NLAD API Version 1.0 Specifications

September 2018

Document Version 5.0.0

Change Log

API Changes	Version
Added a new optional "nv" field to POST /subscriber, POST /transfer, POST /verify, PUT /subscriber	4.0.0
Added NV Enroll Subscriber.	4.0.0
Added NV Update Subscriber.	4.0.0
Added NV API Benefit Transfer.	4.0.0
Added NV Verify Information.	4.0.0
Added NV ETC Reporting API.	4.0.0
Added deceased subscriber resolution parameter to POST /resolution/submit API	3.17.0
Added ETC Recertification Snapshot report to Reporting API.	3.2.0
Added new conditional field to PUT /subscriber and DELETE /subscriber API	3.0.0
Added new optional field to POST /subscriber and POST /transfer	3.0.0
Added new required field and validation to POST /subscriber API	3.0.0
Added new required field and validation to POST /transfer API	3.0.0
Added new required field and validation to POST /verify API	3.0.0
Added new required field and validation to PUT/subscriber API	3.0.0
Added new resolution code and validations to POST /resolution/submit API	3.0.0
“lifelineTribalBenefitFlag” is now a required field: POST /subscriber, POST /transfer, POST /verify, PUT /subscriber	3.0.0
Retired E5, E6, E7, E12 Eligibility codes.	3.0.0
Added 1 new Eligibility code.	3.0.0
Added specifications for GET /report/subscriber/snapshot	2.8.3
Removed specifications for POST /resolution	2.8.2
Added new POST service for Submit Resolution Request with additional fields and validations.	2.6.0
Added urbanization code as optional parameters for subscriber transactions and lookup functionalities (Enroll Subscriber, Update Subscriber, Benefit Transfer, ETC Check Subscriber POST, ETC Check Subscriber GET, Verify Information, Confirm Link Up). The urbanization codes are optional parameters and not required to be included in the request.	2.5.5

Added new POST service for ETC Check Subscriber.		2.5.2
Replaced GET /linkup service with POST /linkup service to Confirm Link Up.		2.5.2
Added duplicate type parameter for Duplicate Resolution De-Enroll Report.		2.3.11
Added Confirm Link Up.		2.3.4
Added Duplicate Resolution De-Enroll report to Reporting API.		1.9
Added De-Enroll Duplicate Subscriber.		1.8
Added Update Duplicate Subscriber.		1.8
Added Duplicate Subscriber report to Reporting API.		1.7
Failure type "Track 1" changed to "Duplicate subscriber".		1.6
Failure type "Track 2" changed to "Duplicate address".		1.6
Failure type "Track 1/Track 2" changed to "Duplicate subscriber/duplicate address".		1.6
Failure type "Track 1/Duplicate Phone" changed to "Duplicate subscriber/duplicate phone number".		1.6
Failure type "Track 2/Duplicate Phone" changed to "Duplicate address/duplicate phone number".		1.6
Failure type "Track 1/Track 2/Duplicate Phone" changed to "Duplicate subscriber/duplicate address/duplicate phone number".		1.6
Language for most messages has been revised (message codes remain unchanged). For more information, see Appendix B.		1.6
Document Changes	**Scope**	**Version**
Added new message codes to the message glossary: NOT_AUTHORIZED_FOR_LEGACY	Appendix B.	5.0.0
Updated dispute resolution codes from Field Description table: Added "A13" for "aCode"	Appendix A.	4.6.0
Added new message codes to the message glossary: INVALID_DATE_RANGE	Appendix B.	4.6.0
Updated message descriptions for APPLICATION_NOT_COMPLETE and APPLICATION_NOT_FOUND	Appendix B.	4.1.0
Added summaries for new API specifications for National Verifier states.	Section 6.1, page 13	4.0.0
Added specifications for POST /subscriber for National Verifier.	Section 6.4, page 23	4.0.0
Added specifications for PUT /subscriber for National Verifier.	Section 6.8, page 30	4.0.0
Added specifications for POST /transfer for National Verifier.	Section 6.11, page 37	4.0.0

	Added specifications for POST /verify for National Verifier.	Section 6.16, page 43	4.0.0
	Added 2 new National Verifier Reports: Reverification Subscriber Status Report Failed Reverification De-Enroll Report	Section 6.18, page 55-56	4.0.0
	Added new field description for "nv".	Appendix A.	4.0.0
	Added new message codes to the message glossary: APPLICATION_NOT_COMPLETE APPLICATION_NOT_FOUND APPLICATION_PENDING CANNOT_UPDATE_ADDRESS_FIELDS CANNOT_UPDATE_ADDRESS_FLAGS DUPLICATE_ADDRESS_NLAD DUPLICATE_SUBSCRIBER_NLAD NV_UNAVAILABLE SAC_NOT_AUTHORIZED_FOR_NV STATE_FED_FAIL	Appendix B.	4.0.0
	Updated dispute resolution codes from Field Description table: Removed "M2" for "mCode" Added "T18" and removed "T24" from "fullNameDeceasedTCode"	Appendix A.	3.18.0
	Updated POST /resolution/submit in the Summary of APIs	Section 6.1, page 13	3.17.0
	Added specification to POST /resolution/submit	Section 6.12, page 39	3.17.0
	Added new field description for "fullNameDeceasedTCode"	Appendix A.	3.17.0
	Added new message code for TPIV failure detail to the message glossary: TPIV_FAIL_DECEASED	Appendix B.	3.17.0
	Removed message code from the message glossary: INVALID_BROADBAND_SVCINITDATE	Appendix B.	3.17.0
	Removed record validations for Port Freeze	Section 4.2, page 6-7	3.14.0
	Removed Port Freeze resolution codes from Arguments/Returns section in Summary of APIs for POST /resolution/submit	Section 6.1, page 14	3.14.0

	Removed parameter “portFreezeExceptionCode” from Submit Resolution Request	Section 6.12, page 39	3.14.0
	Removed field name from Field Description table: portFreezeExceptionCode	Appendix A.	3.14.0
	Updated message description for CANNOT_UPDATE_SVCINITDATE	Appendix B.	3.14.0
	Added new message code to the message glossary: INVALID_RESOLUTION_REQUEST	Appendix B.	3.14.0
	Removed message codes from the message glossary: CANNOT_ENROLL_WITHIN_60_DAYS CANNOT_TRANSFER_WITHIN_60_DAYS CANNOT_ENROLL_WITHIN_12_MONTHS CANNOT_TRANSFER_WITHIN_12_MONTH PRIMARY_ADDRESS_MATCH	Appendix B.	3.14.0
	Added record validation for matching subscriber and BQP information.	Section 4.2, page 7	3.12.0
	Added new message code to the message glossary: SUBSCRIBER_BQP_MATCH	Appendix B.	3.12.0
	Added new message codes to the message glossary: SSN_AND_TRIBAL SSN_AND_TRIBAL_BQP	Appendix B.	3.10.0
	Added record validation for disabling updates to “bqpLastName”, “bqpFirstName”, “bqpDOB”, “bqpLast4ssn” and “bqpTribalId” fields.	Section 4.2, page 7	3.5.8
	Updated record validation for disabling updates to include “lastName”, “firstName” and “tribalId” fields.	Section 4.2, page 7	3.5.8
	Added new message code to the message glossary: CANNOT_UPDATE_BQP_IDENTITY_FIELDS	Appendix B.	3.5.2
	Updated message description to the message glossary: INCOMPLETE_BQP_INFORMATION	Appendix B.	3.5.2
	Updated conditional requirements for BQP fields.	Appendix A.	3.5.2
	Added verify to transactionType.	Appendix A.	3.5.2
	Added new optional query string (?rowsProcessed) for GET /batch.	Section 6.2.2, page 18	3.5.0
	Added error message for invalid tribal benefit: INVALID_LIFELINE_TRIBAL_BENEFIT_FLAG	Appendix B.	3.4.11

INVALID_PRIMARY_TRIBAL_FLAG		
Added error message for Invalid City, State or Zip: INVALID_CITY_STATE_ZIP	Appendix B.	3.4.7
Added error message for Multiple SAC: MAX_SACS_LIMIT_REACHED	Appendix B.	3.4.5
Added new message codes for external duplicate check: DUPLICATE_SUBSCRIBER_EXTERNAL STATE_WEBSERVICE_ERROR	Appendix B.	3.4.0
Added new message codes for Invalid Zip Codes and General Delivery on address lines 1 & 2: ZIP_CODE_NOT_MATCHED INVALID_ADDRESS_LINE	Appendix B.	3.3.2
Updated message code for "MULTIPLE_SUBSCRIBERS_FOUND" to include "please use the subscriber ID."	Appendix B.	3.3.1
Added new message codes for Enroll, Update and Transfer to the message glossary: UNAVAILABLE_ELIGIBILITY_CODE	Appendix B.	3.3.1
Added specifications for ETC Recertification Snapshot report to Reporting API Section	Section 6.13, page 43-44.	3.2.0
Added new message codes for svcinitdate restrictions to the message glossary: INVALID_SVCINITDATE INVALID_BROADBAND_SVCINITDATE	Appendix B.	3.0.1
Added new message codes for Update and De-Enroll to the message glossary: INVALID_SUBSCRIBER_ID MISSING_PHONE_NUMBER_OR_SUBSCRIBER_ID CANNOT_ENTER_PHONE_NUMBER_AND_SUBSCRIBER_ID	Appendix B.	3.0.0
Added new field name to the Field Descriptions Table: subscriberId includeSubscriberId	Appendix A.	3.0.0
Added new optional parameters (subscriberId) for DELETE /subscriber and PUT /subscriber.	Section 6.4, page 22 Section 6.6, page 24	3.0.0
Added new optional parameter (includeSubscriberId) for POST /subscriber and POST /transfer to return a subscriber's Subscriber ID on successful transactions.	Section 6.3, page 20 Section 6.8, page 29	3.0.0

	Added new optional parameter (includeSubscriberId) for ETC Subscriber Report, ETC Transaction Report, and ETC Summary & Detail Subscriber Snapshot Report.	Section 6.13.1, page 38 Section 6.13.4, page 40 Section 6.13.6, page 42	3.0.0
	Updated record validation for eligibility codes for enroll, update and transfer.	Section 4.2, page 5.	3.0.0
	Updated and added record validation for 60 day and 12 month port freeze period on Verify, Enroll & Benefit Transfer.	Section 4.2, page 6.	3.0.0
	Added record validation for service-initialization date rule to service type on Update.	Section 4.2, page 7.	3.0.0
	Added record validation for limitation on use of R Code on Submit Resolution Request.	Section 4.2, page 7.	3.0.0
	Added specifications for POST /subscriber	Section 6.3, page 19-20.	3.0.0
	Added specifications for PUT/subscriber	Section 6.6, page 25.	3.0.0
	Added specifications for PUT/duplicate	Section 6.7, page 27.	3.0.0
	Added specifications for POST /transfer	Section 6.8, page 29.	3.0.0
	Added specifications for POST /resolution/submit	Section 6.9, page 31.	3.0.0
	Added specifications for POST /verify	Section 6.12, page 35-36.	3.0.0
	Added new field names to the Field Descriptions Table: serviceType, portFreezeExceptionCode	Appendix A.	3.0.0
	Updated "lifelineTribalBenefitFlag" to a required field.	Appendix A.	3.0.0
	Identified retired Eligibility codes.	Appendix A.	3.0.0
	Added new Eligibility code.	Appendix A.	3.0.0
	Added new message codes to the message glossary: CANNOT_ENROLL_WITHIN_60_DAYS CANNOT_ENROLL_WITHIN_12_MONTHS CANNOT_TRANSFER_WITHIN_12_MONTHS CANNOT_UPDATE_SVCINITDAT INVALID_SERVICE_TYPE PRIMARY_ADDRESS_MATCH SVCINITDATE_DOES_NOT_MATCH	Appendix B.	3.0.0
	Added record validation for disabling updates when no fields are changed.	Section 4.2, page 7.	2.7.4

	Added new message code for Update Subscriber to the message glossary: NO_FIELDS_ARE_UPDATED	Appendix B.	2.7.4
	Updated description of INVALID_PHONE_NUMBER message code.	Appendix B.	2.7.3
	Added new message codes for Submit Resolution Request to the message glossary: RESOLUTION_SUBSCRIBER_ALREADY_SUBMITTED	Appendix B.	2.6.5
	Removed acpFlag from the description for the error code MULTIPLE_ACP_RURAL_TRIBAL.	Appendix B.	2.6.3
	Updated acpFlag data description and required status.	Appendix A	2.6.3
	Removed acpFlag record validation rules regarding use of multiple address flags.	Section 4.2, page 6.	2.6.3
	Updated record validation rules for PO Box primary addresses.	Section 4.2, page 6.	2.6.3
	Added POST /resolution/submit to the Summary of APIs.	Section 6.1, page 14.	2.6.0
	Updated POST /resolution API specifications to deprecated.	Section 6.9, page 31.	2.6.0
	Added specifications for POST /resolution/submit	Section 6.10, page32.	2.6.0
	Updated GET Confirm Linkup API specifications to deprecated.	Section 6.12, page 35.	2.6.0
	Added new field names to the Field Descriptions Table: resolutionId, agentName, agentId, fullNameLast4ssnTCode, fullNameDobTCode, aCode, mCode, desc, certificationFlag	Appendix A.	2.6.0
	Added new message codes for Submit Resolution Request to the message glossary: INVALID_AGENT_NAME INVALID_CERTIFICATION_FLAG INVALID_RESOLUTION_CODE	Appendix B.	2.6.0
	Added optional address parameters (mailingUrbanizationCode and primaryUrbanizationCode) for Enroll Subscriber, Update Subscriber, Benefit Transfer, ETC Check Subscriber POST, ETC Check Subscriber GET, Verify Information, Confirm Link Up, and Appendix A to support Puerto Rican addresses Both urbanization code fields are optional parameters and not required to be included in any requests.	Section 6.3, Section 6.6, Section 6.8, Section 6.10, Section 6.11, Section 6.12, Section 6.14, Appendix.	2.5.5
	Added specifications for ETC Check Subscriber POST.	Section 6.10, page 33.	2.5.2
	Updated specifications for Confirm Link Up.	Section 6.14. page 42.	2.5.2
	Added POST ETC Check Subscriber to the Summary of APIs.	Section 6.1, page 14.	2.5.2

	Replaced GET Confirm Link Up API to a POST in the Summary of APIs.	Section 6.1, page 14.	2.5.2
	Added new message code for disabling updates to “last4ssn” and “dob” fields: CANNOT_UPDATE_IDENTITY_FIELDS	Appendix B.	2.4.6
	Added record validation for disabling updates to “last4ssn” and “dob” fields.	Section 4.2, page 8.	2.4.6
	Updates to Benefit Transfer function due to reported errors.	Section 6.8, page 29.	2.4.4
	Removed the following message code: “CANNOT_CHANGE_IDENTITY_FIELDS_TRANSFER.” This return message will no longer be used.	Appendix B.	2.4.4
	Added new message codes for TPIV failure details to the message glossary: TPIV_FAIL_NAME_SSN4 TPIV_FAIL_DOB TPIV_FAIL_IDENTITY_NOT_FOUND	Appendix B.	2.3.12
	Updated TPIV_FLAG field data description.	Appendix A.	2.3.12
	Removed the following message code for “TPIV_FLAG” validation: INVALID_TPIVFLAG_CODE	Appendix B.	2.3.12
	Removed the following record validation for TPIV_FLAG field: “If the "tpivFlag" field has a value, then it can only contain the value "0" or one of the values listed in NLAD Dispute Resolution.”	Section 4.2, page 7.	2.3.12
	Added new message code for Duplicate Resolution De-Enroll report to the message glossary: INVALID_DUPLICATE_TYPE	Appendix B.	2.3.11
	Added duplicate type parameter for Duplicate Resolution De-Enroll Report.	Section 6.13, page 39.	2.3.11
	Added specifications for Confirm Link Up.	Section 6.14, page 42.	2.3.4
	Added new Confirm Link Up API to the Summary of APIs.	Section 6.1, page 16.	2.3.4
	Added new message codes for Confirm Link Up to message glossary: LINKUP_DATE_FOUND NO_LINKUP_DATE	Appendix B.	2.3.4
	Updated message code for “INVALID_TPIVFLAG_CODE“ to include “T17” code.	Appendix B.	2.3.4
	Updated message code for “CANNOT_TRANSFER_WITHIN_60_DAYS” to include date.	Appendix B.	2.2
	Added tpivFlag Resolution Error Code “T17”	Appendix A.	2.1
	Removed from the message glossary: DUPLICATE_SUBSCRIBER_AND_ADDRESS	Appendix B.	2.1

	Updated Section 5 to reflect it is suggested to interpret the response from NLAD by Message Code and not by Failure Type.	Section 5, page 9.	2.0
	Added new validation rules: TPIV Flag validation Last 4 Digit of SSN Validation	Section 4.2, page 7.	2.0
	Added new error message for TPIV_FLAG to the message glossary: INVALID_TPIVFLAG_CODE	Appendix B.	2.0
	Added specifications for Duplicate Resolution De-Enroll report to Reporting API section.	Section 6.13, page 38.	1.9
	Added specifications for De-Enroll Duplicate Subscriber.	Section 6.5, page 22.	1.8
	Added specifications for Update Duplicate Subscriber.	Section 6.7, page 26.	1.8
	Added new error messages for De-Enroll Duplicate Subscriber and Update Duplicate Subscriber to the message glossary: NOT_WITHIN_DATE_RANGE MULTIPLE_SUBSCRIBERS	Appendix B.	1.8
	Added specifications for Duplicate Subscriber report to Reporting API section.	Section 6.13, page 32.	1.7
	Added missing batch status codes and batch failure codes (only used in batch status objects).	Section 6.2.2, page 16.	1.6
	Removed INVALID_SAC_PHASE from the message glossary (not in use in API).	Appendix B.	1.6
	Fixed erroneous HTTP status codes in the message glossary for: TOO_MANY_HEADERS TOO_FEW_HEADERS FILE_NOT_FOUND INVALID_IEH_RECERTIFICATION_DATE ERRORS IN_PROGRESS	Appendix B.	1.6
	Added error response for multiple-file submissions to the batch API.	Section 6.2, page 15.	1.6
	Expanded batch API section to include file format and submission restrictions, batch status codes, and batch failure codes.	Section 6.2 and 6.2.2.	1.6
	Added two failure types, “DRC – In Progress” and “Internal Error” to Failure Types table.	Section 5, page 10.	1.6
	Revised description of JSON header and body.	Section 5, page 9.	1.6
	Fixed syntax error in JSON response explication.	Section 5, page 9.	1.6

	Congruity changes to Validation Error and Malformed Document to accord to descriptions and other examples.	Section 5, page 10.	1.6
	Changed field description for primaryPermanentAddressFlag from “Permanent Address Flag” to “Temporary Address Flag” to match the UI and function of the field, and added a note to the data description to draw attention to the incongruity between field name and field behavior.	Appendix B, page B-2.	1.5
	Removed erroneous reference to batch-specific API response in Failure Types table.	Section 5, page 10.	1.5
	Changed name of base API URL from “Hypothetical URL” to just “URL” in the URL Deconstruction section.	Section 2.1.2, page 2.	1.5
	ETC Reporting API example responses now properly show the message code in the body of the response.	Section 6.11.	1.4
	Changed all Track 1/Track 2 references to Duplicate Subscriber/Duplicate Address.	Throughout the document.	1.4
	Verify Information API - the serviceInitializationDate (Service Initiation Date) field is now properly noted as being optional.	Section 6.10, page 30.	1.4
	Added explication of API response structure to section 5.	Section 5, page 8.	1.4
	Data Types and Descriptions moved to Appendix A.	Section 3, page 4.	1.4
	List of API-returned message codes created in Appendix B.	Appendix B, page B-1.	1.4
	Updated all example return messages to match changes in API response structure.	Throughout the document.	1.4
	Removed notice that API responses are subject to change.	Throughout the document.	1.4
	Removed erroneous references to "resolution category" as one of the required parameters for the Submit Resolution Request API.	Section 6.1, page 13, and Section 6.11, page 32.	1.4

Contents

1. Introduction

The National Lifeline Accountability Database (NLAD) is designed to help carriers identify and resolve duplicate claims for Lifeline Program-supported service and prevent future duplicates. This is done by providing a means for carriers to check on a real-time and nationwide basis if the consumer is already receiving a Lifeline Program-supported service.

Eligible telecommunications carriers (ETCs) are to provide information about subscribers and obtain status, feedback, and reporting information either through the NLAD Access Portal (NAP) or an Application Program Interface (API).

This document covers specifications related to usage of an API with NLAD, including how to connect to the services, format transactions, and manage and interpret returns and responses.

2. Connecting

This section includes the uniform resource locator (URL) and authentication information necessary in order to establish connection to NLAD.

2.1. Service URLs

Complete URLs for NLAD will be provided by USAC.

2.1.1. URL Construction

The Base URL Construction for the NLAD system is as follows:

```
protocol://hostname/svc/api-version/resource.
```

2.1.2. URL Deconstruction

```
https://nlad.universalservice.org/svc/1/subscriber
```

Code Block 1 Deconstruction

```
"https://" = protocol
"nlad.universalservice.org" = hostname
"svc" = service
"1" = api version
"subscriber" = resource
```

2.2. Authentication and Authorization

NLAD utilizes the HTTP 1.1 Basic Authentication Scheme as described in IETF RFC 2617, Section 2.

- If the user agent wishes to send the userid "Aladdin" and password "open sesame," it would use the following header field: **Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==**, where the last string in the line is the Base64 encoded concatenation of the username, a colon (:) and the password.
- The user agent requesting access must be capable of accepting cookies and following all HTTP redirects. Only requests, submitted using the HTTPS (SSL/TLS), will be accepted.
- The NLAD authorization mechanism restricts each authenticated API user to data related to the SACs assigned to them as well as restricting them to specific API operations and resources when their API account is provisioned by USAC.
- **HINT:** To prevent a 401 Unauthorized message on the first connection attempt in a session, include the cookie OBBasicAuth=fromDialog in the request headers. All subsequent requests should include the cookie that is set by the service, even if the first request generates a 401 Unauthorized message.

3. Data Format

For batch transactions, the file format must be CSV, and the CSV header row must contain all 41 fields as they appear in the Field Descriptions table in **Appendix A - Data Types and Descriptions**.

4. Validations

The following tables list file and record validations.

File validations determine the acceptance of a batch file (more information about batch processing can be found in the **Batch Subscriber Upload** section of the API Specifications).

Record validations determine the acceptance of individual transactions, either the body of a JSON document, or a single transaction row in a CSV file (more information about field criteria can be found in **Appendix A** - **Data Types and Descriptions**).

4.1. File Validations

The following table is a list of all validations to which the batch file must accord. If any one of these rules is false, then the entire file will be rejected.

File Validation Rules
File format must be CSV.
File name cannot contain a space.
File name cannot contain any of the following characters: \ / : * ? " < > |
First six digits of the file name must be all numeric characters, and correspond to a valid SAC number associated with the submitting ETC.
File must contain a header row.
Header row must contain all the fields as in the header of the NLAD Input Template.
Header row must contain the exact field names as specified in header of the NLAD Input Template.
An ETC cannot upload a file for a SAC which does not belong to the ETC.

4.2. Record Validations

The following table is a list of validations to which each subscriber transaction must accord. If any one of these rules is false, then the transaction is rejected. For batch uploads, only the individual row is rejected, not the entire file.

The transaction type column lists the transactions the rule applies to. The full list of possible transactions is:

- `enroll` – Used to enroll a Lifeline eligible subscriber.
- `transfer` – Used to transfer the Lifeline benefits from another carrier.
- `update` – Used to update/change a subscriber's information.
- `deEnrollDeceased` – Used to de-enroll a Lifeline subscriber who has deceased.
- `deEnrollLeaving` – Used to de-enroll a Lifeline subscriber who is opting out of the program, or is no longer eligible for benefits.

- `deEnrollFailedRecertification` – Used to de-enroll a Lifeline subscriber who has not filed their annual recertification.
- `deEnrollNonUsage` – Used to de-enroll a Lifeline subscriber who has not used their benefits for 60 days.

Record Validation Rules	Transaction Type
Every row of the file must have the correct number of fields.	Any
All required fields must contain data. Different transaction types require different sets of fields. See **NLAD Field Descriptions.xlsx** for more information about conditionally required fields.	Any
Fields must contain the correct data types.	Any
Field values cannot exceed the specified maximum length for that field.	Any
Date fields cannot contain dates in the future.	Any
All subscribers must be between 18 and 130 years of age.	Any
A Tribal ID is required for the subscriber if the subscriber's social security number is not provided.	Any
The field "eligibilityCode" can only contain one of the values listed in **NLAD Enrollment Eligibility Codes**. As of 12/1/16, the following eligibility codes can no longer be used, except when using the Update Subscriber service to update fields other than eligibilityCode: E5 E6 E7 E12.	Enroll, Update, Transfer
If any mailing-address field is provided (mailingAddress1, mailingAddress2, mailingCity, mailingState, mailingZipCode), then all mailing address fields, except "mailingAddress2", must be provided.	Any
If field "linkUpServiceDate" has a value, then "lifelineTribalBenefitFlag" must be set to "1". Note: "lifelineTribalBenefitFlag" does not require "linkUpServiceDate".	Enroll, Update, Transfer
If the field "linkUpServiceDate" has a value, then the SAC number must be qualified to provide Link Up service.	Enroll, Update, Transfer
If the Independent Economic Household field (iehFlag) = "1", then the field "iehCertificationDate" is required.	Enroll, Update, Transfer
If the IEH certification date field (iehCertificationDate) has a value, then the field "iehFlag" must be set to "1".	Enroll, Update, Transfer
If any BQP field is provided, then "bqpLastName" is required.	Enroll, Update, Transfer
During initialization, the only valid transaction type is Enroll.	Enroll
During initialization, within a batch file, all subscribers must have the same SAC number as the SAC number in the file name.	Enroll
Within a batch file, all subscribers must have unique telephone numbers.	Enroll

During production, if the phone number already exists in NLAD, the transaction will be rejected. The system will not check the subscriber record against itself during an update or transfer transaction.	Enroll, Update, Transfer
Within a batch file, no two subscribers can have the same combination of last name, date of birth, and (SSN or Tribal ID).	Enroll
During production, if the transaction contains a subscriber that matches an existing subscriber in NLAD, then the transaction will be rejected. The system will not check the subscriber record against itself during an update or transfer transaction.	Enroll, Update, Transfer
During production, If "iehFlag" is set to "0" or "null", then the transaction will be rejected if the subscriber's address matches another subscriber's address already in NLAD.	Enroll, Update, Transfer
If the primary address fails the USPS Address Matching Service, and the"primaryTribalFlag" or "primaryRuralFlag" is not "1", then the subscriber will be rejected.	Enroll, Update, Transfer
If the mailing address fails the USPS Address Matching Service, the transaction will be rejected.	Enroll, Update, Transfer
During production, if the subscriber's identity fails the Third-Party Identification Verification (TPIV), the transaction will be rejected.	Enroll, Transfer
A subscriber can only qualify for one of these address flags: "primaryRuralFlag" or "primaryTribalFlag". If more than one of these fields is set to "1", the transaction will be rejected.	Enroll, Update, Transfer
During production, any transaction with a SAC number, for which the ETC is not authorized, will be rejected.	Enroll, Update, Transfer
If any address field is changed during an Update or Transfer, USPS Address Matching Service will attempt to validate the new address before the Update or Transfer is completed.	Update, Transfer
An ETC can only update its own subscriber.	Update
An ETC cannot transfer a subscriber to an ETC other than itself.	Transfer
An ETC can only De-Enroll its own subscriber.	De-enroll
For Update and De-Enroll transactions, "phoneNumberInNlad" is required.	Update, De-enroll
For Transfer transactions, the subscriber's phone number (phoneNumberInNlad) or identity (lastName, dob, and [last4ssn or tribalId]) will be used to locate the subscriber in NLAD.	Transfer
During a Benefits Transfer, the subscriber's Last Name, Date of Birth, SSN, and Tribal ID cannot differ from the data in NLAD.	Transfer
Transactions with PO Box numbers in primary address fields will be rejected.	Enroll, Update, Transfer
An ETC cannot transfer one of its own subscribers within the same SAC. However, it can transfer a subscriber from one of its SACs to a different one of its SACs.	Transfer
An ETC cannot change the SAC number associated with a subscriber using an Update transaction.	Update

An ETC cannot change the Last Name, First Name, Date of Birth, Last 4 SSN or Tribal ID of a subscriber using an Update transaction.	Update
An ETC cannot change BQP Last Name, BQP First Name, BQP Date of Birth, BQP Last 4 SSN or BQP Tribal ID of a subscriber using an update transaction where BQP information was already submitted.	Update
An Update or a Transfer cannot be performed on a subscriber who is currently de-enrolled.	Update, Transfer
The last 4 digits of a subscriber's social security number or BQP social security number cannot be "0000"	Any
A subscriber field must be modified to complete an update transaction.	Update
A subscriber cannot update the service-initialization date if the service type is not being updated.	Update
If the primary address of re-enrolling/ transferring subscriber to a different SAC matches previously enrolled subscriber, then the ETC cannot submit an R-code for resolution exception process.	Verify, Enroll, Transfer
The field "serviceType" can only contain one of the of the following values: voice, broadband, bundledVoice, bundledBroadband, bundledVoiceBroadband.	Verify, Enroll, Update, Transfer
Only National Verifier SACs qualifiy to use the "nv" field. If non-National Verifier SACs sets this field to "1" the transaction will be rejected.	Verify, Enroll, Update, Transfer
Subscriber information cannot match BQP information if both are provided.	Verify, Enroll, Update, Transfer
An ETC cannot verify, enroll, update or transfer a subscriber using legacy mode if they are in a National Verifier state and the Soft Launch period has ended.	Verify, Enroll, Update, Transfer

5. Success and Error Responses

Responses from the NLAD API are JSON formatted.

Success messages are included in the body of the JSON document. No message code is returned with success messages.

JSON Start		**[**
Body	**Body Start**	**{**
	Success Message	“message”: "<SUCCESS MESSAGE>"
	Body End	**}**
JSON End		**]**

Example Success Response

```
HTTP/1.1 200 OK
 [{"message": "Subscriber de-enrolled."}]
```

Error messages contain a header, and provide more detailed responses in the body.

JSON Start		**{**
Header	**Header Start**	**"header": {**
	Resolution ID	"resolutionId": "<RESOLUTION ID>",
	Failure Type	"failureType": "<FAILURE TYPE>"
	Header End	**},**
Body	**Body Start**	**"body": [**
	Error 01	["<ERROR TYPE>","<MESSAGE>","<MESSAGE CODE>","<OFFENDING DATA>"]
	Error 02	["<ERROR TYPE>","<MESSAGE>","<MESSAGE CODE>"]
	Body End	**]**
JSON End		**}**

The header will conditionally contain a resolution ID (when applicable), and will always contain the failure type of the transaction. If the transaction does not contain the appropriate condition(s) to generate a resolution ID, the resolutionId will be empty. If the transaction cannot generate a resolution ID in any scenario, the resolutionId will be absent from the header.

The failure type is a category of errors, within which a number of specific errors can occur. For example, the failure type "Validation Errors" could include the specific error that a field has data in the wrong format. A list of possible failure types can be found at the end of this section, in the **Failure Types** table. It is recommended to not use failure type to interpret the NLAD response.

The body of the response will contain a list of one or more errors associated with the transaction. Each error will include the *error type*, which may be a type of error, or the name of the field containing the offending data. It will be followed by an *error message*, and that will be followed by a *message code*, which is a unique identifier associated with the error message. Lastly, some errors will return the offending data. It is strongly recommended to use the message code when interpreting the NLAD response. A list of API message codes can be found in **Appendix B – API Return Messages**.

Note: the error type for Batch API responses will always be "error."

In the below examples, the first shows a failure type of "Validation Error," and returns the invalid date input at the end of the error message, while second shows a failure type of "Duplicate subscriber," and does not return the offending data.

Example Error Responses

```
{
      "header": {
            "failureType": "Validation Error"
      },
      "body": [
      ["endDate","Date is in the incorrect format. The correct format is:
MM/DD/YYYY.", "INVALID_DATE_FORMAT","4/1/13"]
      ]
}
```

```
{
      "header": {
            "resolutionId": "999999-JKMBC8",
            "failureType": "Duplicate subscriber"
      },
      "body": [["Subscriber",
      "The subscriber in this transaction is a duplicate of another subscriber.",
      "DUPLICATE_SUBSCRIBER"]]
}
```

The following table provides a list of all failure types for API error responses.

Note: the "body" section of the **Example JSON Responses** has been left intentionally blank. Actual JSON error responses will include errors in the "body," as seen in all examples in the **API Specifications** section.

Failure Types

HTTP Response Code	Failure Type	Example JSON Response	Description

400 BAD REQUEST	DRC - In Progress	{
		"header": {
		"resolutionId": "",
		"failureType": "DRC - In Progress"
		},
		"body": []
		}
			A DRC - In Progress error indicates that the subscriber is currently undergoing duplicate resolution. No resolution ID is generated for this error.

400 BAD REQUEST	Duplicate subscriber	{
		"header":{
		"resolutionId":"999999-xyz123",
		"failureType":"Duplicate subscriber"
		},
		"body":[]
		}
			A Duplicate subscriber error indicates that the subscriber already exists in the NLAD production database. In this event, a resolution ID will be generated.

400 BAD REQUEST	Duplicate address	{
		"header":{
		"resolutionId":"",
		"failureType":"Duplicate address"
		},
		"body":[]
		}
			A Duplicate Address error indicates that a subscriber already exists at the address provided in the request.

400 BAD REQUEST	Duplicate phone number	{
		"header":{
		"resolutionId":"999999-J9K9S4",
		"failureType":"Duplicate phone number"
		},
		"body":[]
		}
			A Duplicate Phone Number error indicates that the phone number already exists in the NLAD production database. In this event, a resolution ID will be generated.

400 BAD REQUEST	Duplicate subscriber/duplicate address	{
		"header":{
		"resolutionId":"999999-J9K9S4",
		"failureType":"Duplicate subscriber/duplicate address"
		},
		"body":[]
		}
			A Duplicate subscriber/duplicate address error indicates that the subscriber already exists, and a subscriber exists at the address provided in the request. In this event, a resolution ID will be generated.

HTTP Response Code	Failure Type	Example JSON Response	Description

400 BAD REQUEST	Duplicate subscriber/duplicate phone number	{
		"header": {
		"resolutionId": "999999-BD7R6O",
		"failureType": "Duplicate subscriber/duplicate phone number"
		},
		"body": []
		}
			A Duplicate subscriber/duplicate phone number error indicates that the subscriber already exists, and the phone number was found in NLAD. In this event, a resolution ID will be generated.

400 BAD REQUEST	Duplicate address/duplicate phone number	{
		"header": {
		"resolutionId": "999999-BD7R6O",
		"failureType": "Duplicate Address/duplicate phone number"
		},
		"body": []
		}
			A Duplicate address/duplicate phone number error indicates that a subscriber exists at the address provided in the request, and the phone number was found in NLAD. In this event, a resolution ID will be generated.

400 BAD REQUEST	Duplicate subscriber/duplicate address/duplicate phone number	{
		"header": {
		"resolutionId": "999999-BD7R6O",
		"failureType": "Duplicate subscriber/duplicate address/duplicate phone number"
		},
		"body": []
		}
			A Duplicate subscriber/Duplicate address/duplicate phone number error indicates that the subscriber already exists, a subscriber exists at the address provided in the request, and the phone number was found in NLAD. In this event, a resolution ID will be generated.

400 BAD REQUEST	Malformed Document	{
		"header": {
		"failureType": "Malformed Document"
		},
		"body": []
		}
			A Malformed Document error indicates that there was an issue with the document provided in the body of the POST, PUT, or DELETE request, or in the query string of a GET request. No resolution ID will be generated for a Malformed Document error.

400 BAD REQUEST	Validation Error	{
		"header":{
		"resolutionId":"",
		"failureType":"Validation Error"
		},
			A Validation Error indicates that there was an issue one or multiple fields in the document provided in the body of the POST, PUT, or DELETE request, or in the query string of a GET request. A Validation Error will also return additional information about the fields that failed during validation. This includes Address Matching Service (AMS) errors and Third-party

HTTP Response Code	Failure Type	Example JSON Response	Description
		"body":[] }	Identification Verification (TPIV) errors. A resolution ID will only be generated for AMS or TPIV Validation Errors.

500 INTERNAL SERVER ERROR	Internal Error	{
		"header": {
		"failureType": "Internal Error"
		},
		"body": []
		}
			An Internal Error can occur when the NLAD system encounters an internal problem processing the transaction. If you receive this failure type, contact NLAD Customer Service at (877) 524-1325.

6. API Specifications

This section includes the detailed specifications for all of the NLAD API endpoints, with request and response message definitions.

6.1. Summary of APIs

The following table is a summary list of all NLAD public REST services and their endpoints.

Operation	Arguments/Returns	Description
GET /batch	Arguments: No body or query string. Returns: 200 OK with JSON Document.	Service that provides results of the most recent batch processed by NLAD per SAC.
GET /subscriber (Deprecated)	Arguments: lastName, dob, last4ssn, primaryAddress, primaryCityStateZip, mailingAddress, mailingCityStateZip in querystring. Returns: 200 OK with JSON document or 400 Bad Request with JSON document.	Service to allow ETC to check if a potential subscriber and subscriber addressisalready in the database.
POST /subscriber/lookupSubscriber	Arguments: lastName, dob, last4ssn,tribalid, primaryAddress,primaryCity,primaryState, primaryZip inquerystring.Returns: 200 OK with JSON document or 400 Bad Request with JSON document.	Service to allow ETC to check if a potential subscriber and subscriber address is already in the database.
POST /batch	Arguments: Body contains CSV batch transaction file. Returns: 200 OK or 400 Bad Request with JSON document.	Service to post batch transaction file to NLAD for asynchronous processing.
POST /subscriber	Arguments: Body contains all subscriber fields required for the enroll transaction in JSON format. Returns: 201 CREATED or 400 Bad Request with JSON document.	Service that accepts a single subscriber per transaction. The subscriber information is validated, then the subscriber is either enrolled or the service returns rejection codes and resolution ID when appropriate.
POST /transfer	Arguments: Body contains all subscriber fields required for the enroll transaction and phoneNumberInNlad for subscriber identification in JSON format.	Service that allows ETC to transfer benefits of a subscriber from the current ETC claiming benefits or returns rejection codes.

Operation	Arguments/Returns	Description
	Returns: 200 OK or 400 BAD REQUEST with JSON document.	
POST /resolution/submit	Arguments: Body contains the following fields in JSON format: resolutionId, agentname, agentid, fullNameL4ssnTcode, fullNameDobTcode, fullNameDeceasedTCode, acode, mcode, rcode, scode, lcode, vcode, desc, certificationFlag. Returns: 201 CREATED or 400 Bad Request with JSON document.	Service for ETCs to submit a resolution request to the NLAD Customer Service helpdesk. Resolution ID, agent name and/or ID, resolution code(s) and certification flag are required fields. Description field is optional.
POST /verify	Arguments: Body contains all subscriber fields required for the enroll transaction in a JSON format, with the option to omit phoneNumber. Returns: 200 OK with JSON document or 400 Bad Request with JSON document.	Service to check if potential subscriber will pass all validations and be eligible for enrollment.
GET /report/subscriber	Arguments: reportType, startDate, endDate, sac. Returns: 200 Created with CSV document or 400 Bad Request with JSON document.	Service to allow ETC to request a summary or detailed report of all subscribers currently enrolled with the ETC.
GET /report/subscriber/snapshot	Arguments: reportType, snapPeriod, sac. Returns: 200 Created with CSV document or 400 Bad Request with JSON document.	Service to allow ETC to request a summary or detailed snapshot report ofNLAD'sactive subscribers listing as of the 1st day of each month at 6 am ET. The snapshot data will be available for download after 7 am ET onsameday. The snapshot data will be available for download up to 24 months. Example, selected data month of August (08/2016) reflects subscribers as of September 01.
GET /report/migdup	Argument: sac Returns: 200 Created with CSV document or 400 Bad Request with JSON document.	Service to allow an ETC to request a detailed report of all duplicate subscribers and subscribers identified as migration duplicates.
GET /report/transaction	Arguments: reportType, startDate, endDate, sac.	Service to allow ETC to request a summary or detailed report of all transactions that have occurred during a specified time period.

Operation	Arguments/Returns	Description
	Returns: 200 Created with CSV document or 400 Bad Request with JSON document.	
GET /report/resolution	Arguments: reportType, startDate, endDate, resolutionRequestStatus, sac. Returns: 200 Created with CSV document or 400 Bad Request with JSON document.	Service to allow ETC to request a summary or detailed report of all resolution requests that have occurred during a specified time period.
PUT /subscriber	Arguments: Body contains all subscriber fields required for the enroll transaction and phoneNumberInNlad for subscriber identification in JSON format. Returns: 200 OK or 400 BAD REQUEST with JSON document.	Service that allows ETC to update a current subscriber's information. The subscriber information is validated, then the subscriber is either updated or the service returns rejection codes and potentially a resolution ID.
DELETE /subscriber	Arguments: Body contains the following fields in JSON document: transactionType, phoneNumberInNlad, transactionEffectiveDate. Returns: 200 OK or 400 BAD REQUEST with JSON document.	Service that validates that a subscriber exists, then de-enrolls that subscriber or returns rejection codes.
DELETE /duplicate	Arguments: Body contains the following fields in JSON document: transactionType, phoneNumberInNlad, transactionEffectiveDate. Returns: 200 OK or 400 BAD REQUEST with JSON document.	Service that validates that a subscriber exists in duplicate resolution, then de-enrolls that subscriber or returns rejection codes.
PUT /duplicate	Arguments: Body contains all subscriber fields required for the enroll transaction and phoneNumberInNlad for subscriber identification in JSON format. Returns: 200 OK or 400 BAD REQUEST with JSON document.	Service that allows an ETC to update a subscriber's phone number. The subscriber information is validated, then the subscriber is either updated or the service returns rejection codes.
POST /linkup	Arguments: lastName, dob, last4ssn or tribalId, primaryAddress, primaryCity, primaryState, primaryZipCode in querystring. Returns: 200 OK with JSON document or 400 Bad Request with JSON document.	Service that allows an ETC to confirm Link Up to determine whether a subscriber has already received a Link Up benefit a specific address.

Operation	Arguments/Returns	Description
NV POST /subscriber	Arguments: Body contains all subscriber fields required for the enroll transaction in JSON format. Returns: 201 CREATED or 400 Bad Request with JSON document.	National Verifierstates:Service that accepts a single subscriber per transaction. The subscriber information is validated through the National Verifier, then the subscriber is either enrolled or the service returns rejection codes which may require document uploads through the National Verifier web portal.
NV POST /transfer	Arguments: Body contains all subscriber fields required for the enroll transaction and phoneNumberInNlad for subscriber identification in JSON format. Returns: 200 OK or 400 BAD REQUEST with JSON document.	National Verifier states: Service that allows ETC to transfer benefits of a subscriber from the current ETC claiming benefits or returns rejection codes.
NV PUT /subscriber	Arguments: Body contains all subscriber fields required for the enroll transaction and phoneNumberInNlad for subscriber identification in JSON format. Returns: 200 OK or 400 BAD REQUEST with JSON document.	National Verifier states: Service that allows ETC to update a current subscriber's information. The subscriber information is validated, then the subscriber is either updated or the service returns rejection codes which may require document uploads through the National Verifier web portal.
NV POST /verify	Arguments: Body contains all subscriber fields required for the enroll transaction in a JSON format, with the option to omit phoneNumber. Returns: 200 OK with JSON document or 400 Bad Request with JSON document.	National Verifier states: Service to check if potential subscriber will pass all validations and be eligible for enrollment.
NV GET /report/revstatus	Arguments: sac, group, status. Returns: 200 Created with CSV document or 400 Bad Request with JSON document.	Service to allow ETC to request a report of all subscriber reverification statuses.
NV GET /report/revdeenroll	Arguments: sac. Returns: 200 Created with CSV document or 400 Bad Request with JSON document.	Service to allow ETC to request a report of all subscribers de-enrolled for failed reverification.

Operation	Arguments/Returns	Description
NV GET /report/recertstatus	Arguments: sac, type, recertCheckStartDate, recertCheckEndDate. Returns: 200 Created with CSV document or 400 Bad Request with JSON document.	Service to allow ETC to request a report of all subscriber recertification statuses.
NV GET /report/recertdeenroll	Arguments: sac, startDate, endDate. Returns: 200 Created with CSV document or 400 Bad Request with JSON document.	Service to allow ETC to request a report of all subscribers de-enrolled for failed recertification.

6.2. Batch Subscriber Upload

6.2.1. Post Batch

The batch upload API is an HTTP POST method that is used to perform multiple transactions from a single file upload, where each transaction is a separate row in that file. This method will conduct duplicate-subscriber check, duplicate-address check, required-field validations, and third-party identification verification for each subscriber before performing the transaction.

The batch upload API accepts a CSV document in the body of the POST.

Restrictions

- The file must be in CSV format and must have the .csv extension (if you change the extension of a non-CSV file, like an Excel document, to ".csv" you will receive the error code: "TOO_FEW_HEADERS").
- The batch file must include all subscriber fields found in the input template as the header row (first row).
- File name cannot contain a space or any of the following characters: \ / : * ? " < > |
- The batch file name must start with the six-digit SAC number. You may append additional alphanumeric characters to the file name for your own reference, but you must compose the file name in this format: *six-digit SAC number*, *hyphen*, then up to *43 alphanumeric characters* of your choice. For example, **999999-batch_99_NOV.csv**.
- You may only submit one batch file at a time.

Each transaction in a batch file is processed individually, and in the order each appears in the batch file, from top to bottom.

This API will return a response when the file has been ingested by the service.

Code Block 2 Request

```
POST /batch
Content-Type: multipart/form-date
```

Code Block 3 Response

```
HTTP/1.1 200 OK
[{
      "message": "File submitted."
}]
```

Code Block 4 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
      "header": {
            "failureType": "Validation Error"
      },
      "body": [["error","SAC number is not 6 digits.","SAC_NOT_SIX_DIGITS"]]
}
```

If you submit more than one batch file at a time, you will receive the following error message with no header:

Code Block 5 Multi-File Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

[{
      "error":"This service only accepts one file at a time."
}]
```

6.2.2. Get Batch Status

The batch status API is an HTTP GET method that is used to check the status of the most recent batch upload, submitted for each SAC that the user has permission to see. This method does not accept a JSON document, and does not require a query string. The batch status API returns a JSON document with a unique object for each batch for that user. Each JSON object includes these fields:

- **filename** - Name of the submitted file.
- **sac** - The SAC number of the submitted file.
- **batchId** - The NLAD batch identifier.
- **uploadDateTime** - The date and time the file was uploaded to NLAD.
- **statusCode** - The batch status code. The possible batch status codes are:

  - ERRORS - Batch processed with errors. Check the rejected-row file for more information.
  - PLEASE_RESUBMIT - The file needs to be resubmitted.
  - REJECTED - The file was rejected.
  - IN_PROGRESS - Batch is currently being processed.
  - SUCCESS - Batch was processed without errors.
- **totalRows** - The total number of rows processed.
- **rowsRejected** - The number of rows that were rejected.
- **failedReasonCd** - The error code associated with the failure. The possible failed reason codes are:
  - TOO_FEW_HEADERS - File was rejected for having too few headers.
  - TOO_MANY_HEADERS - File was rejected for having too many headers.
  - PROCESSING_ERROR - There was an unspecified error processing the file.
  - SYSTEM_ERROR - There was an unspecified error.
  - HEADERS_INVALID - File contains invalid headers.
  - FILE_EMPTY - File contains no transactions.
- **user** - The submitting user name for the NLAD API account.
- **rejectedRowFile** - The rejected rows CSV file that contains the rejected transaction details (when available).

Code Block 6 Request

```
GET /batch?rowsProcessed
```

Code Block 7 Response

```
HTTP/1.1 200 OK
Content-Type: application/json

[
     {
     "filename":"999999-5000.csv",
     "sac":999999,
     "batchId":57252,
     "uploadDateTime":"2013-09-24T10:57:59",
     "statusCode":"REJECTED",
     "rowsProcessed":0,
     "totalRows":0,
     "rowsRejected":0,
     "failedReasonCd":"HEADERS_INVALID",
     "user":"user1",
     "rejectedRowFile":""
     },
     {
     "filename":"999998.csv",
```

```
      "sac":999998,
      "batchId":62500,
      "uploadDateTime":"2013-10-10T08:49:01",
      "statusCode":"ERRORS",
      "rowsProcessed":20,
      "totalRows":20,
      "rowsRejected":19,
      "failedReasonCd":null,
      "user":"user1",
      "rejectedRowFile":"/svc/1/report/62500"
      },
      {
      "filename":"999997-daily.csv",
      "sac":999997,
      "batchId":56108,
      "uploadDateTime":"2013-09-18T10:47:05",
      "statusCode":"SUCCESS",

      "rowsProcessed":75,
      "totalRows":75,
      "rowsRejected":0,
      "failedReasonCd":null,
      "user":"user1",
      "rejectedRowFile":""
      }
]
```

Code Block 8 Data Options

```
rowsProcessed: Use query string "rowsProcessed" (/batch?rowsProcessed) to include
the rows processed in the response message.
```

Code Block 9 Error Response

```
No application specific error messages returned.
```

6.3. Enroll Subscriber

The enroll subscriber API is an HTTP POST method that is used to enroll a single subscriber into NLAD. This method will conduct duplicate-subscriber check, duplicate-address check, required-field validations, and third-party identification verification before enrolling the subscriber.

The enroll subscriber API accepts a JSON document in the body of the POST, which must include all of the subscriber fields (see **Appendix A - Data Types and Descriptions** for more information), as represented in the following API specification.

Code Block 10 Request

```
POST /subscriber
Content-Type: application/json

{
     "transactionType": "enroll",
     "transactionEffectiveDate": "12/12/2000",
     "sac": "999999",
     "lastName": "DOE",
     "firstName": "JOHN",
     "middleName": "Wayne",
     "phoneNumber": "5555555555",
     "phoneNumberInNlad": "",
     "last4ssn": "1234",
     "tribalId": "1234abcd",
     "dob": "04/21/1966",
     "serviceType": "Voice",
     "iehFlag": "1",
     "iehCertificationDate": "09/01/2012",
     "iehRecertificationDate": "12/12/2012",
     "primaryAddress1": "175 E 196TH ST",
     "primaryAddress2": "APT 1138",
     "primaryCity": "NEW YORK",
     "primaryState": "NY",
     "primaryZipCode": "10001",
     "primaryUrbanizationCode": "",
     "primaryPermanentAddressFlag": "0",
     "primaryTribalFlag": "0",
     "primaryRuralFlag": "0",
     "mailingAddress1": "175 E 196TH ST",
     "mailingAddress2": "APT 1138",
     "mailingCity": "NEW YORK",
     "mailingState": "NY",
     "mailingZipCode": "10001",
     "mailingUrbanizationCode": "",
     "serviceInitializationDate": "01/23/2012",
     "serviceReverificationDate": "07/11/2012",
     "eligibilityCode": "E13",
     "bqpLastName": "DOE",
     "bqpFirstName": "Jane",
     "bqpMiddleName": "Wanda",
     "bqpDob": "08/22/1988",
     "bqpLast4ssn": "2234",
     "bqpTribalId": "",
     "linkUpServiceDate": "",
     "lifelineTribalBenefitFlag": "0",
     "etcGeneralUse": "A-123456",
     "tpivFlag": "0",
     "includeSubscriberId": "0"
}
```

Code Block 11 Response

```
HTTP/1.1 201 Created
Content-Type: application/json
```

```
[{"message":"Subscriber successfully enrolled"}]
```

Code Block 12 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
     "header": {
            "resolutionId": "999999-JCYF71",
            "failureType": "Validation Error"
     },
     "body": [
     ["dob","No subscriber can be less than 18 years of
age.","SUBSCRIBER_UNDER_18","Sat Apr 21 00:00:00 EDT 2007"]
     ]
}
```

Error Response Info

Failure types:

- Duplicate subscriber: Subscriber matches an existing subscriber.
- Duplicate address: Subscriber's primary address matches an existing subscriber's address.
- Duplicate phone number: Subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate subscriber/duplicate address: Both the subscriber and the subscriber's address match existing subscriber(s).
- Duplicate subscriber/duplicate phone number: Subscriber matches an existing subscriber, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate address/duplicate phone number: Subscriber's primary address matches an existing subscriber's address, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate subscriber/duplicate address/duplicate phone number: Subscriber matches an existing subscriber, subscriber's primary address matches an existing subscriber's address, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Malformed Document: Format or structure of the batch file is incorrect.
- Validation Error: A field or fields has failed one or more of the validations. Validation errors can include Address Matching Service errors and Third Party Identification Verification errors.

6.4. NV Enroll Subscriber

The enroll subscriber API is an HTTP POST method that is used to enroll a single subscriber into NLAD. This method will confirm required-field validations, check qualification status in LED and conduct a duplicate subscriber and duplicate address check before enrolling the subscriber.

The enroll subscriber API accepts a JSON document in the body of the POST, which must include all of the subscriber fields (see **Appendix A - Data Types and Descriptions** for more information), as represented in the following API specification.

Code Block 13 Request

```
POST /subscriber
Content-Type: application/json

{
      "nv": "1",
      "transactionType": "enroll",
      "transactionEffectiveDate": "12/12/2000",
      "sac": "123456",
      "lastName": "DOE",
      "firstName": "JOHN",
      "middleName": "Wayne",
      "phoneNumber": "5555555555",
      "phoneNumberInNlad": "",
      "last4ssn": "1234",
      "tribalId": "1234abcd",
      "dob": "04/21/1966",
      "serviceType": "Voice",
      "primaryAddress1": "175 E 196TH ST",
      "primaryAddress2": "APT 1138",
      "primaryCity": "NEW YORK",
      "primaryState": "NY",
      "primaryZipCode": "10001",
      "primaryUrbanizationCode": "",
      "mailingAddress1": "175 E 196TH ST",
      "mailingAddress2": "APT 1138",
      "mailingCity": "NEW YORK",
      "mailingState": "NY",
      "mailingZipCode": "10001",
      "mailingUrbanizationCode": "",
      "serviceInitializationDate": "01/23/2012",
      "bqpLastName": "DOE",
      "bqpFirstName": "Jane",
      "bqpMiddleName": "Wanda",
      "bqpDob": "08/22/1988",
      "bqpLast4ssn": "2234",
      "bqpTribalId": "",
      "linkUpServiceDate": "",
      "lifelineTribalBenefitFlag": "0",
      "etcGeneralUse": "A-123456",
      "includeSubscriberId": "0"
}
```

Code Block 14 Response

```
HTTP/1.1 201 Created
Content-Type: application/json

[{
      "message":"Subscriber successfully enrolled"
}]
```

Code Block 15 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
      "header": {
            "failureType": "Validation Error"
      },
      "body": [
      ["applicant","The subscriber has not qualified through the Lifeline National
Verifier yet or their application has expired. You can qualify them now at
checklifeline.org","APPLICATION_NOT_FOUND"]
      ]
}
```

Error Response Info

Failure types:

- Duplicate subscriber: Subscriber matches an existing subscriber.
- Duplicate address: Subscriber's primary address matches an existing subscriber's address.
- Duplicate phone number: Subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate subscriber/duplicate address: Both the subscriber and the subscriber's address match existing subscriber(s).
- Duplicate subscriber/duplicate phone number: Subscriber matches an existing subscriber, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate address/duplicate phone number: Subscriber's primary address matches an existing subscriber's address, and subscriber's phone number matches an existing subscriber's phone number in NLAD.

- Duplicate subscriber/duplicate address/duplicate phone number: Subscriber matches an existing subscriber, subscriber's primary address matches an existing subscriber's address, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Malformed Document: Format or structure of the batch file is incorrect.
- Validation Error: A field or fields have failed one or more of the validations. Refer to Appendix B for validation errors.

6.5. De-Enroll Subscriber

The de-enroll subscriber API is an HTTP DELETE method that is used to de-enroll a single existing Lifeline subscriber from NLAD. This method identifies a subscriber based on their current subscriber phone number or their subscriber ID.

The de-enroll subscriber API accepts a JSON document in the body of the DELETE, which must include the de-enroll fields found in the following API specification. Note: the de-enroll subscriber API only requires the three fields listed below, even when included within a batch file, though the header row of the batch file must still contain all fields.

Each de-enroll transaction must use the appropriate de-enroll code from the list below:

- `deEnrollDeceased` – Used to de-enroll a Lifeline subscriber who has deceased.
- `deEnrollLeaving` – Used to de-enroll a Lifeline subscriber who is opting out of the program, or is no longer eligible for benefits.
- `deEnrollFailedRecertification` – Used to de-enroll a Lifeline subscriber who has not filed their annual recertification.
- `deEnrollNonUsage` – Used to de-enroll a Lifeline subscriber who has not used their benefits for 60 days.

Code Block 16 Request

```
DELETE /subscriber
Content-Type: application/json

{
     "transactionType": "deEnrollLeaving",
     "phoneNumberInNlad": "7035555555",
     "subscriberId": "",
     "transactionEffectiveDate": "07/23/2013"

}
```

Code Block 17 Response

```
HTTP/1.1 200 Ok
Content-Type: application/json

[{
      "message": "Subscriber de-enrolled."
}]
```

Code Block 18 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
      "header": {
            "failureType": "Validation Error"
      },
      "body": [
      ["phoneNumberInNlad","Subscriber Not
Found","SUBSCRIBER_NOT_FOUND_ERROR","5555555555"]
      ]
}
```

Error Response Info

Failure types:

- Malformed Document: Format or structure of the batch file is incorrect.
- Validation Error: A field or fields has failed one or more of the validations. Validation errors can include Address Matching Service errors and Third Party Identification Verification errors.

6.6. De-Enroll Duplicate Subscriber

The de-enroll duplicate subscriber API is an HTTP DELETE method that is used to de-enroll a single existing Lifeline subscriber from the NLAD duplicate resolution process. This method identifies a subscriber based on their current, subscriber phone number.

The de-enroll duplicate subscriber API accepts a JSON document in the body of the DELETE, which must include the de-enroll fields found in the following API specification. Note: the de-enroll duplicate subscriber API only requires the three fields listed below.

Each de-enroll duplicate subscriber transaction must use the appropriate de-enroll code from the list below:

- `deEnrollDeceased` – Used to de-enroll a Lifeline subscriber who has deceased.

- `deEnrollLeaving` – Used to de-enroll a Lifeline subscriber who is opting out of the program, or is no longer eligible for benefits.
- `deEnrollFailedRecertification` – Used to de-enroll a Lifeline subscriber who has not filed their annual recertification.
- `deEnrollNonUsage` – Used to de-enroll a Lifeline subscriber who has not used their benefits for 60 days.

Code Block 19 Request

```
DELETE /duplicate
Content-Type: application/json

{
      "transactionType": "deEnrollLeaving",
      "phoneNumberInNlad": "7035555555",
      "transactionEffectiveDate": "07/23/2013"

}
```

Code Block 20 Response

```
HTTP/1.1 200 Ok
Content-Type: application/json

[{
      "message": "Subscriber de-enrolled."
}]
```

Code Block 21 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
      "header": {
            "failureType": "Validation Error"
      },
      "body": [
      ["phoneNumberInNlad","Subscriber Not
Found","SUBSCRIBER_NOT_FOUND_ERROR","5555555555"]
      ]
}
```

Error Response Info

Failure types:

- Malformed Document: Format or structure of the batch file is incorrect.

- Validation Error: A field or fields has failed one or more of the validations. Validation errors can include Address Matching Service errors and Third Party Identification Verification errors.

6.7. Update Subscriber

The update subscriber API is an HTTP PUT method that is used to update an existing Lifeline subscriber's information in NLAD. This method will conduct required-field validations, and may conduct third-party identification verification, and address-matching service verification before the subscriber's information is updated in the system.

The update subscriber API accepts a JSON document in the body of the PUT, which must include the subscriber's phone number in NLAD or subscriber ID, and all of the subscriber fields (see **Appendix A - Data Types and Descriptions** for more information), as represented in the following API specification. This data will replace the existing data in NLAD. Note: sending blank information in optional fields will remove the existing data from those fields in NLAD.

Code Block 22 Request

```
PUT /subscriber
Content-Type: application/json
{
      "transactionType": "update",
      "phoneNumberInNlad": "7035555555",
      "subscriberId": "",
      "transactionEffectiveDate": "12/12/2000",
      "sac": "123456",
      "lastName": "DOE",
      "firstName": "JOHN",
      "middleName": "Wayne",
      "phoneNumber": "5555555555",
      "last4ssn": "1234",
      "tribalId": "1234abcd",
      "dob": "04/21/1966",
      "serviceType": "Voice",
      "iehFlag": "1",
      "iehCertificationDate": "09/01/2012",
      "iehRecertificationDate": "12/12/2012",
      "primaryAddress1": "175 E 196TH ST",
      "primaryAddress2": "APT 1138",
      "primaryCity": "NEW YORK",
      "primaryState": "NY",
      "primaryZipCode": "10001",
      "primaryUrbanizationCode": "",
      "primaryPermanentAddressFlag": "1",
      "primaryTribalFlag": "1",
      "primaryRuralFlag": "1",
      "mailingAddress1": "175 E 196TH ST",
      "mailingAddress2": "APT 1138",
      "mailingCity": "NEW YORK",
      "mailingState": "NY",
      "mailingZipCode": "10001",
      "mailingUrbanizationCode": "",
      "serviceInitializationDate": "01/23/2012",
```

```
      "serviceReverificationDate": "07/11/2012",
      "eligibilityCode": "E13",
      "bqpLastName": "DOE",
      "bqpFirstName": "Jane",
      "bqpMiddleName": "Wanda",
      "bqpDob": "08/22/1988",
      "bqpLast4ssn": "2234",
      "bqpTribalId": "1234abcd",
      "linkUpServiceDate": "08/23/2003",
      "lifelineTribalBenefitFlag": "0",
      "etcGeneralUse": "A-123456X",
      "tpivFlag": "0"
}
```

Code Block 23 Response

```
HTTP/1.1 200 OK
Content-Type: application/json

[{
      "message": "Subscriber successfully updated"
}]
```

Code Block 24 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
      "header": {
            "resolutionId": "999999-TLNENK",
            "failureType": "Validation Error"
      },
      "body": [
      ["Primary Address","Address Not Found. ZIP4 did not
match.","AMS_FAILURE_ANALYSIS","9999 Neverwhere st. Faketopia VA 99999"]
      ]
}
```

Error Response Info

Failure types:

- Duplicate subscriber: Subscriber matches an existing subscriber.
- Duplicate address: Subscriber's primary address matches an existing subscriber's address.

- Duplicate phone number: Subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate subscriber/duplicate address: Both the subscriber and the subscriber's address match existing subscriber(s).
- Duplicate subscriber/duplicate phone number: Subscriber matches an existing subscriber, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate address/duplicate phone number: Subscriber's primary address matches an existing subscriber's address, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate subscriber/duplicate address/duplicate phone number: Subscriber matches an existing subscriber, subscriber's primary address matches an existing subscriber's address, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Malformed Document: Format or structure of the batch file is incorrect.
- Validation Error: A field or fields has failed one or more of the validations. Validation errors can include Address Matching Service errors and Third Party Identification Verification errors.

6.8. NV Update Subscriber

The update subscriber API is an HTTP PUT method that is used to update an existing Lifeline subscriber's information in NLAD. This method will confirm required-field validations, check qualification status in LED and conduct a duplicate subscriber and duplicate address check before updating the subscriber.

The update subscriber API accepts a JSON document in the body of the PUT, which must include the subscriber's phone number in NLAD or subscriber ID, and all of the subscriber fields (see **Appendix A - Data Types and Descriptions** for more information), as represented in the following API specification. This data will replace the existing data in NLAD. Note: sending blank information in optional fields will remove the existing data from those fields in NLAD.

Code Block 25 Request

```
PUT /subscriber
Content-Type: application/json

{
      "nv": "1",
      "transactionType": "update",
      "phoneNumberInNlad": "7035555555",
      "subscriberId": "",
      "transactionEffectiveDate": "12/12/2000",
      "sac": "123456",
      "lastName": "DOE",
      "firstName": "JOHN",
      "middleName": "Wayne",
      "phoneNumber": "5555555555",
      "last4ssn": "1234",
```

```
      "tribalId": "1234abcd",
      "dob": "04/21/1966",
      "serviceType": "Voice",
      "primaryAddress1": "175 E 196TH ST",
      "primaryAddress2": "APT 1138",
      "primaryCity": "NEW YORK",
      "primaryState": "NY",
      "primaryZipCode": "10001",
      "primaryUrbanizationCode": "",
      "mailingAddress1": "175 E 196TH ST",
      "mailingAddress2": "APT 1138",
      "mailingCity": "NEW YORK",
      "mailingState": "NY",
      "mailingZipCode": "10001",
      "mailingUrbanizationCode": "",
      "serviceInitializationDate": "01/23/2012",
      "bqpLastName": "DOE",
      "bqpFirstName": "Jane",
      "bqpMiddleName": "Wanda",
      "bqpDob": "08/22/1988",
      "bqpLast4ssn": "2234",
      "bqpTribalId": "1234abcd",
      "linkUpServiceDate": "08/23/2003",
      "lifelineTribalBenefitFlag": "0",
      "etcGeneralUse": "A-123456X"
}
```

Code Block 26 Response

```
HTTP/1.1 200 OK
Content-Type: application/json

[{
      "message": "Subscriber successfully updated"
}]
```

Code Block 27 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
      "header": {
            "failureType": "Validation Error"
      },
      "body": [
      ["applicant","The subscriber has not qualified through the Lifeline National
Verifier yet or their application has expired. You can qualify them now at
checklifeline.org","APPLICATION_NOT_FOUND"]
      ]
}
```

Error Response Info

Failure types:

- Duplicate subscriber: Subscriber matches an existing subscriber.
- Duplicate address: Subscriber's primary address matches an existing subscriber's address.
- Duplicate phone number: Subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate subscriber/duplicate address: Both the subscriber and the subscriber's address match existing subscriber(s).
- Duplicate subscriber/duplicate phone number: Subscriber matches an existing subscriber, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate address/duplicate phone number: Subscriber's primary address matches an existing subscriber's address, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate subscriber/duplicate address/duplicate phone number: Subscriber matches an existing subscriber, subscriber's primary address matches an existing subscriber's address, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Malformed Document: Format or structure of the batch file is incorrect.
- Validation Error: A field or fields have failed one or more of the validations. Refer to Appendix B for validation errors.

6.9. Update Duplicate Subscriber

The update duplicate subscriber API is an HTTP PUT method that is used to update an existing Lifeline subscriber's information in NLAD, which has been identified in the Duplicate Resolution Process. This method will conduct required-field validations, and may conduct third-party identification verification, and address-matching service verification before the subscriber's information is updated in the system.

The update duplicate subscriber API accepts a JSON document in the body of the PUT, which must include the subscriber's phone number in NLAD and all of the subscriber fields (see **Appendix A - Data Types and Descriptions** for more information), as represented in the following API specification. This data will only replace the existing phone number data in NLAD. Note: sending blank information in optional fields will remove the existing data from those fields in NLAD.

Code Block 28 Request

```
PUT /duplicate


Content-Type: application/json


{
    "transactionType": "update",
    "phoneNumberInNlad": "7035555555",
    "transactionEffectiveDate": "12/12/2000",
    "sac": "123456",
    "lastName": "DOE",
    "firstName": "JOHN",
    "middleName": "Wayne",
    "phoneNumber": "5555555555",
    "last4ssn": "1234",
    "tribalId": "1234abcd",
    "dob": "04/21/1966",
     "serviceType": "Voice",
    "iehFlag": "1",
    "iehCertificationDate": "09/01/2012",
    "iehRecertificationDate": "12/12/2012",
    "primaryAddress1": "175 E 196TH ST",
    "primaryAddress2": "APT 1138",
    "primaryCity": "NEW YORK",
    "primaryState": "NY",
    "primaryZipCode": "10001",
    "primaryPermanentAddressFlag": "1",
    "primaryTribalFlag": "1",
    "primaryRuralFlag": "1",
    "mailingAddress1": "175 E 196TH ST",
    "mailingAddress2": "APT 1138",
    "mailingCity": "NEW YORK",
    "mailingState": "NY",
    "mailingZipCode": "10001",
    "serviceInitializationDate": "01/23/2012",
    "serviceReverificationDate": "07/11/2012",
    "eligibilityCode": "E13",
    "bqpLastName": "DOE",
    "bqpFirstName": "Jane",
    "bqpMiddleName": "Wanda",
    "bqpDob": "08/22/1988",
    "bqpLast4ssn": "2234",
    "bqpTribalId": "1234abcd",
    "linkUpServiceDate": "08/23/2003",
    "lifelineTribalBenefitFlag": "0",
    "etcGeneralUse": "A-123456X",
    "tpivFlag": "0"
}
```

Code Block 29 Response

```
HTTP/1.1 200 OK
Content-Type: application/json

[{
```

```
      "message": "Subscriber successfully updated"
}]
```

Code Block 30 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
      "header": {
            "resolutionId": "999999-TLNENK",
            "failureType": "Validation Error"
      },
      "body": [
      ["Primary Address","Address Not Found. ZIP4 did not
match.","AMS_FAILURE_ANALYSIS","9999 Neverwhere st. Faketopia VA 99999"]
      ]
}
```

Error Response Info

Failure types:

- Duplicate Phone Number: Subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Malformed Document: Format or structure of the batch file is incorrect.
- Validation Error: A field or fields has failed one or more of the validations. Validation errors can include Address Matching Service errors and Third Party Identification Verification errors.

6.10. Benefit Transfer

The benefit transfer API is an HTTP POST method that is used to transfer a single, existing subscriber from one ETC to another, without removing that subscriber from NLAD. This method will conduct required-field validations, and may conduct third-party identification verification and address-matching service verification before transferring the subscriber.

The benefit transfer API accepts a JSON document in the body of the POST, which may include the subscriber's phone number on file but it is not required. The remaining fields are the same as the subscriber fields found in the input template, as seen in the following API specification.

Code Block 31 Request

```
POST /transfer
Content-Type: application/json
```

```
{
      "transactionType": "transfer",
      "phoneNumberInNlad": "7035555555",
      "transactionEffectiveDate": "12/12/2000",
      "sac": "123456",
      "lastName": "DOE",
      "firstName": "JOHN",
      "middleName": "Wayne",
      "phoneNumber": "5555555555",
      "last4ssn": "1234",
      "tribalId": "1234abcd",
      "dob": "04/21/1966",
      "serviceType": "Voice",
      "iehFlag": "1",
      "iehCertificationDate": "09/01/2012",
      "iehRecertificationDate": "12/12/2012",
      "primaryAddress1": "175 E 196TH ST",
      "primaryAddress2": "APT 1138",
      "primaryCity": "NEW YORK",
      "primaryState": "NY",
      "primaryZipCode": "10001",
      "primaryUrbanizationCode": "",
      "primaryPermanentAddressFlag": "1",
      "primaryTribalFlag": "1",
      "primaryRuralFlag": "1",
      "mailingAddress1": "175 E 196TH ST",
      "mailingAddress2": "APT 1138",
      "mailingCity": "NEW YORK",
      "mailingState": "NY",
      "mailingZipCode": "10001",
      "mailingUrbanizationCode": "",
      "serviceInitializationDate": "01/23/2012",
      "serviceReverificationDate": "07/11/2012",
      "eligibilityCode": "E13",
      "bqpLastName": "DOE",
      "bqpFirstName": "Jane",
      "bqpMiddleName": "Wanda",
      "bqpDob": "08/22/1988",
      "bqpLast4ssn": "2234",
      "bqpTribalId": "1234abcd",
      "linkUpServiceDate": "08/23/2003",
      "lifelineTribalBenefitFlag": "0",
      "etcGeneralUse": "A-123456X",
      "tpivFlag": "0",
      "includeSubscriberId": "0"
}
```

Code Block 32 Response

```
HTTP/1.1 200 OK
Content-Type: application/json

[{
      "message": "Subscriber successfully transferred"
}]
```

Code Block 33 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json


{
     "header": {
            "resolutionId": "",
            "failureType": "Validation Error"
     },
     "body": [["sac","Cannot transfer a subscriber to the same SAC
number.","CANNOT_TRANSFER_SAME_SAC","999999"]]
}
```

Error Response Info

Failure types:

- Duplicate subscriber: Subscriber matches an existing subscriber.
- Duplicate address: Subscriber's primary address matches an existing subscriber's address.
- Duplicate phone number: Subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate subscriber/duplicate address: Both the subscriber and the subscriber's address match existing subscriber(s).
- Duplicate subscriber/duplicate phone number: Subscriber matches an existing subscriber, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate address/duplicate phone number: Subscriber's primary address matches an existing subscriber's address, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate subscriber/duplicate address/duplicate phone number: Subscriber matches an existing subscriber, subscriber's primary address matches an existing subscriber's address, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Malformed Document: Format or structure of the batch file is incorrect.
- Validation Error: A field or fields has failed one or more of the validations. Validation errors can include Address Matching Service errors and Third Party Identification Verification errors.

6.11. NV Benefit Transfer

The benefit transfer API is an HTTP POST method that is used to transfer a single, existing subscriber from one ETC to another, without removing that subscriber from NLAD. This method will confirm required-field validations, check qualification status in LED and conduct a duplicate subscriber and duplicate address check before transferring the subscriber.

The benefit transfer API accepts a JSON document in the body of the POST, which may include the subscriber's phone number on file but it is not required. The remaining fields are the same as the subscriber fields found in the input template, as seen in the following API specification.

Code Block 34 Request

```
POST /transfer
Content-Type: application/json

{
      "nv": "1",
      "transactionType": "transfer",
      "phoneNumberInNlad": "7035555555",
      "transactionEffectiveDate": "12/12/2000",
      "sac": "123456",
      "lastName": "DOE",
      "firstName": "JOHN",
      "middleName": "Wayne",
      "phoneNumber": "5555555555",
      "last4ssn": "1234",
      "tribalId": "1234abcd",
      "dob": "04/21/1966",
      "serviceType": "Voice",
      "primaryAddress1": "175 E 196TH ST",
      "primaryAddress2": "APT 1138",
      "primaryCity": "NEW YORK",
      "primaryState": "NY",
      "primaryZipCode": "10001",
      "primaryUrbanizationCode": "",
      "mailingAddress1": "175 E 196TH ST",
      "mailingAddress2": "APT 1138",
      "mailingCity": "NEW YORK",
      "mailingState": "NY",
      "mailingZipCode": "10001",
      "mailingUrbanizationCode": "",
      "serviceInitializationDate": "01/23/2012",
      "bqpLastName": "DOE",
      "bqpFirstName": "Jane",
      "bqpMiddleName": "Wanda",
      "bqpDob": "08/22/1988",
      "bqpLast4ssn": "2234",
      "bqpTribalId": "1234abcd",
      "linkUpServiceDate": "08/23/2003",
      "lifelineTribalBenefitFlag": "0",
      "etcGeneralUse": "A-123456X",
      "includeSubscriberId": "0"
}
```

Code Block 35 Response

```
HTTP/1.1 200 OK
Content-Type: application/json

[{
      "message": "Subscriber successfully transferred"
}]
```

Code Block 36 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
      "header": {
            "failureType": "Validation Error"
      },
      "body": [["applicant","The subscriber has not qualified through the Lifeline
National Verifier yet or their application has expired. You can qualify them now
at checklifeline.org","APPLICATION_NOT_FOUND"]
      ]
}
```

Error Response Info

Failure types:

- Duplicate subscriber: Subscriber matches an existing subscriber.
- Duplicate address: Subscriber's primary address matches an existing subscriber's address.
- Duplicate phone number: Subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate subscriber/duplicate address: Both the subscriber and the subscriber's address match existing subscriber(s).
- Duplicate subscriber/duplicate phone number: Subscriber matches an existing subscriber, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate address/duplicate phone number: Subscriber's primary address matches an existing subscriber's address, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate subscriber/duplicate address/duplicate phone number: Subscriber matches an existing subscriber, subscriber's primary address

```
   matches an existing subscriber's address, and subscriber's phone number
   matches an existing subscriber's phone number in NLAD.

•  Malformed Document: Format or structure of the batch file is incorrect.

•  Validation Error: A field or fields have failed one or more of the
   validations. Refer to Appendix B for validation errors.
```

6.12. Submit Resolution Request

The submit resolution request API is an HTTP POST method that is used to submit a resolution ID to the NLAD Customer Service help desk to begin the resolution process for a rejected transaction. A unique resolution ID is automatically generated and returned when transactions fail certain validations.

The submit resolution request API accepts a JSON document in the body of the POST that includes a resolution ID, agent name and/or ID, resolution code(s), optional description of the issue, and certification flag. Please refer to the Dispute Resolution page on the USAC.org website for the resolution codes.

Code Block 37 Request

```
POST /resolution/submit
Content-Type: application/json

{
     "resolutionId": "999999-ABCDEF",
     "agentName": "Joe Smith",
     "agentId":"1234ABCD",
     "fullNameLast4ssnTCode":"T3",
     "fullNameDobTCode":"T1",
     "fullNameDeceasedTCode":"T14",
     "aCode":"A1",
     "mCode":"M1",
     "desc": "",
     "certificationFlag":"y"
}
```

Code Block 38 Response

```
HTTP/1.1 201 Created
[{
     "message": "A case has been successfully created with NLAD Customer Service
for this resolution ID."
}]
```

Code Block 39 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json
 {
     "header": {
            "failureType": "Validation Error"
     },
     "body": [
     ["invalidResolutionCode", "The resolution code(s) provided do not meet the
criteria for the transaction failure types.  Please refer to the Dispute
Resolution page on the USAC.org website.", "INVALID_RESOLUTION_CODE"]
     ]
}
```

Error Response Info

Failure types:

- Malformed Document: Format or structure of the batch file is incorrect.
- Validation Error: A field or fields has failed one or more of the validations. Validation errors can include Address Matching Service errors and Third Party Identification Verification errors.

6.13. ETC Check Subscriber POST

The ETC check subscriber API is an HTTP POST method that allows an ETC to check if a potential subscriber and their address is already registered in the Lifeline program. This method will search for a subscriber based on the subscriber's last name, date of birth, last four digits of their social security number and their primary address. Date format must be: MM/DD/YYYY. If a social security number is not provided, a Tribal ID must be provided.

The ETC check subscriber query API accepts a JSON document in the body of the POST, which must include the following subscriber fields:

- lastName
- firstName
- dob
- last4ssn or tribalId
- primaryAddress1
- primaryCity
- primaryState
- primaryZipCode

Code Block 40 Request

```
POST /subscriber/lookupSubscriber
Content-Type: application/json

{
      "lastName": "DOE",
      "firstName": "JOHN",
      "last4ssn": "1234",
      "tribalId": "1234abcd",
      "dob": "04/21/1966",
      "primaryAddress1": "175 E 196TH ST",
      "primaryAddress2": "APT 1138",
      "primaryCity": "NEW YORK",
      "primaryState": "NY",
      "primaryZipCode": "10001",
      "primaryUrbanizationCode": ""
}
```

Code Block 41 Response

```
HTTP/1.1 200 OK
Content-Type: application/json
[{
      "message": "Address found and subscriber not found"
}]
```

Code Block 42 Response Info

```
Possible success messages:
"Subscriber and address not found"
"Subscriber and matching address found."
"Address found and subscriber not found"
```

Code Block 43 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
      "header": {
            "failureType": "Malformed Document"
      },
      "body": [
      ["dob","Subscriber's date of birth is required.","DOB_REQUIRED"],
      ["last4ssn","Must provide the last four digits of the subscriber's social
security number or the subscriber's Tribal ID.","EMPTY_L4DSSN_AND_TRIBALID"],
      ["lastName","Subscriber's last name is required.","LASTNAME_REQUIRED"]
      ]
}
```

Error Response Info

Failure types:

- Malformed Document: Format or structure of the batch file is incorrect.
- Validation Error: A field or fields has failed one or more of the validations. Validation errors can include Address Matching Service errors and Third Party Identification Verification errors.

6.14. ETC Check Subscriber (Deprecated)

The ETC check subscriber API is an HTTP GET method that allows an ETC to check if a potential subscriber and their address is already registered in the Lifeline program. This method will search for a subscriber based on the subscriber's last name, date of birth, last four digits of their social security number and their primary address. Date format must be: MM/DD/YYYY. If a social security number is not provided, a Tribal ID must be provided.

The ETC check subscriber query string is built in the URL of the GET request, and the format for the query string can be found in this following API specification.

Code Block 44 Request

```
GET
/subscriber?last4ssn=1234&tribalId=&lastName=doe&dob=11/05/1985&primaryAddress=175
E 196th ST, APT 1138&primaryCityStateZip=New York, NY,
10001&primaryUrbanizationCode=EXAMPLE
```

Code Block 45 Response

```
HTTP/1.1 200 OK
Content-Type: application/json
[{
     "message": "Address found and subscriber not found"
}]
```

Code Block 46 Response Info

```
Possible success messages:
"Subscriber and address not found"
"Subscriber and matching address found."
"Address found and subscriber not found"
```

Code Block 47 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
     "header": {
            "failureType": "Malformed Document"
     },
     "body": [
     ["dob","Subscriber's date of birth is required.","DOB_REQUIRED"],
     ["last4ssn","Must provide the last four digits of the subscriber's social
security number or the subscriber's Tribal ID.","EMPTY_L4DSSN_AND_TRIBALID"],
     ["lastName","Subscriber's last name is required.","LASTNAME_REQUIRED"]
     ]
}
```

Error Response Info

Failure types:

- Malformed Document: Format or structure of the batch file is incorrect.
- Validation Error: A field or fields has failed one or more of the validations. Validation errors can include Address Matching Service errors and Third Party Identification Verification errors.

6.15. Verify Information

The verify information API can be used by an ETC to ensure the validity of a potential subscriber's information before allocating a phone number for them and enrolling them in the system. The verify information API is an exact duplicate of the enroll subscriber API, except the phoneNumber and serviceInitializationDate fields are optional (may be left blank) and the subscriber will not be enrolled into NLAD, even if the record passes all validations and business rules. If a phone number is included, it too will be checked.

The verify information API will never generate a resolution ID.

The verify information API accepts a JSON document in the body of the POST, which must include all of the subscriber fields used in the enroll API, as represented in the following API specification.

Code Block 48 Request

```
POST /verify
Content-Type: application/json

{
    "transactionType": "enroll",
    "transactionEffectiveDate": "12/12/2000",
    "sac": "123456",
    "lastName": "DOE",
```

```
    "firstName": "JOHN",
    "middleName": "Wayne",
    "phoneNumber": "5555555555",
    "phoneNumberInNlad": "",
    "last4ssn": "1234",
    "tribalId": "1234abcd",
    "dob": "04/21/1966",
     "serviceType": "Voice",
    "iehFlag": "1",
    "iehCertificationDate": "09/01/2012",
    "iehRecertificationDate": "12/12/2012",
    "primaryAddress1": "175 E 196TH ST",
    "primaryAddress2": "APT 1138",
    "primaryCity": "NEW YORK",
    "primaryState": "NY",
    "primaryZipCode": "10001",
    "primaryUrbanizationCode": "",
    "primaryPermanentAddressFlag": "0",
    "primaryTribalFlag": "0",
    "primaryRuralFlag": "0",
    "mailingAddress1": "175 E 196TH ST",
    "mailingAddress2": "APT 1138",
    "mailingCity": "NEW YORK",
    "mailingState": "NY",
    "mailingZipCode": "10001",
    "mailingUrbanizationCode": "",
    "serviceInitializationDate": "01/23/2012",
    "serviceReverificationDate": "07/11/2012",
    "eligibilityCode": "E13",
    "bqpLastName": "DOE",
    "bqpFirstName": "Jane",
    "bqpMiddleName": "Wanda",
    "bqpDob": "08/22/1988",
    "bqpLast4ssn": "2234",
    "bqpTribalId": "",
    "linkUpServiceDate": "",
    "lifelineTribalBenefitFlag": "0",
    "etcGeneralUse": "A-123456",
    "tpivFlag": "0"
}
```

Code Block 49 Response

```
HTTP/1.1 200 Ok

[{
     "message": "Subscriber passed all validations and verifications"
}]
```

Code Block 50 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
     "header": {
```

```
            "resolutionId": "",
            "failureType": "Duplicate subscriber"
      },
      "body": [
      ["Subscriber","The subscriber in this transaction is a duplicate of another
subscriber.","DUPLICATE_SUBSCRIBER"]
      ]
}
```

Error Response Info

Failure types:

- Duplicate subscriber: Subscriber matches an existing subscriber.
- Duplicate address: Subscriber's primary address matches an existing subscriber's address.
- Duplicate phone number: Subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate subscriber/duplicate address: Both the subscriber and the subscriber's address match existing subscriber(s).
- Duplicate subscriber/duplicate phone number: Subscriber matches an existing subscriber, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate address/duplicate phone number: Subscriber's primary address matches an existing subscriber's address, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate subscriber/duplicate address/duplicate phone number: Subscriber matches an existing subscriber, subscriber's primary address matches an existing subscriber's address, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Malformed Document: Format or structure of the batch file is incorrect.
- Validation Error: A field or fields has failed one or more of the validations. Validation errors can include Address Matching Service errors and Third Party Identification Verification errors.

6.16. NV Verify Information

The verify information API can be used by an ETC to ensure the validity of a potential subscriber's information before allocating a phone number for them and enrolling them in the system. The verify information API is an exact duplicate of the enroll subscriber API, except the phoneNumber and serviceInitializationDate fields are optional (may be left blank) and the subscriber will not be enrolled into NLAD, even if the record passes all validations and business rules. If a phone number is included, it too will be checked.

The verify information API accepts a JSON document in the body of the POST, which must include all of the subscriber fields used in the enroll API, as represented in the following API specification.

Code Block 51 Request

```
POST /verify
Content-Type: application/json

{
      "nv": "1",
      "transactionType": "enroll",
      "transactionEffectiveDate": "12/12/2000",
      "sac": "123456",
      "lastName": "DOE",
      "firstName": "JOHN",
      "middleName": "Wayne",
      "phoneNumber": "5555555555",
      "phoneNumberInNlad": "",
      "last4ssn": "1234",
      "tribalId": "1234abcd",
      "dob": "04/21/1966",
      "serviceType": "Voice",
      "primaryAddress1": "175 E 196TH ST",
      "primaryAddress2": "APT 1138",
      "primaryCity": "NEW YORK",
      "primaryState": "NY",
      "primaryZipCode": "10001",
      "primaryUrbanizationCode": "",
      "mailingAddress1": "175 E 196TH ST",
      "mailingAddress2": "APT 1138",
      "mailingCity": "NEW YORK",
      "mailingState": "NY",
      "mailingZipCode": "10001",
      "mailingUrbanizationCode": "",
      "serviceInitializationDate": "01/23/2012",
      "bqpLastName": "DOE",
      "bqpFirstName": "Jane",
      "bqpMiddleName": "Wanda",
      "bqpDob": "08/22/1988",
      "bqpLast4ssn": "2234",
      "bqpTribalId": "",
      "linkUpServiceDate": "",
      "lifelineTribalBenefitFlag": "0",
      "etcGeneralUse": "A-123456"
}
```

Code Block 52 Response

```
HTTP/1.1 200 Ok

[{
      "message": "Subscriber passed all validations and verifications"
}]
```

Code Block 53 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
     "header": {
           "failureType": "Validation Error"
     },
     "body": [
     ["applicant","The subscriber has not qualified through the Lifeline National
Verifier yet or their application has expired. You can qualify them now at
checklifeline.org","APPLICATION_NOT_FOUND"]
     ]
}
```

Error Response Info

Failure types:

- Duplicate subscriber: Subscriber matches an existing subscriber.
- Duplicate address: Subscriber's primary address matches an existing subscriber's address.
- Duplicate phone number: Subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate subscriber/duplicate address: Both the subscriber and the subscriber's address match existing subscriber(s).
- Duplicate subscriber/duplicate phone number: Subscriber matches an existing subscriber, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate address/duplicate phone number: Subscriber's primary address matches an existing subscriber's address, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Duplicate subscriber/duplicate address/duplicate phone number: Subscriber matches an existing subscriber, subscriber's primary address matches an existing subscriber's address, and subscriber's phone number matches an existing subscriber's phone number in NLAD.
- Malformed Document: Format or structure of the batch file is incorrect.
- Validation Error: A field or fields have failed one or more of the validations. Refer to Appendix B for validation errors.

6.17. ETC Reporting API

ETC reporting API is an HTTP GET method that allows an ETC to generate detailed or summary reports from NLAD listing subscribers, transaction history, or resolution history, depending on the parameters used. Summary reports provide only counts of records that fall within the specified parameters. Detailed reports return the full records of currently active subscribers along with other relevant reporting fields.

6.17.1. ETC Subscriber Report

The ETC Subscriber Report query string is built in the URL of the GET request, and the format for the query string can be found in this following API specification.

Code Block 54 Request

```
GET
/report/subscriber?sac=123456,234567,345678&reportType=detail&startDate=12/12/2012
&endDate=04/21/2013&includeSubscriberId=1
```

Code Block 55 Data Options

```
reportType: summary or detail.
startDate/endDate: summary only. Date format: MM/DD/YYYY
SAC: Any of the SACs the requesting party has authorization to see. Summary
reports can include data from multiple SACs. Detailed reports are limited to a
single SAC.
Detailed reports do not require start and end dates.
includeSubscriberId: Available in detail reportType only. To return the Subscriber
ID in the detailed report, insert a value of 1.
```

Code Block 56 Response

```
HTTP/1.1 200 Created
Content-Type: text/csv

SubscriberDetailReport.csv
```

Code Block 57 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
      "header":{
            "failureType":"Validation Error"
      },
      "body":[
            ["endDate","Date is in the incorrect format. The correct format is:
MM/DD/YYYY.","INVALID_DATE_FORMAT","4/1/13"]
      ]
}
```

6.17.2. ETC Duplicate Subscriber Report

The ETC Duplicate Subscriber Report query string is built in the URL of the GET request, and the format for the query string can be found in this following API specification.

Code Block 58 Request

```
GET /report/migdup?sac=123456
```

Code Block 59 Data Options

```
SAC: Any of the SACs the requesting party has authorization to see. Detailed
reports are limited to a single SAC.
```

Code Block 60 Response

```
HTTP/1.1 200 Created
Content-Type: text/csv

DuplicateSubscriberDetailReport.csv
```

Code Block 61 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
      "header":{
            "failureType":"Validation Error"
      },
      "body":[
            ["sac","The user is not authorized to perform transactions with this
SAC number.","USER_NOT_AUTHORIZED_FOR_SAC"]
      ]
}
```

6.17.3. ETC Duplicate Resolution De-Enroll Report

The ETC Duplicate Resolution De-Enroll Report query string is built in the URL of the GET request, and the format for the query string can be found in this following API specification.

Code Block 62 Request

```
GET /report/dupdeenroll?sac=123456
OR
GET /report/dupdeenroll?sac=123456&duplicateType=DUPLICATE_SUBSCRIBER,
MIGRATION_DUPLICATE_SUBSCRIBER, PRODUCTION_DUPLICATE_SUBSCRIBER
```

Code Block 63 Data Options

```
SAC: Any of the SACs the requesting party has authorization to see. Detailed
reports are limited to a single SAC.
duplicateType: optional, DUPLICATE_SUBSCRIBER, MIGRATION_DUPLICATE_SUBSCRIBER,
PRODUCTION_DUPLICATE_SUBSCRIBER. Reports are limited to a single type.
```

Code Block 64 Response

```
If duplicateType not specified (or) DUPLICATE_SUBSCRIBER type selected
HTTP/1.1 200 Created
Content-Type: text/csv

DuplicateDeEnrollReport.csv
```

Code Block 65 Response

```
If duplicateType is specified as MIGRATION_DUPLICATE_SUBSCRIBER (or)
PRODUCTION_DUPLICATE_SUBSCRIBER
HTTP/1.1 200 Created
Content-Type: text/csv

DuplicateDeEnrollReport_MIGRATION_DUPLICATE_SUBSCRIBER.csv
OR
DuplicateDeEnrollReport_PRODUCTION_DUPLICATE_SUBSCRIBER.csv
```

Code Block 66 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
      "header":(
            "failureType":"Validation Error"
      },
      "body":[
            ["duplicateType","Invalid Duplicate Type. The only acceptable values
for duplicate type are " DUPLICATE_SUBSCRIBER", " MIGRATION_DUPLICATE_SUBSCRIBER",
or " PRODUCTION_DUPLICATE_SUBSCRIBER".","INVALID_DUPLICATE_TYPE"]
      ]
}
```

6.17.4. ETC Transaction Report

The ETC Transaction Report query string is built in the URL of the GET request, and the format for the query string can be found in this following API specification.

Code Block 67 Request

```
GET
/report/transaction?sac=123455,234567&reportType=detail&startDate=12/12/2012&endDa
te=04/21/2013&type=enroll,deenroll,update&includeSubscriberId=1
```

Code Block 68 Data Options

```
reportType: summary or detail.
startDate/endDate: summary or detail. Date format: MM/DD/YYYY
type: enroll, deenroll, update, and transfer. Summary reports can include data
from multiple types. Detailed reports are limited to a single type.
sac: Any of the SACs the requesting party has authorization to see. Summary
reports can include data from multiple SACs. Detailed reports are limited to a
single SAC.
includeSubscriberId: Available in detail reportType only. To return the Subscriber
ID in the detailed report, insert a value of 1.
```

Code Block 69 Response

```
HTTP/1.1 200 Created
Content-Type: text/csv

TransactionReport.csv
```

Code Block 70 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
     "header":{
            "failureType":"Validation Error"
     },
     "body":[
            ["endDate","Date is in the incorrect format. The correct format is:
MM/DD/YYYY.","INVALID_DATE_FORMAT","4/1/13"]
     ]
}
```

6.17.5. ETC Resolution Report

The ETC Resolution Report query string is built in the URL of the GET request, and the format for the query string can be found in this following API specification.

Code Block 71 Request

```
GET
/report/resolution?sac=123456,234567,345678&reportType=summary&startDate=12/12/201
2&endDate=04/21/2013&resolutionRequestStatus=open,in_progress,closed
```

Code Block 72 Data Options

```
reportType: summary or detail
startDate/endDate: summary or detail. Date format: MM/DD/YYYY
resolutionRequestStatus: open, in_progress, and closed. Summary reports can
include data from multiple statuses. Detailed reports are limited to a single
status.
sac: Any of the SACs the requesting party has authorization to see. Summary
reports can include data from multiple SACs. Detailed reports are limited to a
single SAC.
```

Code Block 73 Response

```
HTTP/1.1 200 Created
Content-Type: text/csv

ResolutionDetailReport.csv
```

Code Block 74 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
      "header":{
            "failureType":"Validation Error"
      },
      "body":[
            ["endDate","Date is in the incorrect format. The correct format is:
MM/DD/YYYY.","INVALID_DATE_FORMAT","4/1/13"]
      ]
}
```

6.17.6. ETC Summary and Detail Subscriber Snapshot Report

The ETC Summary and Detail Subscriber Snapshot Report query string is built in the URL of the GET request, and the format for the query string can be found in this following API specification.

Code Block 75 Request

```
GET
/report/subscriber/snapshot?reportType=detail&sac=123456,234567,345678&snapPeriod=
04/2016&includeSubscriberId=1
```

Code Block 76 Data Options

```
reportType: summary or detail.
snapPeriod: Date format: MM/YYYY.  Summary and Detail reports are limited to a
single Snap Period.
SAC: Any of the SACs the requesting party has authorization to see. Summary
reports can include data from multiple SACs. Detailed reports are limited to a
single SAC.
includeSubscriberId: Available in detail reportType only. To return the Subscriber
ID in the detailed report, insert a value of 1.
```

Code Block 77 Response

```
HTTP/1.1 200 Created
Content-Type: text/csv

File name for the Summary report: SubscriberSnapshotSummaryReport.csv
File name for the Detail report: SubscriberSnapshotDetailReport_[SAC#].csv.
Example, when you run the Detail Snapshot Report for SAC 123456, the filename will
be SubscriberSnapshotDetailReport_123456.CSV.
```

Code Block 78 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
      "header":{
            "failureType":"Validation Error"
      },
      "body":[
         ["snapPeriod","SnapPeriod is in the incorrect format. The correct format
is: MM/YYYY","INVALID_SNAP_PERIOD_FORMAT","0416"]
      ]
}
```

6.17.7. ETC Recertification Snapshot Report

The ETC Recertification Snapshot Report query string is built in the URL of the GET request, and the format for the query string can be found in this following API specification.

Code Block 79 Request

```
GET /report/recertificationsnapshot?snapPeriod=JAN-2017
```

Code Block 80 Data Options

```
snapPeriod: Date format: MMM-YYYY. The recertification snapshot reports are
limited to a single Snap Period. MMM should be ALL CAPS.
```

Code Block 81 Response

```
HTTP/1.1 200 Created
Content-Type: text/csv

RecertificationSnapshotReport_MMM-YYYY.csv
Example, when you run the Recertification Snapshot Report for snapPeriod JAN-2017,
the filename will be RecertificationSnapshotReport_JAN-2017.csv.
```

Code Block 82 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
  "header": {
    "failureType": "Validation Error"
  },
  "body": [
    [
      "snapPeriod",
      "SnapPeriod is in the incorrect format. The correct format is: MMM-YYYY",
      "INVALID_RECERT_SNAP_PERIOD_FORMAT",
      "01-2017"
    ]
  ]
}
```

6.18. NV ETC Reporting API

ETC reporting API is an HTTP GET method that allows an ETC to generate detailed or summary reports from NLAD listing subscribers or transaction history depending on the parameters used. Summary reports provide only counts of records that fall within the specified parameters. Detailed reports return the full records of currently active subscribers along with other relevant reporting fields.

6.18.1. ETC Reverification Subscriber Status Report

The ETC Reverification Subscriber Status Report query string is built in the URL of the GET request, and the format for the query string can be found in this following API specification.

Code Block 83 Request

```
GET /report/revstatus?sac=123456,234567&group=1,2,3,4&status=pass,fail
```

Code Block 84 Data Options

```
SAC: Any of the SACs the requesting party has authorization to see. Multiple SAC
selection is available.
Group: Optional parameter. Reverification group can be 1, 2, 3 or 4. If not
specified, then the report will retrieve all groups.
Status: Optional parameter. Reverification status can be pass or fail. If not
specified, then the report will retrieve all statuses.
```

Code Block 85 Response

```
HTTP/1.1 200 Created
Content-Type: text/csv

ReverificationSubscriberStatusReport.csv
```

Code Block 86 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
      "header":{
            "failureType":"Validation Error"
      },
      "body":[
            ["sac","The user is not authorized to perform transactions with this
SAC number.","USER_NOT_AUTHORIZED_FOR_SAC"]
      ]
}
```

6.18.2. ETC Failed Reverification De-Enroll Report

The ETC Failed Reverification De-Enroll Report query string is built in the URL of the GET request, and the format for the query string can be found in this following API specification.

Code Block 87 Request

```
GET /report/revdeenroll?sac=123456,234567
```

Code Block 88 Data Options

```
SAC: Any of the SACs the requesting party has authorization to see. Multiple SAC
selection is available.
```

Code Block 89 Response

```
HTTP/1.1 200 Created
Content-Type: text/csv
```

```
FailedReverificationDeEnrollReport.csv
```

Code Block 90 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
      "header":{
            "failureType":"Validation Error"
      },
      "body":[
            ["sac","The user is not authorized to perform transactions with this
SAC number.","USER_NOT_AUTHORIZED_FOR_SAC"]
      ]
}
```

6.18.3. ETC Recertification Subscriber Status Report

The ETC Recertification Subscriber Status Report query string is built in the URL of the GET request, and the format for the query string can be found in this following API specification.

Code Block 91 request

```
GET
/report/recertstatus?sac=123456&type=recertconfirmed&recertCheckStartDate=06/20/18
&recertCheckEndDate=06/20/2018
```

Code Block 92 Data Options

```
sac: Any of the SACs the requesting party has authorization to see.
type: recertconfirmed, recertnotconfirmed, recertall. Limited to a single type.
recertCheckStartDate: Date format: MM/DD/YYYY
recertCheckEndDate: Date format: MM/DD/YYYY
```

Code Block 93 Response

```
HTTP/1.1 200 Created
Content-Type: text/csv

RecertificationSubscriberStatusReport.csv
```

Code Block 94 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json
```

```
{


    "header": (
       "failureType":"Validation Error"
    },
    "body":[
       ["recertCheckStartDate","Date is in the incorrect format. The correct
format is: MM/DD/YYYY.","INVALID_DATE_FORMAT","6/20/18"]
    ]
}
```

6.18.4. ETC Failed Recertification De-Enroll Report

The ETC Failed Recertification De-Enroll Report query string is built in the URL of the GET request, and the format for the query string can be found in this following API specification.

Code Block 95 Request

```
GET /report/recertdeenroll?sac=123456&startDate=06/20/2018&endDate=06/20/2018
```

Code Block 96 Data Options

```
sac: Any of the SACs the requesting party has authorization to see.
startDate: Date format: MM/DD/YYYY
endDate: Date format: MM/DD/YYYY
```

Code Block 97 Response

```
HTTP/1.1 200 Created
Content-Type: text/csv

FailedRecertificationDeEnrollReport.csv
```

Code Block 98 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json


{
    "header": (
       "failureType":"Validation Error"
    },
    "body":[
       ["endDate","Date is in the incorrect format. The correct format is:
MM/DD/YYYY.","INVALID_DATE_FORMAT","6/20/18"]
    ]
}
```

6.19. Confirm Link Up

The Confirm Link Up API is an HTTP POST method that allows an ETC to confirm Link Up to determine whether a subscriber has already received a Link Up benefit a specific address. This method will search for a subscriber based on the subscriber's last name, date of birth, last four digits of their social security number or Tribal ID and their primary address. Date format must be: MM/DD/YYYY. If a social security number is not provided, a Tribal ID must be provided.

The ETC check subscriber query API accepts a JSON document in the body of the POST, which must include the following subscriber fields:

- lastName
- dob
- last4ssn or tribalId
- primaryAddress1
- primaryCity
- primaryState
- primaryZipCode

Code Block 99 Request

```
POST /linkup
Content-Type: application/json

{
    "lastName": "DOE",
    "last4ssn": "1234",
    "tribalId": "1234abcd",
    "dob": "04/21/1966",
    "primaryAddress1": "175 E 196TH ST",
    "primaryAddress2": "APT 1138",
    "primaryCity": "NEW YORK",
    "primaryState": "NY",
    "primaryZipCode": "10001",
    "primaryUrbanizationCode": ""
}
```

Code Block 100 Response

```
HTTP/1.1 200 OK
Content-Type: application/json
[{
     "message": "No Tribal Link Up information found. Link Up benefit may be
provided."
}]
```

Code Block 101 Response Info

```
Possible success messages:
"No Tribal Link Up information found. Link Up benefit may be provided."
"Tribal Link Up benefit existing. Link Up Service Date is xx/xx/xxxx"
```

Code Block 102 Error Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
       "header": {
              "failureType": "Malformed Document"
       },
       "body": [
       ["dob","Subscriber's date of birth is required.","DOB_REQUIRED"],
       ["last4ssn","Must provide the last four digits of the subscriber's social
security number or the subscriber's Tribal ID.","EMPTY_L4DSSN_AND_TRIBALID"],
       ["lastName","Subscriber's last name is required.","LASTNAME_REQUIRED"]
       ]
}
```

Error Response Info

Failure types:

- Malformed Document: Format or structure of the batch file is incorrect.
- Validation Error: A field or fields has failed one or more of the validations. Validation errors can include Address Matching Service errors and Third Party Identification Verification errors.

7. APPENDIX A

Field Descriptions Table – The following table lists all fields used by NLAD public APIs, including the field names, descriptions, types, formats, input limitations, and requirement and conditional requirement states.

Field Name	Field Description	Data Description	Required Status	Conditional Requirements	Type	Length	Format	Default Value
nv	National Verifier workflow	ETCs may use this flag for SACs in National Verifier states during the soft launch period to verify subscribers through the National Verifier.	Optional	Optional during Soft Launch for National Verifier states. Required during hard launch.	Bit	1	0 = no / 1 = yes	default = 0
`transactionType`	Transaction Type	There are a number of transactions you can perform with NLAD. Every transaction must have its type declared in this field. For batch transactions, each row of the batch file is a single transaction, meaning each row must have the transaction type entered. Full list of available transactions: `verify` – Used to verify a Lifeline eligible subscriber. `enroll` – Used to enroll a Lifeline eligible subscriber. `transfer` – Used to transfer the Lifeline benefits from another carrier. `update` – Used to update/change a subscriber's information.	Required		Alphabetic			

Field Name	Field Description	Data Description	Required Status	Conditional Requirements	Type	Length	Format	Default Value
		`deEnrollDeceased` – Used to de-enroll a Lifeline subscriber who has deceased. `deEnrollLeaving` – Used to de-enroll a Lifeline subscriber who is opting out of the program, or is no longer eligible for benefits. `deEnrollFailedRecertification` – Used to de-enroll a Lifeline subscriber who has not filed their annual recertification. `deEnrollNonUsage` – Used to de-enroll a Lifeline subscriber who has not used their benefits for 60 days.						
`phoneNumberInNlad`	Subscriber's Current Telephone Number	Subscriber's current phone number in NLAD, associated with their Lifeline benefits. This field is required when de-enrolling a subscriber. This may be the same number you enter in the phoneNumber field. When performing an update transaction that updates the phone number, put the current phone number in this field and the new phone number in the phoneNumber field.	Conditional	Required if `transactionType` = `update` `deEnrollDeceased` `deEnrollLeaving` `deEnrollFailedRecertification` `deEnrollNonUsage`	Numeric	10	xxxxxxxxxx	
`transactionEffectiveDate`	Transaction Effective Date	The transaction effective date is the date the transaction is effective with the ETC. For example, if you sign up a subscriber on 12/**10**/2013, but submit the enroll	Required		Date		mm/dd/yyyy	

Field Name	Field Description	Data Description	Required Status	Conditional Requirements	Type	Length	Format	Default Value
		transaction for that subscriber on 12/**11**/2013, the transaction *effective* date is 12/**10**/2013, which is what should be entered in this field.						
sac	Study Area Code	This is the 6-digit number associated with the ETC providing the Lifeline benefit to the subscriber. Every transaction must include the appropriate SAC number for that subscriber.	Required		Numeric	6	xxxxxx	
lastName	Last Name	Full, last name of subscriber. Minimum of two alphabetic characters. Accepts the SPACE character, and these special characters: **Name** / **Character** Apostrophe / ' Accent Grave / ` Dash / -	Required		Alphabetic	50		
firstName	First Name	First name of subscriber. Accepts the SPACE character, and these special characters: **Name** / **Character** Apostrophe / ' Accent Grave / `	Required		Alphabetic	50		

Field Name	Field Description	Data Description	Required Status	Conditional Requirements	Type	Length	Format	Default Value
		Dash –						
middleName	Middle Name	Middle name of subscriber. Accepts the SPACE character, and these special characters: **Name** / **Character** Apostrophe ' Accent Grave ` Dash – Period .	Optional		Alphabetic	50		
phoneNumber	Telephone Number	Telephone number of the Lifeline-eligible subscriber. This can be the new number you are assigning to the subscriber. If the subscriber is already enrolled in NLAD, you can put their current phone number here, in which case it will be the same as phoneNumberInNlad. No two subscribers can have the same phone number.	Required		Numeric	10	xxxxxxxxxx	
last4ssn	Last Four Digits of Social Security Number	Last four digits of the subscriber's social security number.	Conditional	last4ssn or tribalId must be provided.	Numeric	4	xxxx	

Field Name	Field Description	Data Description	Required Status	Conditional Requirements	Type	Length	Format	Default Value
tribalId	Tribal Identification Number	The Tribal identification number, or Tribal enrollment number of the subscriber. ETCs that have collected partial Tribal IDs may submit them so long as they have at least two characters. This field accepts some special characters, including (but not limited to) the dash character (-).	Conditional	last4ssn or tribalId must be provided.	Alphanumeric	20		
dob	Date of Birth	Subscriber's date of birth.	Required		Date		mm/dd/yyyy	
serviceType	Service Type	The service provided to the subscriber that meets the minimum service requirements.	Required		Alphabetic	21	Voice, Broadband, BundledVoice, BundledBroadband or BundledVoiceBroadband	
iehFlag	Independent Economic Household Flag	The Independent Economic Household (IEH) Flag IEH flag indicates the subscriber is an independent economic entity sharing an address with another Lifeline subscriber. ETCs must collect and retain the IEH worksheet.	Optional		Bit	1	0 = no / 1 = yes	default = 0
iehCertificationDate	IEH Date of Certification	Enter the date IEH certification was performed for the subscriber.	Conditional	required if iehFlag = 1	Date		mm/dd/yyyy	

Field Name	Field Description	Data Description	Required Status	Conditional Requirements	Type	Length	Format	Default Value
iehRecertificationDate	IEH Date of Recertification	Enter the date IEH certification for the subscriber was recertified.	Optional		Date		mm/dd/yyyy	
primaryAddress1	Street Address	Subscriber's street address.	Required		Alphanumeric	50		
primaryAddress2	Secondary Address	Subscriber's secondary address.	Optional		Alphanumeric	50		
primaryCity	City	Subscriber's city of residence.	Required		Alphabetic	50		
primaryState	State	Subscriber's state of residence.	Required		Alphabetic	2		
primaryZipCode	ZIP	Subscriber's ZIP code of residence. Accepts a dash character (–).	Required		Numeric	10	xxxxx or xxxxx-xxxx	
primaryUrbanizationCode	Urbanization Code	This field is used only for Puerto Rico addresses that have an urbanization code.	Optional		Alphabetic	50		
primaryPermanentAddressFlag	Temporary Address Flag	The temporary address flag indicates that the primary address entered is a *temporary* location. A value of "0" indicates that the address is a *permanent* location. **Note: the field "primaryPermanentAddressFlag" is the Temporary Address Flag field.**	Optional		Bit	1	0 = permanent / 1 = temporary	default = 0

Field Name	Field Description	Data Description	Required Status	Conditional Requirements	Type	Length	Format	Default Value
primaryTribalFlag	Tribal Address Flag	The primary Tribal flag indicates that the subscriber's address is in Tribal lands and is not registered with USPS address matching service (AMS).	Optional		Bit	1	0 = no / 1 = yes	
primaryRuralFlag	Non-Deliverable Rural Address Flag	The primary rural flag indicates that the subscriber's primary address is in a rural area, and is not registered with AMS, nor able to receive postal delivery.	Optional		Bit	1	0 = no / 1 = yes	default = 0
mailingAddress1	Mailing Street Address	Subscriber's mailing street address.	Optional		Alphanumeric	50		
mailingAddress2	Mailing Secondary Address	Subscriber's secondary mailing address.	Optional		Alphanumeric	50		
mailingCity	Mailing City	Subscriber's mailing city.	Optional		Alphabetic	50		
mailingState	Mailing State	Subscriber's mailing state.	Optional		Alphabetic	2		
mailingZipCode	Mailing ZIP	Subscriber's mailing ZIP code. Accepts a dash character (-).	Optional		Numeric	10	xxxxx or xxxxx-xxxx	
mailingUrbanizationCode	Mailing Urbanization Code	This field is used only for Puerto Rico addresses that have an urbanization code.	Optional		Alphabetic	50		

Field Name	Field Description	Data Description	Required Status	Conditional Requirements	Type	Length	Format	Default Value
serviceInitializationDate	Service Initiation Date	Date that the service provider determined that the subscriber was eligible for Lifeline service. *If an ETC cannot provide the above date for a customer who was a subscriber prior to June 1, 2012, then the service provider should enter the following date for that subscriber:* May 31, 2012	Required		Date		mm/dd/yyyy	
serviceReverificationDate	Date of Reverification	This is the date the subscriber's Lifeline eligibility was reverified.	Optional		Date		mm/dd/yyyy	
eligibilityCode	Eligibility Program	The program code under which the subscriber is eligible for Lifeline benefits. Acceptable values are: E1 E2 E3 E4 E8 E9 E10 E11 E13 E14 E15 As of 12/1/16, the following eligibility codes can no longer be used, except when	Required		Alphanumeric	3		

Field Name	Field Description	Data Description	Required Status	Conditional Requirements	Type	Length	Format	Default Value
		using the Update Subscriber service to update fields other than eligibilityCode: E5 E6 E7 E12 More information about program codes can be found here: http://usac.org/_res/documents/li/pdf/nlad/Handout_Enrollment-Eligibility-Codes.pdf						
bqpLastName	BQP Last Name	Last name of the benefit-qualifying person (BQP).	Conditional	Required if any BQP field is provided.	Alphabetic	50		
bqpFirstName	BQP First Name	First name of the BQP.	Conditional	Required if any BQP field is provided.	Alphabetic	50		
bqpMiddleName	BQP Middle Name	Middle name of the BQP.	Optional		Alphabetic	50		
bqpDob	BQP Date of Birth	Date of birth of the BQP.	Conditional	Required if any BQP field is provided.	Date		mm/dd/yyyy	
bqpLast4ssn	BQP Last Four Digits of Social Security Number	Last four digits of the BQP's social security number.	Conditional	Required if any BQP field is provided. BQP last4ssn or tribalId must be provided.	Numeric	4	xxxx	

Field Name	Field Description	Data Description	Required Status	Conditional Requirements	Type	Length	Format	Default Value
bqpTribalId	BQP Tribal Identification Number	Tribal identification number of the BQP.	Conditional	Required if any BQP field is provided. BQP last4ssn or tribalId must be provided.	Alphanumeric	20		
linkUpServiceDate	Link Up Date of service	The date Link Up service started.	Optional		Date		mm/dd/yyyy	
lifelineTribalBenefitFlag	Lifeline Tribal Benefit Flag	ETCs may use this flag to claim Lifeline Tribal support for a qualified subscriber to whom the ETC is offering Tribal rates. Note: this field is not related to primaryTribalFlag.	Required		Bit	1	0 = no / 1 = yes	default = 0
acpFlag	Address Confidentiality Program Flag	This field is inactive; any values entered into this field will be changed to null.	Optional		Bit	1		
etcGeneralUse	ETC General Use	This field is for general ETC use. An ETC may populate this field with any value, and it will be returned along with transaction error messages. For example, an ETC could enter a unique identifier in this field that will allow them to automate the process of looking up a subscriber in their own database when a transaction fails.	Optional		Alphanumeric	50		

Field Name	Field Description	Data Description	Required Status	Conditional Requirements	Type	Length	Format	Default Value
		This field accepts alphanumeric characters, the SPACE character, and these special characters: **Name** / **Character** Dash / - Underscore / _ Colon / : Pound / # At Sign / @ Period / .						
tpivFlag	TPIV Flag	This field is inactive; any values entered into this field will be changed to "0". This field is still required as a column in the heading row of a batch file.	Optional		Alphanumeric	3	xxx	default = 0
resolutionId	Resolution ID	The unique identifier of a failed transaction.	Conditional	Required if not in a National Verifier state.	Alphanumeric	12	xxxxxx-xxxxxx	
agentName	Agent Name	Name of the agent that reviewed the subscriber's identity information, address, or age (for minors).	Conditional	Required if agentId field = null	Alphabetic	50		
agentId	Agent ID	Employee ID of the agent that reviewed the subscriber's identity information, address, or age (for minors).	Conditional	Required if agentName field = null	Alphanumeric	30		

Field Name	Field Description	Data Description	Required Status	Conditional Requirements	Type	Length	Format	Default Value
fullNameLast4ssnTCode	Full Name Last 4 digits of SSN Error Resolution Code	Resolution error codes are required to complete a dispute resolution submission. These codes reference specific NLAD failures relating to third party identity verification (TPIV) errors. Please refer to the Dispute Resolution page on the USAC.org website for more information on resolution codes.	Conditional	Required if transaction error = TPIV_FAIL_NAME_SSN4	Alphanumeric		T3, T4, T5, T8, T12, T13, T14, T15, T16, T17	
fullNameDobTCode	Full Name Date of Birth Error Resolution Code	Resolution error codes are required to complete a dispute resolution submission. These codes reference specific NLAD failures relating to third party identity verification (TPIV) errors. Please refer to the Dispute Resolution page on the USAC.org website for more information on resolution codes.	Conditional	Required if transaction error = TPIV_FAIL_DOB	Alphanumeric		T1, T2, T6, T7, T8, T10, T12, T13, T14, T15, T16, T17	
fullNameDeceasedTCode	Full Name Deceased Error Resolution Code	Resolution error codes are required to complete a dispute resolution submission. These codes reference specific NLAD failures relating to third party identity verification (TPIV) errors. Please refer to the Dispute Resolution page on the USAC.org website for more information on resolution codes.	Conditional	Required if transaction error = TPIV_FAIL_DECEASED	Alphanumeric		T14, T16, T18, T19, T20, T21, T22	
aCode	Address Error Resolution Code	Resolution error codes are required to complete a dispute resolution submission. These codes reference specific NLAD failures relating to address errors. Please refer to the Dispute Resolution page on the	Conditional	Required if transaction error = Invalid_Address	Alphanumeric		A1 - A13	

Field Name	Field Description	Data Description	Required Status	Conditional Requirements	Type	Length	Format	Default Value
		USAC.org website for more information on resolution codes.						
mCode	Subscriber Under 18 Error Resolution Code	Resolution error codes are required to complete a dispute resolution submission. These codes reference specific NLAD failures relating to subscriber's age errors. Please refer to the Dispute Resolution page on the USAC.org website for more information on resolution codes.	Conditional	Required if transaction error = Subscriber_Under_18	Alphanumeric		M1	
desc	Description	This field is for ETC use if extra information is needed to help resolve a request. An ETC may populate this field with any value.	Optional for Post /resolution/ submit Required for Post /resolution		Alphanumeric	1000		
certificationFlag	Certification Flag	This flag indicates that the submission of data to the National Lifeline Accountability Database is accurate to the best of your knowledge and any willful false statements made may subject you to federal criminal prosecution and penalties, as well as civil penalties.	Required		Alphabetic	1	y = yes / n = no	
subscriberId	Subscriber ID	This field is an automatically generated unique identification value for each subscriber in NLAD.	Conditional	Phone number or subscriber ID must be provided on update and de-enroll transactions.	Alphanumeric	9	xxxxxxxxx	

Field Name	Field Description	Data Description	Required Status	Conditional Requirements	Type	Length	Format	Default Value
includeSubscriberId	Include Subscriber ID	This field is an optional field to be included on Enroll transactions and Benefit Transfer transactions if the ETC wants the subscriber ID to be returned on a successful transaction.	Optional		Bit	1	0 = no/ 1 = yes	default = 0

8. APPENDIX B

The following table is a list of success and error messages that could be returned through the API. Developers are advised to develop against the message code (MSGCD) parameter, which is formatted to facilitate comparison. The message description is subject to change, and would not make a suitable parameter.

MSGCD	Description	RETURN CODE
A_AMS_OFFLINE	Address matching service is currently offline. Please try again later.	400
A_AMS_OFFLINE_BAU	Address matching service is currently offline. Please try again later.	400
ADDRESS_FOUND	Matching address found.	200
ADDRESS_FOUND_AND_SUBSCRIBER_NOT_FOUND	Address found and subscriber not found	200
AMS_FAILURE_ANALYSIS	[The content of this message is a collection of specific AMS failure messages, all of which can be found at the bottom of this table in the AMS MESSAGES section.]	400
APPLICATION_NOT_FOUND	The subscriber has not qualified through the Lifeline National Verifier yet or their application has expired. You can qualify them now at checklifeline.org	400
APPLICATION_NOT_COMPLETE	The subscriber has not finished qualifying through the Lifeline National Verifier. Submit documents for the below listed error(s) at checklifeline.org	400
APPLICATION_PENDING	The subscriber's application is currently under review. For more information please contact customer support at 1(877) 524-1325.	400
AUTH_REQUIRED	User authentication required.	401
CANNOT_ENTER_PHONE_NUMBER_AND_SUBSCRIBER_ID	Cannot enter both fields for this transaction. Must provide the subscriber's phone number in NLAD or the subscriber ID.	400
CANNOT_TRANSFER_SAME_SAC	Cannot transfer a subscriber to the same SAC number.	400
CANNOT_UPDATE_ADDRESS_FIELDS	Cannot update primary address fields during reverification.	400

CANNOT_UPDATE_ADDRESS_FLAGS	Cannot update address flags during reverification.	400
CANNOT_UPDATE_BQP_IDENTITY_FIELDS	Cannot update BQP identity fields.	400
CANNOT_UPDATE_IDENTITY_FIELDS	Cannot update identity fields.	400
CANNOT_UPDATE_SAC	The SAC number cannot be changed during an update. Please use the transfer service to transfer subscribers between SACs.	400
CANNOT_UPDATE_SVCINITDATE	Service initialization date cannot be updated. The service initialization date must be mm/dd/yyyy.	400
DEENROLL_ERROR	An unspecified error occurred while attempting to de-enroll the subscriber.	400
DESCRIPTION_SIZE	Description cannot exceed 1,000 characters.	400
DETAIL_TRANSACTION_TOO_MANY_TYPES	Only one transaction type can be selected for a detail-transaction report.	400
DOB_IN_FUTURE	Date of birth cannot be in the future.	400
DOB_REQUIRED	Subscriber's date of birth is required.	400
DUPLICATE_ADDRESS	The primary address in this transaction matches the primary address of another subscriber.	400
DUPLICATE_ADDRESS_NLAD	The subscriber in this transaction has a duplicate address of another subscriber. Please contact customer support at 1(877) 524-1325.	400
DUPLICATE_FILE	This file has already been processed.	400
DUPLICATE_PHONE_NUMBER	The phone number in this transaction matches the phone number of another subscriber.	400
DUPLICATE_PRIMARY_ADDRESS	The primary address in this transaction matches the primary address of another subscriber.	400
DUPLICATE_SUBSCRIBER	The subscriber in this transaction is a duplicate of another subscriber.	400
DUPLICATE_SUBSCRIBER_NLAD	The subscriber in this transaction is a duplicate of another subscriber. Please contact customer support at 1(877) 524-1325.	400
DUPLICATE_SUBSCRIBER_AND_PHONE_NUMBER	This transaction contains a duplicate subscriber and a duplicate phone number.	400

DUPLICATE_SUBSCRIBER_EXTERNAL	The subscriber in this transaction is a duplicate of another subscriber in the state's internal lifeline database.	400
DUPLICATE_SUBSCRIBER_IN_PROCESS	This subscriber is currently undergoing duplicate resolution processing.	400
EMPTY_INPUT	Required fields missing.	400
EMPTY_L4DSSN_AND_TRIBALID	Must provide the last four digits of the subscriber's social security number or the subscriber's Tribal ID.	400
EMPTY_RESOLUTION_ID	Resolution ID is required.	400
EMPTY_SAC	SAC number is required.	400
EMPTY_STATE	State is required.	400
END_DATE_BEFORE_START_DATE	The SAC end date cannot occur before the SAC start date.	400
ERRORS	[Batch Status Code] Batch processed with errors. Check the rejected-row file for more information.	
FAILED_TPIV	Subscriber failed third-party identity verification.	400
FILE_ALREADY_SUBMITTED	This file has already been processed.	400
FILE_CREATION_ERROR	An unspecified error occurred while attempting to create the report file. Please try again later.	400
FILE_DOES_NOT_EXIST	The rejected-rows file could not be found.	404
FILE_EMPTY	File contains no transactions.	400
FILE_NAME_LENGTH_EXCEED_LIMIT	The filename cannot exceed 50 characters.	400
FILE_NAME_LENGTH_TOO_SHORT	The SAC number in the filename must be exactly six-characters long.	400
FILE_NOT_CSV	File is not in the CSV format.	400
FILE_NOT_FOUND	File not found.	400
FILE_NOT_IN_CSV_FORMAT	File is not in the CSV format.	500

FILE_READ_ERROR	Cannot read file.	400
FILE_SIZE_EXCEED_LIMIT	File cannot be larger than three megabites.	400
FILE_SIZE_ZERO	File size must be greater than 0 bytes.	400
FILE_SUBMITTED	File submitted.	200
FILENAME_EMPTY	File must have a name.	400
FIRST_NAME_REQUIRED	First name is required.	400
GATEWAY_ERROR	Unspecified gateway error occurred. Please try again later.	504
HEADERS_INVALID	File contains invalid headers.	400
IN_PROGRESS	[Batch Status Code] Batch is currently being processed.	
INCOMPLETE_BQP_INFORMATION	If any BQP information is provided, then all BQP information is required, except for BQP Middle Name.	400
INCOMPLETE_MAILING_ADDRESS	If any data is provided for any fields in the mailing address, then all mailing address fields are required, except for secondary mailing address.	400
INCORRECT_REPORT_FORMAT	Report format must be JSON or CSV.	400
INITSTARTDATE_BEFORE_INITENDATE	Initialization start date cannot occur after initialization end date.	400
INVALID_ADDRESS	Address unrecognized (failed Address Matching Service).	400
INVALID_ADDRESS_LINE	Invalid address line.	400
INVALID_AGENT_NAME	Invalid characters used in the Agent name field, or Agent name field length is incorrect.	400
INVALID_CERTIFICATION_FLAG	The Certification flag field should be set to Y, to certify that my submission of data to the National Lifeline Accountability Database is accurate to the best of my knowledge and any willful false statements made may subject me to federal criminal prosecution and penalties, as well as civil penalties.	400
INVALID_CITY_STATE_ZIP	Invalid city/state/zip code combination.	400
INVALID_DATE_FORMAT	Date is in the incorrect format. The correct format is: MM/DD/YYYY.	400

INVALID_DATE_RANGE	Date range cannot be more than 2 years from current date.	400
INVALID_DEENROLL_TRANSACTION_TYPE	Invalid de-enroll code.	400
INVALID_DOB_DATE_FORMAT	Date of birth format is incorrect. The correct format is: MM/DD/YYYY.	400
INVALID_DOB_VALUE	Date of birth contains an invalid value.	400
INVALID_DUPLICATE_TYPE	Invalid Duplicate Type. The only acceptable values for duplicate type are " DUPLICATE_SUBSCRIBER", " MIGRATION_DUPLICATE_SUBSCRIBER", or " PRODUCTION_DUPLICATE_SUBSCRIBER".	400
INVALID_ELIGIBILITY_CODE	Subscriber eligibility program (eligibilityCode) can not have any value other than one of the following: E1, E2, E3, E4, E5, E6, E7, E8, E9, E10, E11, E12, E13, E14, and E15. As of 12/1/16, the following eligibility codes can no longer be used, except when using the Update Subscriber service to update fields other than eligibilityCode: E5 E6 E7 E12.	400
INVALID_ETC_GENERAL	ETC general use field contains invalid values.	400
INVALID_ETC_SUBSCRIBER_REPORT_TYPE	Report type must be "summary" or "detail".	400
INVALID_EXCEPTION_CODE	The exception code provided does not meet the criteria for the transaction failure type. Please refer to the Dispute Resolution page on the USAC.org website	400
INVALID_FIELD_LENGTH	Incorrect field length.	400
INVALID_FILE	File is invalid	400
INVALID_FILENAME_CHARS	The filename contains a SPACE or one or more of the following invalid characters: \ / : * \" < > |	400
INVALID_FILENAME_LENGTH	The filename can not exceed 50 characters.	400
INVALID_FLAG_FORMAT	This flag field can accept only a value of "0" or "1".	400
INVALID_FUTURE_DATE	Date cannot occur in the future.	400
INVALID_IEH_CERTIFICATION_FLAG	If the subscriber's IEH date of certification is provided, then the subscriber's IEH status must be set to "Yes."	400
INVALID_IEH_RECERTIFICATION_DATE	IEH recertification date cannot be dated before IEH certification date.	400

INVALID_INIT_TRANSACTION_TYPE	During initialization, the only permitted transaction type is "enroll". Any other value in the field "transactionType" will cause the transaction to fail.	400
INVALID_L4DSSN_FORMAT	SSN format incorrect or Invalid character length used.	400
INVALID_LASTNAME	Invalid characters used in the name fields, or name field lengths are incorrect.	400
INVALID_LIFELINE_TRIBAL_BENEFIT_FLAG	Lifeline Tribal Benefit is unavailable for this location.	400
INVALID_LINKUP_REQUEST_SAC	This SAC number is not authorized to submit Link Up requests. Link Up Service date cannot have a value with this SAC number at this time.	400
INVALID_LINKUP_REQUEST_TRIBALBENEFIT	If Link Up Service date is provided, then Lifeline Tribal Benefit must be set to "yes."	400
INVALID_NAME	Invalid characters used in the name fields, or name field lengths are incorrect.	400
INVALID_NUMBER_OF_FIELDS	Incorrect number of fields for this transaction.	400
INVALID_PHASECD	This SAC number is not authorized to submit transactions at this time.	400
INVALID_PHONE_NUMBER	The phone number provided is not a valid phone number.	400
INVALID_PRIMARY_TRIBAL_FLAG	Tribal Address is unavailable for this location.	400
INVALID_RESOLUTION	Resolution ID required for override.	400
INVALID_RESOLUTION_CODE	The resolution codes provided do not meet the criteria for the transaction failure types. Please refer to the Dispute Resolution page on the USAC.org website.	400
INVALID_RESOLUTION_ID	The resolution ID has invalid characters, length, or format.	400
INVALID_RESOLUTION_REPORT_TYPE	Invalid resolution ID status. The only acceptable values for resolution ID status are "OPEN", "IN_PROGRESS", or "CLOSED".	400
INVALID_RESOLUTION_REQUEST	Cannot submit Port Freeze Exception Codes.	400
INVALID_SAC	The SAC number for this transaction differs from the SAC number in the filename.	400
INVALID_SERVICE_TYPE	The service type cannot have any value other than one of the following: Voice, Broadband, BundledVoice, BundledBroadband, BundledVoiceBroadband.	400
INVALID_SESSIONID	There was an error with your session token. Log out then log back in and try again.	400

INVALID_SNAP_PERIOD_FORMAT	SnapPeriod is in the incorrect format. The correct format is: MM/YYYY.	400
INVALID_SNAPSHOT_PERIOD	Data does not exist for the snapshot period.	400
INVALID_SSN	The social security number has invalid characters, length, or format.	400
INVALID_STATE	Unrecognized state code.	400
INVALID_SUBSCRIBER_ID	The subscriber ID has invalid characters, length, or format.	400
INVALID_SVCINITDATE	The service initialization date cannot occur before 12/10/1985.	400
INVALID_TRANSACTION_REPORT_TYPE	Invalid transaction type.	400
INVALID_TRIBAL_ID	Tribal ID has invalid characters, length, or format.	400
INVALID_TRIBALID_FORMAT	The Tribal ID has invalid characters, length, or format.	400
INVALID_UPLOAD_DATE	This SAC number is not authorized to submit transactions at this time.	400
INVALID_ZIP_CODE	Zip code has invalid characters, length, or format.	400
LASTNAME_REQUIRED	Subscriber's last name is required.	400
LINKUP_DATE_FOUND	Tribal Link Up benefit existing. Link Up Service Date is xx/xx/xxxx	200
MALFORMED_DOCUMENT	Malformed Document	400
MAX_SACS_LIMIT_REACHED	Maximum number of SACs should not exceed 250.	400
message	Subscriber successfully updated	200
message	Subscriber successfully enrolled	201
message	Subscriber de-enrolled.	200
message	Subscriber successfully updated	200
message	Subscriber successfully transferred	200
message	Subscriber and address not found	200
message	Address found and subscriber not found	200

message	Subscriber and matching address found.	200
message	Found Subscriber	200
message	A case has been successfully created with NLAD Customer Service for this resolution ID.	200
message	Subscriber passed all validations and verifications	200
METHOD_NOT_ALLOWED	The HTTP request method is not valid for this API.	405
MISSING_IEH_CERTIFICATION_DATE	If the IEH flag is set to "yes," then the IEH Certification Date is required.	400
MISSING_PHONE_NUMBER_OR_SUBSCRIBER_ID	Must provide the subscriber's phone number in NLAD or the subscriber ID.	400
MISSING_REQUIRED_FIELD	Missing required field.	400
MISSING_SSN_OR_TRIBAL	Must provide the last four digits of the subscriber's social security number or the subscriber's Tribal ID.	400
MULTIPLE_ACP_RURAL_TRIBAL	No transaction can have more than one of the following statutes set to "yes": Tribal Address (primaryTribalFlag), Non-Deliverable Rural Address (primaryRuralFlag).	400
MULTIPLE_FILES_ERROR	This service only accepts one file at a time.	400
MULTIPLE_SUBSCRIBERS	This request cannot be processed. Please e-mail NLAD Support at NLADsupport@usac.org	400
MULTIPLE_SUBSCRIBERS_FOUND	Multiple subscribers found matching that identity information, please use the subscriber ID.	400
NO_LINKUP_DATE	No Tribal Link Up information found. Link Up benefit may be provided.	200
NO_RESOLUTION_ERRORS	No failed transactions could be found associated with the submitted resolution ID.	400
NON_EXISTING_SUBSCRIBER	Subscriber not found.	400
NOT_AUTHORIZED_FOR_LEGACY	This transaction must be completed using National Verifier mode.	400
NOT_AUTHORIZED_TO_ACCESS_RESOURCE	You are not authorized to access this resource.	403
NOT_WITHIN_DATE_RANGE	This request cannot be processed. Please e-mail NLAD Support at NLADsupport@usac.org	400

NULL_DATE	This SAC number is not currently in initialization.	400
NV_UNAVAILABLE	The National Verifier service is currently unavailable. Please try again later.	500
PHONE_NUMBER_IN_NLAD_IS_EMPTY	The Phone Number in NLAD field is required.	400
PLEASE_RESUBMIT	[Batch Status Code] The file needs to be resubmitted.	
PROCESSING_ERROR	[Batch Failed Reason Code] There was an unspecified error processing the file.	
REJECTED	[Batch Status Code] The file was rejected.	
RESOLUTION_ALREADY_SUBMITTED	This resolution ID has already been submitted, and cannot be submitted again.	400
RESOLUTION_CREATED	The resolution request was successfully submitted.	201
RESOLUTION_MISSING_DESCRIPTION	The resolution request description is missing.	400
RESOLUTION_MISSING_ERRORS	No failed transaction could be found associated with this resolution ID.	400
RESOLUTION_NOT_FOUND	Resolution ID not found.	400
RESOLUTION_PROCESSING_ERROR	An unspecified error occurred while attempting to processes the resolution request. Please try again later.	400
RESOLUTION_STATUS_UPDATE_SUCCESS	The resolution status was successfully updated.	200
RESOLUTION_SUBMITTED	Resolution request successfully submitted.	200
RESOLUTION_SUBSCRIBER_ALREADY_SUBMITTED	A resolution request resolutionId: xxxxxx-xxxxxx has already been submitted for this subscriber, and cannot be submitted again. Please refer to the 'Detail Resolution Status Report' for the subscriber's resolution status.	400
SAC_ALREADY_IN_PROGRESS	A batch file for this SAC number is already in progress. Please wait until the current batch is finished before submitting another batch file.	400
SAC_BATCH_IN_PROGRESS	A batch file for this SAC number is already in progress. Please wait until the current batch is finished before submitting another batch file.	400
SAC_DOES_NOT_EXIST	The SAC number could not be found.	400
SAC_NOT_AUTHORIZED_FOR_NV	This SAC is not authorized for use in a National Verifier transaction.	400

SAC_NOT_FOUND	The SAC number could not be found.	200
SAC_NOT_SIX_DIGITS	SAC number is not 6 digits.	400
SERVICE_UNAVAILABLE	Service unavailable. Please try again later.	503
SSN_AND_TRIBAL	Provide either the last 4 digits of the subscriber's SSN or the Tribal ID, do not provide both.	400
SSN_AND_TRIBAL_BQP	Provide either the last 4 digits of the benefit qualifying person's SSN or the Tribal ID, do not provide both.	400
STATE_FED_FAIL	Subscriber program eligibility could not be found.	400
STATE_WEBSERVICE_ERROR	State verification services are currently unavailable. Please try again later.	503
SUBSCRIBER_ADDRESS_CANNOT_BE_PO_BOX	The subscriber's primary address cannot be a PO Box.	400
SUBSCRIBER_ADDRESS_INFO_IS_EMPTY	Must submit all primary address fields.	400
SUBSCRIBER_AND_ADDRESS_FOUND	Subscriber and matching address found.	200
SUBSCRIBER_AND_ADDRESS_NOT_FOUND	Subscriber and address not found	404
SUBSCRIBER_BQP_MATCH	The subscriber and benefit qualifying person cannot be the same.	400
SUBSCRIBER_DEENROLLED	Subscriber de-enrolled.	200
SUBSCRIBER_FOUND	Subscriber found.	200
SUBSCRIBER_FOUND_AND_ADDRESS_NOT_FOUND	Subscriber found and address not found	200
SUBSCRIBER_MAILING_ADDRESS_IS_EMPTY	The subscriber mailing address cannot be empty when the mailing City, State, and ZIP is provided.	400
SUBSCRIBER_MAILING_CITYSTATEZIP_IS_EMPTY	The subscribers mailing City, State, and ZIP cannot be empty when the mailing address is provided.	400
SUBSCRIBER_NOT_FOUND	Subscriber not found.	404
SUBSCRIBER_NOT_FOUND_ERROR	Subscriber Not Found	400

SUBSCRIBER_OVER_130	No subscriber can be more than 130 years of age.	400
SUBSCRIBER_UNDER_18	No subscriber can be less than 18 years of age.	400
SUCCESS	[Batch Status Code] Batch was processed without errors.	202
SVCINITDATE_DOES_NOT_MATCH	The service initialization date must be mm/dd/yyyy.	400
SYSTEM_ERROR	[Batch Failed Reason Code] There was an unspecified error.	
TOO_FEW_HEADERS	[Batch Failed Reason Code] File was rejected for having too few headers.	500
TOO_MANY_HEADERS	[Batch Failed Reason Code] File was rejected for having too many headers.	500
TOO_MANY_SACS	Detail reports can only accepts a single SAC number.	400
TPIV_FAIL	Subscriber failed third-party identity verification.	400
TPIV_FAIL_NAME_SSN4	Subscriber name or SSN4 could not be validated.	400
TPIV_FAIL_DECEASED	Subscriber is identified as deceased.	400
TPIV_FAIL_DOB	Subscriber date of birth could not be validated.	400
TPIV_FAIL_IDENTITY_NOT_FOUND	Subscriber identity could not be found.	400
TPIV_WEBSERVICE_ERROR	Third-party identity verification services are currently unavailable. Please try again later.	400
TRANSACTION_EFFECTIVE_DATE_BEFORE_SVCINITDATE	Transaction effective date cannot occur before service initiation date	400
TRANSACTION_EFFECTIVE_DATE_IN_FUTURE	Transaction effective date cannot occur in the future.	400
TRANSACTION_EFFECTIVE_DATE_IS_EMPTY	Transaction effective date is required.	400
TRANSACTION_TYPE_IS_EMPTY	Transaction type is required.	400
TRANSACTION_TYPE_NOT_ENROLL	Transaction type is not "enroll".	400
TRANSACTION_TYPE_NOT_TRANSFER	Transaction type is not "transfer".	400
TRANSACTION_TYPE_NOT_UPDATE	Transaction type is not "update".	400
UNAVAILABLE_ELIGIBILITY_CODE	Invalid Eligibility Code for the selected state/territory.	400

UNSUPPORTED_TYPE_FORMAT	Wrong character type for this field.	400
USER_NOT_AUTHORIZED_FOR_SAC	The user is not authorized to perform transactions with this SAC number.	400
USER_NOT_AUTHORIZED_TO_DEENROLL	User is not authorized to de-enroll this subscriber.	400
USER_NOT_AUTHORIZED_TO_ENROLL	User is not authorized to enroll this subscriber.	400
ZIP_CODE_NOT_MATCHED	Zip code does not match.	400