all details of Export Device Job using job Name • Response of the api would be a json Object containing job Name, Job Status, Start Time, End Time, Total Devices Count , Failed Devices Count and file name
URL	**/v1/resources/export/jobDetails/{jobname}**
Method	GET
URL Params	**JobName** e.g https://des.avaya.com/v1/resources/export/jobDetails/avayaadmin_1622615138940

Request Content Type	**NA**
Response Content Type	**application/json**
Request Data	NA

Success Response, example	**Code**: 200
	Content:
	{
	"jobName":" [*ALPHANUMERIC*] ",
	"Status": " [*ALPHABETIC*] ",
	"startTime":" [*ALPHANUMERIC*] ",
	"endTime":" [*ALPHANUMERIC*] ",
	"totalDevicesCount": " [*NUMERIC*] ",
	"failedDevicesCount":" [*NUMERIC*] ",
	"fileName": " [*ALPHANUMERIC*] "
	}
	e.g
	{
	"jobName": "avayaadmin_1622615138940",
	"Status": "COMPLETE",
	"startTime": "2021-06-02 06:25:38",
	"endTime": "2021-06-02 06:25:39",
	"totalDevicesCount": 18,
	"failedDevicesCount": 0,
	"fileName": "avayaadmin_1622615138940"
	}

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 400 Internal Server Error **Content**

	Please refer Appendix for all event Codes and Error Messages

13. Download exported devices file

Description	- API to download exported devices file
URL	/v1/resources/export/download/{fileName}
Method	GET
URL Params	URL Path Params File Name: devices_<timestamp return by export devices API o that is timestamp appended to job_name>.zip example: `https://des.avaya.com/v1/resources/export/download /devices_1622453061104.zip`
Request Content Type	NA
Response Content Type	application/octet-stream
Request Data	NA
Success Response, example	Code: 200 Content: `File is returned to the user.`
Error Response	Example: Code: 401 UNAUTHORIZED Content: Access Denied to resource. Please contact Avaya Administrator for support OR Code: 500 Internal Server Error Content Please refer Appendix for all event Codes and Error Messages

14. Get all device types, models supported firmware versions details

Description	- **API to get all the device types and models list with supported firmware versions**
URL	**/v1/resources/devicesfamily**
Method	**GET**
URL Params	**No URL Path Params**
Request Content Type	**application/json**
Response Content Type	**application/json**
Request Data	**NA**

Success Response, example	**Code: 200**
	Content:
	
	{
	  "deviceTypesList": [{
	        "deviceType": "Vantage",
	        "deviceFamilyModels": [{
	            "modelNumber": "K155",
	            "deviceFamilyModelFwversions": [
	              "2.2.0.4.7569",
	              "latest",
	              "2.2.0.6.9505",
	              "Same As Account",
	              "2.2.0.6.9005",
	              "off"
	            ]
	          },
	          {
	            "modelNumber": "K165",
	            "deviceFamilyModelFwversions": [
	              "2.2.0.4.7069",
	              "latest",
	              "Same As Account",

	(JSON example continued below)

```
                        "2.2.0.6.9505",
                        "off",
                        "2.2.0.6.9005"
                    ]
                },
                {
                    "modelNumber": "K175",
                    "deviceFamilyModelFwversions": [
                        "2.2.0.4.7069",
                        "latest",
                        "Same As Account",
                        "2.2.0.6.9505",
                        "off",
                        "2.2.0.6.9005"
                    ]
                }
            ]
        },
        {
            "deviceType": "CU360",
            "deviceFamilyModels": [{
                "modelNumber": "CU360",
                "deviceFamilyModelFwversions": [
                    "off",
                    "latest",
                    "Same As Account",
                    "10.2.0.4"
                ]
            }]
        }
    ]
}
```

Error Response	**Example:** **Code: 401 UNAUTHORIZED** **Content:** **Access Denied to resource. Please contact Avaya Administrator for support** **OR** **Code: 500 Internal Server Error** **Content** **Please refer Appendix for all event Codes and Error Messages**

15. Upload Mac-address settings file for device

Description	- API to upload Mac-address settings file for device
URL	/v1/resources/inventory/uploadsettingsfile/{Mac-address}
Method	PUT
URL Params	URL Path Params: Mac-address of the device whose Mac-address.txt has to be uploaded. example: `https://des.avaya.com/v1/resources/inventory/uploadsettingsfile/C81FEA5C7353`
Request Content Type	Multipart/form-data
Response Content Type	application/json
Request Data	- Multipart Form data key: settingsFile - value: <Mac-address>.txt Example: - Multipart Form data key: settingsFile - value: C81FEA5C7353.txt

Success Response, example	Code: 200
	Content:
	{
	"code": "DES_1092",
	"message": "Device Settings File has been Uploaded successfully ."
	}

Error Response	Example: Code: 401 UNAUTHORIZED

	Content: **Access Denied to resource. Please contact Avaya Administrator for support** **OR** **Code: 500 Internal Server Error** **Content** **Please refer Appendix for all event Codes and Error Messages**

16. Delete Mac-address settings file for device

Description	• **API to delete Mac-address settings file against device**
URL	**/v1/resources/inventory/deletesettingsfile/{Mac-address}**
Method	**DELETE**
URL Params	**URL Path Params: Mac-address of the device whose Mac-address.txt has to be deleted.** **example:** `https://des.avaya.com/v1/resources/inventory/deletesettingsfile/C81FEA5C7353`
Request Content Type	**application/json**
Response Content Type	**application/json**
Request Data	**NA**
Success Response, example	**Code: 200** **{** **"code": "DES_1097",** **"message": "Device settings file has been deleted successfully."** **}**

Error Response	**Example:** **Code: 401 UNAUTHORIZED** **Content:** **Access Denied to resource. Please contact Avaya Administrator for support** **OR** **Code: 500 Internal Server Error** **Content** **Please refer Appendix for all event Codes and Error Messages**

Profile Management (V1 REST interface)

1. Create profile

Description	• API to create Provisioning URL • CA Certificate is requiring if Provisioning URL is HTTPS and to have successful TLS connection between device and Configuration Server. • Server Type for Provisioning URL should be either of HTTP(S) or DES.
URL	**/v1/resources/profile**
Method	**POST**
URL Params	*NA*
Request Content Type	**application/json**
Response Content Type	**application/json**

Request Params	{
	  "profileName":"*[alphanumeric]*",
	  "desProvisionUrlCerts":[
	    {
	      "caCert":*[alphanumeric]*,
	      "provisionUrl":"*[alphanumeric]*",
	      "phoneGroup":"*[numeric]*"
	      "**serverType**":"*[HTTPS/DES]*"
	      "**generateAvayaCert**": *[true/false]*
	    }
	  ]
	}
	
	e.g.
	{
	    "profileName":"Alpha_profile",
	    "desProvisionUrlCerts":[
	        {

	"caCert":null, "provisionUrl":"https://devices.prov.avaya.com", "phoneGroup":"100", "serverType":"HTTP(S)" "generateAvayaCert": true }] }
Success Response, example	**Code**: 200 **Content**: **{** **"code":"DES_163",** **"message":"Profile has been created successfully."** **}**
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

2. Get profile

Description	- Get Profile details.
URL	**/v1/resources/profile/{profileName}**
Method	**GET**
URL Params	**Required:** **profileName**=[alphanumeric] example:

	`https://des.avaya.com/v1/resources/profile/alpha_profile`
Request Content Type	*NA*
Response Content Type	**application/json**
Request Params	*NA*

Success Response, example	**Code**: 200
	Content:
	{
	"**id**":14,
	"**profileName**":"alpha_profile",
	"**desProvisionUrlCerts**":[
	{
	"**id**":320,
	"**provisionUrl**":"https://devices.prov.avaya.com",
	"**phoneGroup**":0,
	"**caCert**":null,
	"**serverType**":"HTTP(S)"
	"**generateAvayaCert**": true
	}
	]
	}

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

3. Update profile

Description	- Update Profile details.
URL	**/v1/resources/profile**
Method	**PUT**
URL Params	*NA*
Request Content Type	**application/json**
Response Content Type	**application/json**

Request Params	{
	"id":*[numeric]*,
	"profileName":"*[alphanumeric]*",
	"desProvisionUrlCerts":[
	{
	"id":*[alphanumeric]*,
	"caCert":*[alphanumeric]*,
	"provisionUrl":"*[alphanumeric]*",
	"phoneGroup":*[numeric]*
	"serverType":”HTTP(S)/DES”,
	"generateAvayaCert": *[true/false]*
	}
	]
	}

Success Response, example	**Code**: 200 **Content**: **{** **"code":"DES_164",** **"message":"Profile has been updated successfully."** **}**
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:**

	Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

4. Delete profile

Description	- Delete Profile.
URL	**/v1/resources/profile/{profileName}**
Method	**DELETE**

URL Params	**Required:**
	`profileName``=[alphanumeric]`
	example:
	`https://des.avaya.com/v1/resources/profile/alpha_profile`

Request Content Type	*NA*
Response Content Type	**application/json**
Request Params	*NA*
Success Response, example	**Code**: 200 **Content**: **{** **"code":"DES_165",** **"message":"Profile has been deleted successfully."**

	}
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

Profile Management (V2 REST interface)

5. Create profile

Description	• API to create a profile with provisioning URLs. • Server Type for Provisioning URL should be either of HTTP(S) or DES. • Phone group can be between 0 and 999
URL	**/v2/resources/profile**
Method	**POST**
URL Params	*NA*
Request Content Type	**multipart/form-data**
Response Content Type	**application/json**

Request Data	• **Multipart key: “profileJson”**
	{
	"**profileName**":"*[alphanumeric]*",
	"**desProvisionUrlCerts**": [
	{
	"**provisionUrl**":"*[alphanumeric]*",
	"**phoneGroup**":"*[numeric](0-999)*",
	"**serverType**":"*[HTTP(S)/DES]*",
	"**generateAvayaCert**": *[true/false]*
	}
	]
	}
	
	e.g.
	{
	"**profileName**":"Alpha_profile",
	"**desProvisionUrlCerts**": [
	{
	"**provisionUrl**":"https://devices.prov.avaya.com",

	`"phoneGroup":"100",`
	`"serverType":"HTTP(S)",`
	`"generateAvayaCert": true`
	`}`
	`]`
	`}`
	- **Multipart key: “{provisionUrl}||{phoneGroup}”**
	Allowed values: Valid certificate files with extensions as .crt, .pem or .cer

Success Response, example	**Code**: 200 **Content**: **{** **"code":"DES_163",** **"message":"Profile has been created successfully."** **}**
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

6. Get profile details

Description	- API to get a profile details. Response JSON return complete Profile hierarchy including provisioning URLs and its details.
URL	**/v2/resources/profile/{profileName}**
Method	**GET**

URL Params	**Required:**
	`profileName=[alphanumeric]`
	
	example:
	`https://des.avaya.com/v1/resources/profile/alpha_profile`

Request Content Type	*NA*
Response Content Type	**application/json**
Request Params	*NA*

Success Response, example	**Code**: 200
	Content:
	
	`{`
	`   "id":14,`
	`   "profileName":"alpha_profile",`
	`   "desProvisionUrlCerts":[`
	`      {`
	`         "id":320,`
	`         "provisionUrl":"https://devices.prov.avaya.com",`
	`         "phoneGroup":0,`
	`         "caCert":null,`
	`         "serverType":"HTTP(S)"`
	`         "generateAvayaCert": true`
	`      }`
	`   ]`
	`}`

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error

	Content **Please refer Appendix for all event Codes and Error Messages**

7. Update profile details

Description	• API to update provisioning URLs and its details.
URL	**/v2/resources/profile/{profileName}**
Method	**PUT**
URL Params	**Required:** **`profileName`**`=[alphanumeric]` = Name of profile to be updated example: `https://des.avaya.com/v1/resources/profile/alpha_profile`
Request Content Type	**multipart/form-data**
Response Content Type	**application/json**

Request Data	• **Multipart key: “profileJson”**
	{
	"id":*[numeric]*,
	"profileName":"*[alphanumeric]*",
	"desProvisionUrlCerts":[
	{
	"id":*[numeric]*,
	"provisionUrl":"*[alphanumeric]*",
	"phoneGroup":*[numeric]*
	"serverType":”HTTP(S)/DES”,
	"**generateAvayaCert**": *[true/false]*
	}
	]
	}

	e.g. {
	"**id**":14,
	"**profileName**":"alpha_profile",
	"**desProvisionUrlCerts**":[
	{
	"**id**":320,
	"**provisionUrl**":"https://devices.prov.avaya.com",
	"**phoneGroup**":0,
	"**serverType**":"HTTP(S)"
	"**generateAvayaCert**": false
	}
	]
	}
	- **Multipart key: "{provisionUrl}||{phoneGroup}"**
	
	Allowed values: Valid certificate files with extensions as .crt, .pem or .cer

Success Response, example	**Code**: 200 **{** **"code":"DES_152",** **"message":"Account has been updated successfully."** **}**
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

8. Delete profile

Description	• API delete profile details along with the provisioning URLs and its details. • If a provisioning URL is associated with a customer site, the deletion of a profile will remove the association of customer site with provisioning URL
URL	**/v2/resources/profile/{profileName}**
Method	**DELETE**
URL Params	**Required:** `profileName=[alphanumeric]` = Name of profile to be deleted example: `https://des.avaya.com/v1/resources/profile/alpha_profile`
Request Content-Type	*NA*
Response Content-Type	**application/json**
Request Params	*NA*
Success Response, example	**Code**: 200 **Content**: **{** **"code":"DES_165",** **"message":"Profile has been deleted successfully."** **}**
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error

	Content **Please refer Appendix for all event Codes and Error Messages**

Numeric Enrollment Code (REST interface)

1. Create Numeric Enrollment Code

<table>
<tr><td><b>Description</b></td><td>• Create NEC against Customer Account for specific Customer Site</td></tr>
<tr><td><b>URL</b></td><td><b>/v1/resources/necpolicy</b></td></tr>
<tr><td><b>Method</b></td><td><b>POST</b></td></tr>
<tr><td><b>URL Params</b></td><td><i>NA</i></td></tr>
<tr><td><b>Request Content Type</b></td><td><b>application/json</b></td></tr>
<tr><td><b>Response Content Type</b></td><td><b>application/json</b></td></tr>
<tr><td><b>Request Params</b></td><td>{
  “<b>necPolicyName</b>”:”<i>[alpha]</i>”,
  "<b>expiryType</b>":"<i>[alpha]</i>",
  <b>"expiry”:</b> <i>[numeric]</i>,
  <b>"devicesTobeEnrolled"</b>:<i>[numeric]</i>,
  <b>"customerId”:</b>
  {
    <b>"accountName"</b>:"<i>[alphanumeric]</i>"
  },
  <b>"siteId"</b>:
  {
    <b>"siteLocation"</b>:"<i>[alphanumeric]</i>"
  }
}

<b>expiryType</b>: Should be one of {
<b>{Weeks/Week/Days/Day/Hours/Hour/Never Expire}</b>
<b>expiry</b>: Should be an Integer value.
<table>
<tr><td>expiryType</td><td>expiry</td></tr>
<tr><td>Hour/Hours</td><td>1-72</td></tr>
<tr><td>Day/Days</td><td>1-90</td></tr>
</table>
</td></tr>
</table>

Week/Weeks	1-12
Month/Months	1-3
Never Expire	-1

devicesTobeEnrolled: Should be an Integer value and the maximum limit is 999999.

```
e.g.
{
   "necPolicyName":"Devices-SV_desices-sv-
cust_rahul",
   "expiryType":"Weeks",
   "devicesTobeEnrolled":1000,
   "customerId":{
      "accountName":"desices-sv-cust"
   },
   "siteId":{
      "siteLocation":"rahul"
   },
   "expiry":-1
}
```

```
e.g.: expiryType given as Never Expire

{
   "necPolicyName":"Devices-SV_Custmoer_
test_site",
   "expiryType":"Never Expire",
   "devicesTobeEnrolled":1000,
   "customerId":{
      "accountName":"desices-sv-cust"
   },
   "siteId":{
      "siteLocation":"test_site"
   },
   "expiry":-1,
   "necCode":"31484642"
}
```

Success Response, example

Code: 200
Content:

```
{
"enrollmentCode":"31484642"
```

	}
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

2. Get Numeric Enrollment Code

Description	- Get NEC Policy details.
URL	**/v1/resources/necpolicy/{necCode}**
Method	**GET**

URL Params	**Required:**
	`neccode=[numeric]`
	
	example:
	`https://des.avaya.com/v1/resources/necpolicy/12345678`

Request Content Type	*NA*
Response Content Type	**application/json**
Request Params	*NA*
Success Response, example	**Code**: 200 **Content**:

	{
	"**necCode**":"31484642",
	"**necPolicyName**":"alpha system nec",
	"**custName**":"alpha system",
	"**siteLocation**":"alpha site",
	"**totalDevicesToBeEnrolled**":1000,
	"**enrolledDevicesCount**":0,
	"**expiryTimeLimit**":1,
	"**expiryType**":"Week",
	"**isExpired**":"false",
	}

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

3. Update Numeric Enrollment Code policy

Description	- Update NEC Policy details. Only Expiry Type, Number of devices to Enrolled & Expiry
URL	**/v1/resources/necpolicy/{necCode}**
Method	**PUT**
URL Params	**Required:** **neccode**=[numeric] example: https://des.avaya.com/v1/resources/necpolicy/12345678
Request Content	**application/json**

Type	
Response Content Type	**application/json**

Request Params	{
	  "necPolicyName":"*[alphanumeric]*",
	  "expiryType":"*[alphabetic]*",
	  "expiry":*[numeric]*,
	  "devicesTobeEnrolled":*[numeric]*,
	  "customerId":{
	    "accountName":"*[alphanumeric]*"
	  },
	  "siteId":{
	    "siteLocation":"*[alphanumeric]*"
	  },
	  "necCode":"*[numeric]*"
	}
	
	expiryType: Should be one of {**Week/Days/Day/Hours/Hour**}
	expiry: Should be an Integer value and the maximum expiry limit is 1 week
	devicesTobeEnrolled: Should be an Integer value and the maximum limit is 999999.
	
	e.g.
	{
	   "necPolicyName":"Devices-SV_Custmoer_test_site",
	   "expiryType":"Week",
	   "devicesTobeEnrolled":1000,
	   "customerId":{
	      "accountName":"desices-sv-cust"
	   },
	   "siteId":{
	      "siteLocation":"test_site"
	   },
	   "expiry":1,
	   "necCode":"31484642"
	}

Success Response, example	**Code**: 200 **Content**:

	{
	"code":"DES_604",
	"message":"Enrollment code policy has been updated successfully."
	}

Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

4. Delete Numeric Enrollment Code

Description	• Delete NEC against Customer Account for specific Customer Site
URL	**/v1/resources/necpolicy/{necCode}**
Method	**DELETE**
URL Params	**Required:** **necode**=[alphanumeric] example: `https://des.avaya.com/v1/resources/necpolicy/12345678`
Request Content Type	*NA*
Response Content Type	**application/json**

Request Params	*NA*
Success Response, example	**Code**: 200 **Content**: **{** **"code":"DES_605",** **"message":"Enrollment code policy has been deleted successfully."** **}**
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

5. Regenerate Numeric Enrollment Code policy

Description	• This API Regenerate NEC for customer site based on the existing NEC Policy. • After regeneration existing NEC will be no longer exists.
URL	**/v1/resources/necpolicy/{necCode}/regenerate**
Method	**PUT**

URL Params	**Required:**
	`neccode=[numeric]`
	
	example:
	`https://des.avaya.com/v1/resources/necpolicy/12345678`

Request Content	**application/json**

Type	
Response Content Type	**application/json**
Request Params	NA
Success Response, example	**Code**: 200 **Content**: **{** **"enrollmentCode":"64762085"** **}**
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR **Code:** 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

Masquerading DES API (REST interface)

1. Raise Masquerading request access

Description	- API for raising Complete Hierarchy request. By default, masquerading access is not permitted for Service Provider. - Request access needs to be raised to allow to Switch / masquerade as Reseller account.
URL	**/v1/resources/hierarchyaccess**
Method	**POST**
URL Params	*NA*
Request Content Type	*NA*
Response Content Type	**application/json**
Request Params	*NA*
Success Response, example	**Code**: 200 **Content**: **{** **" DES_310 ":" Hierarchy access request has been created successfully."** **}**
Error Response	**Example:** **Code:** 401 UNAUTHORIZED **Content:** Access Denied to resource. Please contact Avaya Administrator for support OR

	Code: 500 Internal Server Error **Content** **Please refer Appendix for all event Codes and Error Messages**

2. DES API with Masquerading feature.

Description	• For sending any DES API as a masquerading request, a respective Reseller DES Account ID needs to be added in HTTP header. • Rest all API signature and response remain same as describe in respective API.
HTTP Header	`MASQUERADE-ACCOUNT-ID: {DES_Account_Id}`

Exceptions

Device Enrollment Services REST API methods support CRUD operation for one or more records. The returned object of exposed Web methods has status details containing the error or info message depending on the operation status.

For error code, see Appendix A.

Appendix A – Error Code

This appendix lists the specific error codes methods supported in the UM WS.

```
DES_001=Null or empty URL received.
DES_002=Provisioning URL {0} is invalid.
DES_003=Null or empty email ID received.
DES_004=Email ID {0} is invalid.
DES_005=Profile name {0} already exists.
DES_006=Exception occurred.
DES_007=Null JSON received.
DES_008=Profile has already been deleted or profile with name {0} does not
exist.
DES_009=Null or empty profile name received.
DES_010=Device has already been deleted or device with provided details does
not exist.
DES_011=Latest input for manufacturing date and time required to update device
details.
DES_012=Only one provisioning URL is allowed per profile.
DES_013=Only one provisioning URL is allowed per customer site location.
DES_014=Profile ID to be updated is null in JSON.
DES_015=Cannot load properties from file '{0}'.
DES_016=Cannot create profile: At least one provisioning URL is required.
DES_018=Pagination information is insufficient or invalid.
DES_019=Permission for {1} operation denied on {0}. Please contact Avaya
administrator.
DES_020=Certificate is invalid.


DES_021=Email ID is null or empty.
DES_022=First name is null or empty.
DES_023=Last name is null or empty.
DES_024=Password is null or empty for account administrator {0}.
DES_025=MAC address is null or empty.
DES_026=Serial Number is null or empty.
DES_027=Customer site location is null or empty.
DES_028=Login name is null or empty.
DES_029=Device with MAC address {0} and serial number {1} does not exist.
DES_030=Permission denied. Only attributed users can be deleted.
DES_031=Cannot delete account {0}. It has attributed users.
DES_032=Login name cannot exceed 255 characters.
DES_033=Email ID cannot exceed 255 characters.
DES_034=First name cannot exceed 255 characters.
DES_035=Last name cannot exceed 255 characters.
DES_036=Password cannot exceed 255 characters.
DES_037=Customer site location cannot exceed 255 characters.

DES_038=No account information
DES_039=Account name is null or empty.
DES_040=Account ID not found.
DES_043=Business phone number field is null or empty.
DES_045=Account name cannot exceed 255 characters.
```

```
DES_049=Business phone number field cannot exceed 255 characters.
DES_051=Address Line 1 field cannot exceed 255 characters.
DES_052=City field cannot exceed 255 characters.
DES_053=State/Province field cannot exceed 255 characters.
DES_054=Country field cannot exceed 255 characters.
DES_055=ZIP/Postal code field cannot exceed 255 characters.
DES_056=Address Line 2 field cannot exceed 255 characters.
DES_057=Address Line 1 field is null or empty.
DES_058=City field is null or empty.
DES_059=State/Province field is null or empty
DES_060=Country field is null or empty.
DES_061=ZIP/Postal code field is null or empty.
DES_062=Cannot enable/disable two factor authentication for non-owned
accounts.
DES_063=Cannot enable/disable two factor authentication for other users.
DES_064=Cannot disable two factor authentication if enabled for the company
account.
DES_065=Account creation has been requested successfully.
DES_066=Cannot request account creation.
DES_067=Account name {0} already exists. Please try using other account name.
DES_068=Email ID {0} already exists. Please try using other email ID.
DES_069=Could not get account sign up request.
DES_070=Login name {0} already exists. Please try using other login name.
DES_071=Invalid account type name.
DES_072=Account request is invalid.
DES_073=Devices not associated successfully.
DES_074=Customer site does not exist with provided details.
DES_075=Cannot get site details.
DES_076=Cannot get site bulk jobs.
DES_077=Invalid account desUniqueId provided.
DES_078=Invalid profile details.
DES_079=The owner account of all the customer accounts should be same.
DES_082=Basic provisioning file is required for https://des.avaya.com
provisioning URL.
DES_081=Account and site details not found to associate profile.
DES_080=Cannot get site jobs details.
DES_083=Invalid site location provided.
DES_084=Only one Avaya Administrator created customer is allowed for bulk site
update.
DES_085=Hierarchy Accesss updation request is invalid


DES_151=Account has been created successfully.
DES_152=Account has been updated successfully.
DES_153=Account has been deleted successfully.
DES_154=Account could not be created.
DES_155=Account could not be updated.
DES_156=Account could not be deleted.
DES_157=Devices have been imported successfully.
DES_158=Some devices could not be imported.
DES_257=Sites have been updated successfully.
DES_258=Some sites could not be updated.
DES_159=Device has been updated successfully.
DES_160=Device could not be updated.
DES_161=Device has been deleted successfully.
```

```
DES_162=Device could not be deleted.
DES_163=Profile has been created successfully.
DES_164=Profile has been updated successfully.
DES_165=Profile has been deleted successfully.
DES_166=Configuration has been created successfully.
DES_167=Configuration has been updated successfully.
DES_168=Configuration has been deleted successfully.
DES_169=User has been created successfully.
DES_170=User has been updated successfully.
DES_171=User(s) have been deleted successfully.
DES_172=All accounts has been deleted successfully.
DES_173=All accounts could not be deleted.
DES_174=Invalid URL.
DES_175=Firmware download request: {0}
DES_176=Firmware version bundle has been uploaded successfully.
DES_177=Firmware version bundle could not be uploaded.
DES_178=Device type is required to upload firmware version bundle.
DES_179=File to upload firmware version bundle was not received.
DES_180=Invalid firmware version bundle.
DES_181=Sign-up request Approved.
DES_182=desAccountUsers is not a valid field.
DES_183=desProfiles is not a valid field.
DES_184=desOwnerAccount is not a valid field.
DES_185=desDevices is not a valid field.
DES_186=deviceLabels is not a valid field.
DES_187=desNecPolicies is not a valid field.
DES_188=desNecPoliciesForCustomer is not a valid field.
DES_189=desEmailSettings is not a valid field.
DES_191=No email notification setting found for user {0}.
DES_192=Email notification setting has been updated successfully.
DES_193=Email notification setting could not be updated.
DES_194=Cannot get email notification setting details.
DES_198=Cannot export devices.
DES_195=Cannot update account name.
DES_196=Cannot update self account name.
DES_197=Cannot update Avaya Administrator account name.
DES_199=Device is not authorized to download basic provisioning settings file.


DES_200=Bad Request.
DES_201=Password is invalid. Password should match the password policy.
DES_202=Authz request processing is encountering an error.
DES_203=Invalid password: {0}
DES_204=No account found for user: {0}.
DES_205=No LDAP configuration exists.
DES_206=The LDAP configuration could not be updated.
DES_207=Multiple accounts linked to LDAP user.
DES_208=The LDAP configuration could not be deleted.
DES_209=Authentication failed or LDAP URL is invalid.
DES_210=Connection and Authentication successful.
DES_211=Session timeouts cannot be less than {0} seconds.
DES_212=Maximum session timeout cannot be less than Idle session timeout.
DES_213=Maximum user session settings must be between {0} and {1}.
DES_214=Provision Url "https://des.avaya.com" requires Server Type "DES"
DES_215=Server Type "DES" requires Provision Url "https://des.avaya.com"
```

```
DES_216=Server Type other than "DES" or "HTTP(S)" are not accepted.
DES_217=Avaya Certificate Generation flag can be set true for only HTTP(S)
URLs.
DES_218=Cannot get device lockout settings.
DES_219=Invalid device lockout time value, it should be between 5 to 60
minutes.
DES_220=Invalid enrollment code invalid attempts value, it should be between 3
to 10.
DES_221=Cannot update device lockout settings.
DES_222=Cannot export audit logs.

DES_250=Report type is invalid.

DES_301=Account type is invalid.
DES_302=Account is not of Manufacturer type.
DES_303=Account is not of Service Provider or Reseller type.
DES_304=Account is not of Avaya Admin type.
DES_305=Customer Site has been updated successfully.
DES_306=Account is not of Avaya Services type.
DES_307=Account is not of Avaya Admin or Auditor type.
DES_308=Hierarchy access is already requested.
DES_309=Hierarchy access request is already approved.
DES_310=Hierarchy access request has been created successfully.
DES_311=Cannot create hierarchy access request
DES_312=Hierarchy request has been approved successfully.
DES_313=Hierarchy access request is already declined.
DES_314=Error while performing action on hierarchy access request.
DES_315=Unable to approve hierarchy access request.
DES_316=Unable to decline hierarchy access request.
DES_317=Hierarchy request has been declined successfully.
DES_320=No Hierarchy access request found.
DES_321=Logged in user does not have permission to masquerade.
DES_322=Reseller account does not exist.
DES_323=Error while masquerading. Please contact Avaya Administrator.
DES_351=Account is not of Auditor type.
DES_353=Account is not allowed to perform sync operation
DES_354=Sync operation cannot be accompanied by other operations in device
bulk operation functionality.
DES_356=Device with given mac address or serial number does not exist.
DES_357=Device could not be claimed during Sync operation.
DES_358=Device could not be activated during Sync operation.
DES_359=Device could not be associated during Sync operation.
DES_361=Device syncing failed.
DES_362=Device doesn't have a reseller order id associated to it.
DES_363=Customer Firmware Settings has been updated successfully.
DES_364=Device synced successfully.

DES_400=Firmware version setting already exists with device type {0}.
DES_401=Firmware version setting added successfully.
DES_402=Firmware version setting has been updated successfully.
DES_403=Firmware version setting has been deleted successfully.
DES_404=Firmware version setting could not be updated.
DES_405=Firmware version setting could not be deleted.
DES_406=Firmware version setting could not be added.
DES_407=Global firmware URL is invalid. Please provide a valid firmware URL.
```

```
DES_500=Account name {0} already exists.
DES_501=Internal error occurred.
DES_502=Account does not exist.
DES_503=Invalid email ID {0}.
DES_504=Profile name {0} already exists.
DES_505=Exception occurred.
DES_506=User login name {0} already exists.
DES_507=Account could not be created.
DES_508=Account could not be updated.
DES_509=Account could not be deleted.
DES_510=Account with ID {0} does not exist or has already been deleted.
DES_511=Cannot delete Avaya administrator account.
DES_514=Cannot disable Avaya administrator account.
DES_515=Cannot delete logged in user {0}.
DES_516=User update failed. Cannot delete logged in user {0}.
DES_517=User update failed. Cannot delete user {0}.
DES_518=Account with ID {0} could not be deleted. The account does not exist
or has already been deleted.
DES_519=User could not be created.
DES_520=User could not be updated.
DES_521=User could not be deleted.
DES_522=User could not be updated. User {0} does not exist.
DES_523=User could not be deleted. User {0} does not exist.
DES_524=Account could not be created. Cannot create account with type 'Avaya
Admin'.
DES_525=Device(s) could not be associated. Every customer site location must
be associated with a configuration service profile for device(s) association.
DES_526=User {0} does not exist.
DES_527=Cannot change avaya administrator's login name.
DES_528=Cannot change user details as user is disabled.
DES_529=User name is null or empty.
DES_530=OTP Access key cannot be updated.
DES_531=OTP Start time cannot be updated.
DES_532=Validation private key cannot be updated.
DES_533=Validation public key cannot be updated.
DES_534=Account type is incorrect.
DES_535=Account does not exist.
DES_536=Cannot add user as account {0} is disabled.
DES_537=Error while sending email. Check SMTP server configuration.
DES_538=Unauthorized logged in user trying to send bulk user emails.
DES_539=Cannot send empty email to users.
DES_540=Email FROM field cannot be empty.
DES_541=Email TO field cannot be empty.
DES_542=Cannot create auditor account without logged in user type as 'Avaya
Admin'.
DES_543=Is Active column does not support 'Starts With', 'Ends With' or
'Contains' operators.
DES_544=Please check the filters entered. Incompatible values.
DES_545=Please click on "Show Track Devices" and then use client ip filter in
Advanced Search.
DES_546=Is Active filter cannot have values other than "true" or "false".
DES_547=Date-Time columns do not support "Contains" operator.
DES_548=Could not remove TOTP configuration of user {0}.
DES_549=Could not add CONFIGURE_TOTP required action for user {0}.
```

```
DES_550=Cannot associate devices to account which is not created by logged in
user account.
DES_551=BP Link ID does not match.
DES_552=BP Link ID is not valid.
DES_553=Request is already sent for account linking.
DES_554=Account linking request has already been approved.
DES_555=Account linking request has been successfully sent.
DES_556=Reseller account with provided BP Link ID does not exist.
DES_557=BP Link ID is empty.
DES_558=Account has been successfully linked.
DES_559=Account linking could not be approved.
DES_560=BP Link ID {0} already exists.
DES_561=Account could not be deleted as it was not created by logged in
account.
DES_562=Cannot get account details.
DES_563=Cannot get account hierarchy details.
DES_564=Email sent successfully.
DES_565=Cannot link with reseller account.
DES_566=Customer Site couldn't be updated.
DES_567=Invalid reseller account for linking.
DES_568=Cannot set Provision ID without site associated with a provisioning
url.
DES_569=Customer site does not support multiple provisioning urls.
DES_570=Invalid action to update an account.
DES_571= SAP Sold to Number/ Functional Location (FL) does not match.
DES_572= SAP Sold to Number/ Functional Location (FL) is not valid.
DES_573= SAP Sold to Number/ Functional Location (FL) {0} already exists.
DES_574= SAP Sold to Number/ Functional Location (FL) is empty.
DES_575=Reseller account with provided SAP Sold to Number/ Functional Location
(FL) does not exist.
DES_576=BP Link ID cannot be assigned in a customer signup request.
DES_577= SAP Sold to Number/ Functional Location (FL) cannot be assigned in a
non-customer signup request
DES_578=User not permitted to filter audit logs on event code.
DES_579=User not permitted to filter audit logs on serial number.
DES_580=Customer Site deleted successfully.
DES_581=Customer Site does not exist.
DES_590=Customer Site created successfully.
DES_591=Customer Site already exists.
DES_592=Customer Site could not be created.
DES_593=Cannot add basic provisioning settings file to a site with non DES
type URL.
DES_594=The basic provisioning settings file size cannot exceed {0} MB.
DES_595=SAP Sold to Number/Functional location is not valid, length is greater
than 10 digits.
DES_596=SAP Sold to Number/Functional location is not applicable for this
account type.
DES_597=BP Link ID is not applicable for this account type.
DES_598=BP Link ID is not valid, length should be less than or equal to 10
digits for Customer account.
DES_599=desCustomerSites is not a valid field.

DES_600=Enrollment code policy name already exists.
DES_601=Enrollment code policy has been created successfully.
DES_603=Enrollment code policy could not be created.
```

```
DES_604=Enrollment code policy has been updated successfully.
DES_602=Enrollment code policy has already been created for this customer
account and customer site location.
DES_605=Enrollment code policy has been deleted successfully.
DES_606=Enrollment code policy could not be updated.
DES_607=Enrollment code policy could not be deleted.
DES_608=Customer site location is invalid for the specified customer.
DES_609=Enrollment code policy has been regenerated successfully.
DES_610=Enrollment code policy could not be regenerated.
DES_611=Enrollment code policy expiry time limit cannot be more than 12 weeks
or less than equal to 0 weeks.
DES_612=Enrollment code policy expiry time limit cannot be more than 90 days
or less than equal to 0 days.
DES_613=Enrollment code policy expiry time limit cannot be more than 72 hours
or less than equal to 0 hours.
DES_614=Enrollment code policy could not be added as customer account {0} is
disabled.
DES_615=Cannot update enrollment code policy as customer account {0} is
disabled.
DES_616=Cannot regenerate enrollment code policy as customer account {0} is
disabled.
DES_617=Cannot generate enrollment code policy as customer account {0} and
customer site location {1} is not associated with any configuration service
profile.
DES_618=Cannot regenerate enrollment code policy as customer account {0}
customer site location {1} is not associated with any configuration service
profile.
DES_619=Cannot update enrollment code policy as customer account {0} customer
site location {1} is not associated with any configuration service profile.
DES_620=Cannot update enrollment code policy as the enrollment code has
expired. Please regenerate the enrollment code to update any details.
DES_621=Maximum number of devices for which enrollment code is generated has
exceeded.
DES_622=Cannot delete enrollment code policy as it does not belong to account
{0}.
DES_623=Cannot update enrollment code policy as it does not belong to account
{0}.
DES_624=Cannot regenerate enrollment code policy as it does not belong to
account {0}.
DES_625=Invalid expiry type for enrollment code policy.
DES_626=Enrollment code policy expiry time limit cannot be more than 3 months
or less than equal to 0 months.
DES_627=The account {0} does not belong to the logged in account to add
enrollment code policy.
DES_628=Enrollment code policy could not be deleted as customer account {0} is
disabled.
DES_629=Enrollment code policy expiry time limit not applicable for expiry
type 'NEVER'.
DES_630=Error while aadding/updating firmware details.
DES_631=Invalid job name.
DES_650=Cannot use advanced filter for the logged in account type.
DES_651=Enrollment code length is invalid, it should be 8 or 12 digit.
DES_652=Enrollment code length is required.
DES_653=Firmware flag is disabled for this account.
DES_654=Firmware Settings is only applicable to customer accounts.
```

```
DES_655=Can not get firmware settings.
DES_656=Device Family and Model with the given firmware version is not
present.
DES_657=Customer Firmware Settings couldn't be updated.
DES_658=Logged in account does not own this customer.
DES_659=Customer with this id does not exist.
DES_660=DesUniqueId is not provided.
DES_661=Device Family not provided.
DES_662=Device model not provided.
DES_663=Firmware version not provided.
DES_664=Provisioning URL should be associated to site.


DES_800=Device Type {0} doesn't support open firmware download.
DES_801=Model Number {0} provided in user agent is not supported by DES.
DES_802=User Agent {0} is invalid or not supported by DES.


DES_900=Cannot upload export devices file.
DES_901=Cannot download export devices file.
DES_902=Invalid file type.
DES_903=File not found.
DES_904=Cannot upload basic provisioning settings file.
DES_905=Cannot download basic provisioning file.
DES_906=Provisioning URL must only be "https://des.avaya.com".
DES_907=Invalid basic provisioning settings file {0}.
DES_908=Invalid CSR format.
DES_909=Cannot get basic provisioning settings file {0}.
DES_910=The file size cannot exceed 50 KB.
DES_911=Not Modified
DES_912=Could not parse the HTTP header if-modified-since
DES_913=Could not obtain file meta data.
DES_914=Could not delete file from EFS.
DES_915=Invalid basic provisioning settings file {0}. 46xxsettings file should
be in lower case.
DES_916=Invalid basic provisioning settings file. Zip file should not contain
any folder.
DES_917=Too big file. Please provide a File less than 1 MB in size.
DES_918=Invalid settings file name. The Settings file name must match with mac
address provided in the URL.

DES_920=Device family support has been added successfully.
DES_921=Cannot add support for device type {0}.
DES_922=Device type {0} already exists.
DES_923=Device type support has been deleted successfully.
DES_924=Cannot delete support for device type {0}.
DES_925=Device type {0} does not exist.
DES_926=Device family support has been updated successfully.
DES_927=Cannot update support for device type {0}.
DES_928=Model Number {0} does not exist.
DES_929=Cannot delete support for model number {0}.
DES_930=Device model support has been deleted successfully.
DES_931=changing model wise firmware support from true to false is not
allowed.
DES_932=Device type is required.
```

```
DES_933=Device firmware version(s) have been deleted successfully.
DES_934=Invalid device firmware version(s) provided.
DES_935=ModelWiseFirmwareUpgrade value should be false to delete firmware
version of all model numbers under the device type provided.
DES_937=Firmware version {0} is going to be deprecated from DES, please use
different firmware version.
DES_936=The firmware version {0} is already marked for deletion.

DES_941=Device label {0} already exists for account {1}.
DES_942=Device label has been added successfully.
DES_943=Cannot add device label.
DES_944=Device label has been deleted successfully.
DES_945=Device label {0} does not exist.
DES_946=Cannot delete device label.

DES_948=Device labels have been associated successfully.
DES_949=Device labels could not be associated.
DES_950=Device label {0} already associated with device.
DES_951=Device label has been dissociated successfully.
DES_952=Device label could not be dissociated.
DES_953=Device label(s) are not associated with device having MAC address {0}.
DES_954=Label name is not valid.
DES_955=Label Name is not provided.


DES_956=Account Name can contain only letters (a-zA-Z), numbers(0-9), special
characters(_, $, &, +, :, ;, =, ?, @, #, ', <>, [], ^, *, (), %, !, {}, -),
commas(,), spaces, periods(.).
DES_957=Profile Name can contain only letters (a-zA-Z), numbers(0-9), special
characters(_, $, &, +, :, ;, =, ?, @, #, ', <>, [], ^, *, (), %, !, {}, -),
commas(,), spaces, periods(.).
DES_958=Site Name can contain only letters (a-zA-Z), numbers(0-9), special
characters(_, $, &, +, :, ;, =, ?, @, #, ', <>, [], ^, *, (), %, !, {}, -),
commas(,), spaces, periods(.).
DES_959=Username can contain only letters (a-zA-Z), numbers(0-9), special
characters(_, $, &, +, :, ;, =, ?, @, #, ', <>, [], ^, *, (), %, !, {}, -),
commas(,), spaces, periods(.).
DES_960=Label name can contain only letters (a-zA-Z), numbers(0-9), special
characters(_, $, &, +, :, ;, =, ?, @, #, ', <>, [], ^, *, (), %, !, {}, -),
commas(,), spaces, periods(.).

DES_970=SMTP Timeout should be in between 5 and 10 seconds.

DES_971=No device labels found to delete.
DES_972=All device labels have been deleted successfully.
DES_973=Cannot export analytics report.

DES_1001=JSON format is invalid.
DES_1002=Device details are invalid.
DES_1004=Cannot claim the device as MAC address or serial number is invalid.
DES_1006=Manufacturing date and time is invalid.
DES_1007=Cannot update device as device details are invalid.
DES_1008=MAC address format is invalid.
DES_1009=MAC address does not belong to Avaya.
DES_1010=bulk operation contains no devices.
```

```
DES_1110=Bulk operation does not contain any sites.
DES_1011=Device has been activated successfully.
DES_1012=Device could not be activated.
DES_1013=Device has been deactivated successfully.
DES_1014=Device could not be deactivated.
DES_1015=Device has been released successfully.
DES_1016=Device could not be released.
DES_1017=Device has been claimed successfully.
DES_1018=Device could not be claimed.
DES_1019=Job has been approved successfully.
DES_1020=Job could not be approved.
DES_1021=Devices has already been activated.
DES_1022=Devices has already been deactivated.
DES_1023=Devices has already been released.
DES_1024=Selected job {0} has already been approved.
DES_1025=Cannot claim device as device does not exist with provided details.
Please contact Avaya Administrator.
DES_1026=Permission denied while performing operation.
DES_1027=Device has already been claimed.
DES_1028=Device cannot be deleted.
DES_1029=Cannot update device as it is associated with Service Provider or
Reseller.
DES_1030=Error occurred while performing bulk operation.
DES_1031=Error occurred while performing select all operation.
DES_1032=Cannot claim device as it is associated with another account.
DES_1033=Device already exists with provided details. Please provide latest
manufacturing date and time if the device needs to be updated.
DES_1034=Invalid JSON file content received {0}
DES_1035=Invalid CSV file content received {0}
DES_1036=Device is already associated with account {0}.
DES_1037=Device could not be associated.
DES_1038=Device is already associated with account {0} and customer site
location {1}.
DES_1039=No account {0} exists to associate a device.
DES_1040=No customer site location {0} exists for account {1} to associate a
device.
DES_1041=No account {0} and customer site location {1} exist to associate a
device.
DES_1042=No device with provided details exists.
DES_1043=Device has been associated successfully.
DES_1044=Cannot associate device as it is not owned by logged in account.
DES_1045=Customer site location is required to associate device to customer
account {0}.
DES_1046=Cannot approve job as it is currently running.
DES_1047=Cannot approve job as number of successful devices is 0.
DES_1048=Cannot delete device as it was imported by another account.
DES_1049=Value for {0} cannot be blank at line number {1}.
DES_1050=All the values for properties {0} are required for {1} devices at
line number {2}.
DES_1051=Device is a duplicate.
DES_1052=Please associate customer site location {0} with profile first and
then associate devices with it.
DES_1053=Device could not be updated.
DES_1054=File is required to perform bulk operation on devices.
DES_1055=Error: Invalid input {0} at {1}.
```

```
DES_1056=Either MAC or serial is required to {0} a device.
DES_1057=MAC address or serial number is required to claim a device.
DES_1058=Bulk operation {0} is not allowed using device labels.
DES_1059=Invalid job types, Please provide correct sequence of job types.
DES_1060=Either Serial or MAC is required to perform bulk operation.
DES_1061=Cannot activate device as it is not owned by logged in account.
DES_1062=Cannot deactivate device as it is not owned by logged in account.
DES_1063=Serial or Thumbprint already exists.
DES_1064=Cannot download bulk import job file.
DES_1065=Job status should be complete to download file.
DES_1066=MAC address or serial number is required to associate a device.
DES_1067=Device already associated with another account.
DES_1068=Cannot get device related details.
DES_1069=Cannot get device as it is not owned by logged in account.
DES_1070=Please provide appropriate MAC or Serial details.
DES_1071=Device has been disassociated successfully.
DES_1072=Invalid details. Please provide correct device details.
DES_1073=Device has already been disassociated.
DES_1074=Please disassociate the device first, if it is associated with any
account or site and then try to release.
DES_1075=Parsing issue for Access Token.
DES_1076=Error while creating/updating access token entries.
DES_1077=Cannot disassociate device, please check if device is owned by logged
in account or device is already disassociated.
DES_1078=Cannot disassociate label from the device, please check if device is
owned by logged in account.
DES_1079=Cannot associate label to the device, please check if device is owned
by logged in account.
DES_1080=New Access Token cannot be retrieved, as user {0} already has an
active Access Token. New Access Token can be obtained after {1} (seconds) time
after the current Access Token expires.
DES_1081=Error occurred while performing site bulk operation.
DES_1082=Basic provisioning file doesn't exist in the api payload.
DES_1083=Illegal arguments as part of Upload Form.
DES_1084=Invalid job type.
DES_1085=The Site Filter Property object should contain a valid Des Unique ID.
DES_1086=Device could not be synced.
DES_1087=Avaya Admin cannot delete masqueraded user {0}
DES_1088=Empty device list received to query the DES Inventory.
DES_1089=More than 100 devices are not allowed in the input list.
DES_1090=Device with given mac address does not exist.
DES_1091=Device is not associated with site.
DES_1092=Device Settings File has been Uploaded successfully.
DES_1093=User can only upload mac.txt for devices present in his own
hierarchy.
DES_1097=Device settings file has been deleted successfully.
DES_1098=Cannot delete device settings file.
DES_1099=Cannot delete device settings file as device is not associated with
any site.

DES_2010=reCAPTCHA verification failed for the request. Please refresh the
page and retry.

DES_3000=Access to DES APIs is denied.
DES_3001=Method not allowed.
```

```
DES_3002=Invalid Request.
DES_3003=PID {0} already exists.
DES_3004=Provision Id should be of 4 digits length.

DES_4001=Could not add or update DES user last time access details.
DES_4002=DES user last time access details added successfully.

DES_0021=Firmware URL is required.
DES_0022=Firmware URL is invalid.
DES_0023=Phone group is required.
DES_0024=Phone group is invalid.
DES_0025=Sign-up request already approved or declined.
DES_0026=Could not update the request.
DES_0027=Request approved.
DES_0028=Request declined.
```