Veratad Technologies LLC

Integration Document

Age & Identity Verification Solutions

Veratad Web Services

Output version: 4.3b
Output Date: April 2011

500 Frank W Burr Blvd, Center West – Suite 14

Teaneck, New Jersey 07666

201-510-6000

www.veratad.com

TABLE OF CONTENTS

Preface

The following style conventions are used throughout this document:

STYLE CONVENTIONS

- `constant width`: Text presented in constant width format indicates an exact input requirement as well as an exact output specification.
- **bold:** Text presented in bold indicates a mandatory input requirement or constant output specification.
- XML paths: In some cases it is helpful to refer to an area of XML using a path. Any paths in this document will be paths that start at the root node. "{a,b}" syntax is used to specify either of node a or node b.

```
<root>
   <main>
      <part-a>value-a</part-a>
      <part-b>value-b</part-b>
   </main>
</root>
```

For example /root/main/{part-a, part-b}, would indicate that what the text is referring describes will apply equally to /root/main/part-a and /root/main/part-b.

EXAMPLE DATA

Data used to create the examples is totally fictitious, and is provided to assist in the implementation process. Your initial implementation account will be activated with our Test Data unless otherwise instructed. You will receive a Test Key which contains sample identities for use during implementation and testing. When you are ready to go live, contact Veratad Support (support@veratad.com) to activate your live data feed.

Introduction

Veratad services are available through three delivery methods:

- Website Interface via Virtual Terminal
- Custom Batching Process
- API Web Service

This document describes the Veratad API Web Service and provides rules and specifications to access and make use of the verification services and resources provided by Veratad.

The Veratad API Web Service is delivered as XML over https.

Tags returned obey certain rules regarding type. For example, when `<date>` tags are returned they will contain 8 digits in the form YYYYMMDD which is standard or will be blank if no date is provided. In cases where partial date information is returned such as month and year only, content is rounded in a "safe" fashion. A date which will effect a minimum age calculation is always rounded up so as to never exaggerate an age. If a date of birth return contains year and month only, for example "197807", the tag will round forward; "19780731". On the other hand, dates of record such as a last updated tag are rounded downward so as not to indicate the record was updated more recently than it may have been. If a full date is returned, no rounding is performed.

In addition, spaces are used to identify component separation when parsing. For example, a street address entry of "26 N Franklin St" provides a different meaning from "26N Franklin St". "26 N" from the first address entry will be interpreted as house number = 26 and a street pre-direction = "N" with an equivalency = North. The second example with "26N" (no space) will be interpreted as house number = 26N with no street address pre-direction.

Finally, capitalization rules do not apply. For example, the first name entry of "stephen" provides the same meaning as "Stephen".

Accessing the XML Gateway

An API Web Services Client should connect to the IDResponse[SM] host using the https protocol, which provides Secure Socket Layer (SSL) encryption. Any program that sends and retrieves XML data in this manner can be utilized. Veratad uses cURL to send/receive XML queries.

If you are using cURL instructions may be found at (http://curl.haxx.se). Veratad supports curl and other API implementation options. Examples provided below are based on using cURL on a Linux platform either directly with the "cURL" application using the PHP API. cURL for Microsoft Windows is also available.

All API Web Service queries will access an IDResponse gateway:

- Standard Gateway Input/Output: https://idresponse.com:6650/gateway.php
- Standard with OFAC Input/Output: https://ofac.idresponse.com:6650/gateway.php
- Contact us at support.veratad.com for other address availability.

Queries are sent into the gateway and response results are returned in one https transaction.

THE HTTPS REQUEST

To form a query, group the required XML tags and data into a string, and assign the string to an HTTP variable named "query".

For example:

```
query=
<query>
    <user>username@mydomain.com</user>
    <pass>secret</pass>
    <function>age</function>
    ...
</query>
```

Send the query string to the Gateway URL using either a POST or GET HTTP request. The POST method is preferred. If you use the GET method, you must URL- encode the body of the query string. Time-out values may need to be adjusted to accommodate processing time variations. Examples below illustrate time-out value modification along with other cURL setting adjustments.

See figures 1 and 2 for examples of queries using the command-line version of cURL. The "-k" option causes cURL to continue with the request regardless of possible ssl problems. The "-d" option specifies the POST data to send in the request. For a complete summary of all of the available command-line options, run curl with the "—help" flag or consult the on-line documentation. See figure 3 for an example of using cURL from within php. For a complete list of options, see (http://www.php.net/manual/en/function.curl-setopt.php).

```
curl -k -d \
"query=<query><user>johndoe@somedomain.com</user><pass>secre
t</pass><function>idr-calc</function></query>" \
https://idresponse.com:6650/gateway.php
```

Figure 1: POST command-line query

```
curl -k https://idresponse.com:6650/gateway.php?\
query=%3Cquery%3E%3Cuser%3Ejohndoe%40somedomain.com%3C%2Fuse
r%3E\%3Cpass%3Esecret%3C%2Fpass%3E\%3Cfunction%3Eidr-
calc%3C%2Ffunction%3E%3C%2Fquery%3E
```

Figure 2: GET command-line query

```
$error_occurred = true;
$url = https://idresponse.com:6650/gateway.php;
$xml_query = "<query><user>johndoe@somedomain.com</user>" .
"<pass>secret</pass><function>idr-calc</function></query>"
// initialize cURL
$curl_channel = curl_init();
// specify destination and data
curl_setopt($curl_channel, CURLOPT_POSTFIELDS, 'query=' . $xml_query);
curl_setopt($curl_channel, CURLOPT_POST, 1);
curl_setopt($curl_channel, CURLOPT_URL, $url);
/*
* for GET request you would replace the above three lines with
* curl_setopt($ch1, CURLOPT_URL,
* $url . "?" . 'query=' .urlencode($xml_query));
*/
// set other options
curl_setopt($curl_channel, CURLOPT_FAILONERROR, 1);
curl_setopt($curl_channel, CURLOPT_FOLLOWLOCATION, 1);
curl_setopt($curl_channel, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($curl_channel, CURLOPT_CONNECTTIMEOUT, 30);
curl_setopt($curl_channel, CURLOPT_TIMEOUT, 200);
curl_setopt($curl_channel, CURLOPT_SSL_VERIFYPEER, 0);
// perform the request
$xml_result = curl_exec($curl_channel);
// check for errors
if ($xml_result === false)
$error_occurred = true;
```

Figure 3: Using cURL in php

THE BASIC QUERY

Each query must be submitted for a particular service. Within the XML request the service is submitted within the <function> tag. Although all queries are specific to function, all queries share common elements. For details regarding function-specific elements, please refer to the section of the documentation relating to that function. Function options are listed in Table 1.

Function Tags:

Function	Product name
<age>	AgeMatch/IDMatch
<authenticate4>	IDMatch+PLUS
<idr-calc>	IDR-Calc

Table 1: List of web service functions

To return the most current output-type enter 4.3b in the output-type tag as shown below:

```
<query>
    <user>username@mydomain.com</user>
    <pass>secret</pass>
    <function>ssn</function>
    <output-type>4.3b</output-type>
    <user-sequence></user-sequence>
    <client-side></client-side>
    <!-- function specific tags... -->
</query>
```

Additional details of the component tags are provided below:

<query> - The XML Query root node.

<user> - Veratad provided user name required to gain access to the web service.

<pass> - Password associated with the above user name.

<function> - The web service function you are requesting. See Table 1 for a list of functions, and later sections for detailed information about each.

<output-type> - The output format to return. The current version is 4.3b. The Output tag is *required* to contain the current Web Service version to produce the appropriate return.

<user-sequence> - Text or XML content that is passed through to the result but is not recorded in the transaction record. Any XML is allowed as long as it is combined with the <user-sequence> tag and is well-formed.

<client-side> - XML data which is recorded with the transaction and is available in the reporting download section of Virtual Terminal. Use these tags to send in reference information such as order number, transaction id, customer id or any other client defined information that you would like to include in the XML

return message. Any XML is acceptable, as long as it, combined with the <client-side> tag, is well-formed. Note: data is stored only one level deep.

Client Side Tags Use:

Input	stored key	stored value
<client-side> <first_name>James</first_name> <last_name>Smith</last_name> <csip>172.16.1.1</csip> <OrderNo>12345</OrderNo> <Amount>50.00</Amount> <simple>simplevalue</simple> <nested><a>12</nested> </client-side>	 first_name last_name csip OrderNo Amount Simple Nested	 James Smith 172.16.1.1 12345 50.00 Simplevalue <a>12

Table 2 Example use of client-side tag

Reserved Client Side Tags:

<first_name>	First name submitted for verification and retained for reporting purposes
<last_name>	Last name submitted for verification and retained for reporting purposes.
<ip>	IP address of your server that the end-user is using.
<csip>	IP address of the end-user.
<orderno>	To record your transaction number.
<amount>	To record amount of the transaction.
<salestax>	To record tax amount.
<shipping>	To record shipping amount.
<custusername>	To record username of the end-user.
<custemail>	To record email address of the end-user.

Table 3 Listing of reserved client-side tags

THE BASIC RESPONSE

Each query response is returned in XML format enclosed within the root element <idr-message>. Although all responses are specific to the function requested, all responses share common elements. For details regarding function-specific elements of the response, please refer to the portion of the documentation relating to that function.

The following is an example of a response, leaving out most function specific items:

```
<?xml version="1.0" encoding="UTF-8"?>
<idr-message xmlns="http://idresponse.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://idresponse.com idr_return_4_3b.xsd">
   <copyright>This programming...</copyright>
   <status>0</status>
```

```
    <message></message>
    <transaction-id>239562</transaction-id>
    <time-stamp>2005-12-07T12:28:37-05:00</time-stamp>
    <user-sequence></user-sequence>
    <client-side></client-side>
    <output-type>4.3b</output-type>
    <legal-age>21</legal-age>
    <function>age</function>
    <!-- function specific tags -->
</idr-message>
```

The following is an expanded sample: (For brevity, portions of the response not pertinent to the example have been removed.)

```
<?xml version="1.0" encoding="UTF-8"?><idr-message
xmlns="http://idresponse.com/idr"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://idresponse.com/idridr_return_4_3b.xsd">
<copyright>This programming... </copyright>
<status>0</status>
<message></message>
<transaction-id>10115254</transaction-id>
<time-stamp>2010-08-03T11:40:46-04:00</time-stamp>
<client-side>... </client-side>
<output-type>4.3b</output-type>
<legal-age>18</legal-age>
<function>age</function>
<original-query>... </original-query>
<authentication>
   <age><code>1</code><text>positive</text><deceased><code>2</code>
   <text>negative</text></deceased></age>
   <bill>
      <code>1</code><text>positive</text><percentage>94</percentage>
      <closest>... </closest>
      <match>
         <firstname><code>1</code><text>positive</text></firstname>
         <lastname><code>1</code><text>positive</text></lastname>
         <streetnumber><code>1</code><text>positive</text></streetnumber
      >
      ... </match>
   </bill>...
</authentication>
```

Additional details of The Basic Response component tags are provided below:

`<idr-message>` - The XML Return root node.

`<copyright>` - The complete copyright statement as listed in appendix A.

`<status>` - The status indicate success or failure of the entire transaction. See Table 4 for additional information.

`<message>` - Message is a text description associated with the status. The message tag always appears in the output, even if there is no message returned.

<transaction-id> - The number Veratad uses to record a transaction. We recommend storing this number with your database along with other information retained about each transaction for reference purposes.

<time-stamp> - The date and time the transaction was processed in ISO 8601 format.

<user-sequence> - The user sequence provided in the query, if any, as described above.

<client-side> - Contains the information sent to the IDResponse Web Service via the client-side tags.

<output-type> - The output type of this result, which should correspond to the requested output type requested.

<legal-age> - The default minimum age for the user account performing the transaction. This is included as supporting information to make clear the meaning of any age code result that might be in the return.

<function> - The function that was requested.

STATUS CODES

Status Codes indicate the success or failure of the entire transaction and answers the question: Was the request successfully submitted and handled? They are returned within the <status> tag and are accompanied by a corresponding text description which provides additional information in the event the request was not handled successfully.

Status Tags

<status>	<message> details	(appended to <message>)
0	Transaction was successful	
1	Person not found with information entered	
2	No relevant information found	
3	User ID and/or password not provided in query	
4	Incorrectly formed XML request	No query string provided
4	Incorrectly formed XML request	Query did not include a function value
4	Incorrectly formed XML request	Unable to begin parsing
4	Incorrectly formed XML request	Query did not include a function value
4	Incorrectly formed XML request permission to function denied	Unknown function
5	Userid and/or password is incorrect	
5	Userid and/or password is incorrect	user not found
5	Userid and/or password is incorrect	user not active
5	Userid and/or password is incorrect	company not found
5	Userid and/or password is incorrect	company not active

<status>	<message> details	(appended to <message>)
5	Userid and/or password is incorrect	incorrect password
5	Userid and/or password is incorrect	invalid client address
6	Your IDResponse account is out of funds	Please add funds
Status 1000-9999 are reserved for system error status codes.		
2000-2003	system is experiencing difficulties with data communication	
3000-3999	unexpected system conditions or possible program errors	
7000-7999	system is failing to retrieve data correctly	

Table 4: Veratad Status Codes

RETURN CODES

Return Codes indicate the result of a specific function within the transaction. They are returned within the <code> tag with a corresponding <text> tag providing additional information about the result.

Example:

```
<age>
<code>1</code>
<text>positive</text>
<deceased>
<code>2</code>
<text>negative</text>
</deceased>
</age>
```

The example contains two Return Codes. The first is within the <age> function and indicates a positive response. The second is within the <deceased> function and indicates a negative response.

Return Codes

<code>	<text>
<1>	<positive>
<2>	<negative>
<3>	<information unavailable>
<4>	<not sufficiently associated with identity>
<5>	<information is ambiguous>
<6>	<nickname match>
<7>	<misspelling>
<8>	<initial match>
<9>	<equivalence table match>
<10>	<vanity city>
<11>	<n/a>
Blank	(see below)

Table 5: Veratad Return Codes

While certain Return Codes are general codes that apply to all functions, others are reserved to specific functions. Additional description of Return Codes meanings are provided below:

<1><positive> - Indicates the field in the query sufficiently matches the data found. Qualifications for a "positive" result vary by field.

<2><negative> - Indicates the field in the query did not sufficiently match the data found.

<3><information unavailable> - Indicates the information required for the specific function was not returned. (For example, in the context of an age result, this would indicate that a record was returned however the record did not contain date of birth)

<4><not sufficiently associated>- Indicates the field in the query matches the data found however, the information returned did not meet Veratad's tolerance qualification.

<5><ambiguous> - Indicates multiple results with unresolved selection criteria. Additional information is provided in the result, as well as resolution criteria which can be used to modify the query to obtain an appropriate result.

<6><nickname>- Indicates a first name field matches the data found by utilizing Veratad's nickname equivalency tables.

<8><initial> - Indicates an initial match and is only applicable for first or middle name.

<9><equivalence>- Indicates a match utilizing the appropriate Veratad equivalency table. For example, the submission of "St" within an address field would result in an equivalency to "Street".

<10><vanity city>- Indicates a match utilizing a specific Veratad equivalency table which maintains commonly used substitutions for city names. This special form of equivalence match is dependent on the zip codes of the addresses that are being compared. Two arbitrary city names with the same zip code does not necessarily represent a vanity city.

<11><n/a>- No information was submitted.

PARENT CODES

Return Codes are associated with a Parent Code which is identified within a function tag. In the following example the <positive> Return Code is presented within the Parent Code = <age>.

Example:

```
<age>
    <code>1</code>
    <text>positive</text>
</age>
```

In the next example Return Codes are presented within three Parent Codes: (1) <ssn-9>, (2) <ssn-4> and the (3) <idr-calc>.

Example:

```
<ssn>
    <ssn-9>
        <code>11</code>
        <text>n/a</text>
    </ssn-9>
    <ssn-4>
        <code>1</code>
        <text>positive</text>
    </ssn-4>
    <idr-calc>
        <code>11</code>
        <text>n/a</text>
    </idr-calc>
</ssn>
```

In addition to using Return Codes for Age, Deceased and SSN results, Return Codes are used in more detailed tests associated with specific name and address components. Each detailed test presented with a Parent Code. The following table lists those Parent Codes along with applicable response codes.

Type of field	Possible Return Codes
All	1 (positive) 2 (negative) 3 (unavailable) 11 (not applicable) Blank
Title	9 (Equivalency)
First name	6 (Nick Name) 8 (Initial)
Middle name	8(Initial)
Name suffix	9 (Equivalency)
Street Pre direction	9 (Equivalency)
Street Post direction	9 (Equivalency)

Type of field	Possible Return Codes
Street Suffix	9 (Equivalency)
PO Box Designation	9 (Equivalency)
Route Designation	9 (Equivalency)
Apt Designation	9 (Equivalency)
City	10 (Vanity City)

Table 6: Veratad Field Type and Return Codes Listing

In addition to Response Code and Text tags, Name and Address match results may provide even more specifics including `<percentage>` and/or `<details>` tags. The `<details>` tag appears where there is a difference between the information entered as compared to what was found. The `<percentage>` tag adds a value to the variation between the submission and the return.

Example:

```
<city>
   <code>1</code>
   <text>positive</text>
   <percentage>90</percentage>
   <details>wellsvill --->wellsville</details>
</city>
```

AgeMatch/IDMatch℠ Web Service

The Veratad AgeMatch/IDMatch Service for the purpose of this document is the base service provided by Veratad. All other Veratad Services build upon the foundation provided by AgeMatch/ID Match. AgeMatch/ID Match utilizes information obtained from a series of trusted data sources providing access to billions of records containing data about individuals.

The Base Function provided by AgeMatch/IDMatch is the association of name and address accompanied by a determination of whether an individual meets a pre-set age requirement established with an account.

In addition to this base function, AgeMatch/IDMatch provides for the optional validation of additional verifiers such as date of birth information and SSN or SSN components information.

THE QUERY

The following section expands upon The Basic Query Section to provide instruction for building the XML input associated with the Veratad AgeMatch/IDMatch Web Service. The following is an example of Function tags which will form a simple AgeMatch/IDMatch query.

```
<query>
    <user>username@mydomain.com</user>
    <pass>secret</pass>
    function>age</function >
    <output-type>4.3b</output-type >
    <client-side></client-side>
    <fn>John</fn>
    <ln>Doe</ln>
    <bill>
        <addr>123 W. Long Rd. Apt 5N</addr>
        <city>Big City</city>
        <state>NJ</state>
        <zip>08934</zip>
    </bill>
</query>
```

VERIFYING INFORMATION

In addition to required name and address input tags, optional input tags are available which provide additional verifying information or "validation points". Verifying information is any of the following:

- IDRCalc℠: Calculations utilizing components of a social security number.
- Full Social Security Number.
- Last Four Digits of the Social Security Number
- Date of Birth
- Age

The following table lists the types of verifying information along with their tag reference:

Type	XML tag
IDR-Calc questions and answer	<question><id><answer></question>
Full Social Security Number	<last4>
Last four digits of the social security number	<last4>
Age	<age>
Date of Birth	<dob>, <dob-type>

Table 7 Verifying Information Tags

Full SSN and/or last 4 of SSN use same tag <last4>

Verifying information may be sent with any query. One of the effects of including verifying information is to reduce the number of ambiguous returns and to provide additional validation. It is important to note that verifying information is not used directly in the determination of the age result.

The following expands on the simple AgeMatch/IDMatch query to include validation points for the last four digits of a Social Security Number and the Date of Birth.

```
<query>
    <user>username@mydomain.com</user>
    <pass>secret</pass>
    <function>age</function>
    <output-type>4.3b</output-type>
    <client-side></client-side>
    <fn>John</fn>
    <ln>Doe</ln>
    <bill>
        <addr>123 W. Long Rd. Apt 5N</addr>
        <city>Big City</city>
        <state>NJ</state>
        <zip>08934</zip>
    </bill>
    <last4>7775</last4>
    <dob>09041964</dob>
    <dob-type>MMDDYYYY</dob-type>
</query>
```

IDR CALC QUESTIONS

One example of verifying information is Veratad's IDRCalc Product with provides an SSN validation based on a simple calculation performed against an SSN. To request an IDRCalc verifier you must identify a particular IDRCalc question when submitting the query. You can "anchor" your implementation to a single IDRCalc option by limiting the input to a single IDRCalc ID. You can

choose to rotate the IDR Calc by randomly rotating the IDRCalc ID you submit. The following is a sample input incorporating IDR Calc:

```
. . .</bill>
   <question>
      <id>dcalc3</id>
      <answer>19</answer>
   </question>
</query>
```

The following table lists current IDR Calc questions along with their corresponding IDRCalc ID:

idrcalcid	IDR Calc Question Selection
<dcalc1>	What is the sum of the **first digit and the last digit** of your Social Security Number?
<dcalc3>	What is the sum of the **last four digits** of your Social Security Number?
<dcalc4>	What is the sum of the **last 3 digits** of your Social Security Number?
<dcalc6>	What is the sum of the **first digit and fifth digit** of your Social Security Number?
<dcalc7>	What is the sum of the **6th digit** and **last digit** of your Social Security Number?
<dcalc8>	What is the sum of the **7th digit and 8th digit** of your Social Security Number?

Table 8 IDR Calc Questions

THE RESPONSE

The following sections expand upon The Basic Response section to provide instructions for interpreting the XML response associated with the Veratad AgeMatch/IDMatch Web Service. Function specific tags will be returned following those tags described in The Basic Response section. The first if these tags will be encapsulated within the <original-query> parent tag and presents a recap of the query which was submitted. For example:

```
...<function>age
</original-query>
   <name>
      <input-firstname>Mr John H</input-firstname>
      <input-lastname>Doe Jr</input-lastname>
      <title>mr</title>
      <firstname>John</firstname>
      <middlename>H</middlename>
      <lastname>Doe</lastname>
      <suffix>jr</suffix>
      <ssn>1234</ssn>
      <dob>19700131</dob>
   </name>
   <bill>...
</original-query>
```

In addition to the tags submitted in the query, additional detail is provided. The input first name and last name is parsed into title, first, middle, last and suffix tags. The input address is parsed into the appropriate address fields. Continuation of above example:

```
...<bill>
      <input_address>123 W Long Rd Apt 5N</input_address>
      <streetnumber>123</streetnumber>
      <streetpredirection>w</streetpredirection>
      <streetname>long</streetname>
      <streetsuffix>rd</streetsuffix>
      <streetpostdirection></streetpostdirection>
      <boxdesignation></boxdesignation>
      <boxnumber></boxnumber>
      <routedesignation></routedesignation>
      <routenumber></routenumber>
      <unitdesignation>apt</unitdesignation>
      <unitnumber>5n</unitnumber>
      <city>Big City</city>
      <state>NJ</state>
      <zip>08934</zip>
      <phone>5551212</phone>
   </bill>
```

Supplemental data is sometimes returned if an alert associated with the original input address is found. The alert contains an <id> and <message> tag and is found within the <flag> parent code. Using the prior example if an alert was found it would be presented as follows:

```
<bill> . . .
   <flag>
      <id>FA01</id>
      <message>Address invalid</message>
   </flag>
</bill>
```

The following table lists those Alerts along with their applicable descriptive message.

<id>	<message>
<FA01>	Address invalid
<AA01>	Prison address
<AA02>	Temporary address
<AA03>	Mail delivery address
<AA04>	Military address
<AA05>	Business address
<AA06>	Address unit number missing
<AP01>	Phone listed to a prison
<AP02>	Phone listed to a temporary address
<AP03>	Phone listed to a mail delivery location

Table 9: Supplemental address and phone information

AUTHENTICATION

Immediately following the </original input> tag, the <authentication> parent tag presents the results of the query. The <authentication> tag is returned only when there was no error, i.e., <status> is "0". The results are then encapsulated with authentication sub groupings

Presented within the <age> tag, the first section of the authentication will tell you whether the verifying data indicates the name and address are associated with each other and if the individual meets the minimum age established for your account, typically 18 or 21 years old. The following is an example of an <age> return:

```
</original-query>
<authentication>
   <age>
      <code>1</code>
      <text>positive</text>
      <deceased>
         <code>2</code>
         <text>negative</text>
      </deceased>
   </age>
```

The following table lists possible Age Codes along with their interpretation.

<code>	<text>	Interpretation
<1>	<positive>	Meets minimum age and address is associated with name
<2>	<negative>	Does not meet minimum age
<3>	<information unavailable>	Verifying date of birth data could not be found.
<4>	<not sufficiently associated with identity>	Meets minimum age however there were differences identified in the address.
<5>	<information is ambiguous>	More than one identity was found that matched the input.

Table 10: Age Code Interpretation

An additional numeric value is returned when a date of birth is submitted in the original query. The value is referred to as an Age Delta which represents the difference between the year of birth submitted and the year of birth located. A positive numeric value indicates a submitted age is greater the age found while a negative numeric value indicates that verifying data indicates the individual is younger than indicated. Age Delta is only returned for Age Codes 1, 2 and 4. The following is an example <age> return with an Age Delta.

```
   </original-query>
<authentication>
   <age>
      <code delta="5">1</code>
      <text>positive</text>
      <deceased>
         <code>2</code>
         <text>negative</text>
```

```
    </deceased>
</age>
```

The <age> function also provides an alert in the event the individual is identified as being deceased. The alert is presented in the <deceased> tag.

The following table lists Deceased Codes along with their interpretation.

<code>	<text>	Interpretation
<1>	<positive>	Indicates the individual is deceased
<2>	<negative>	Indicates the individual is not deceased
<5>	<information is ambiguous>	More than one identity was found that matched the input.

Table 11: Deceased Code Interpretation

Presented within the <phone> tag, the second section of the authentication will tell you whether the submitted phone number matches any of the phone number returned from verifying data. As a result of the ubiquitous nature of mobile phones and the increased transience of phone numbers Veratad suggests that <phone> results only be used as a passive verifier.

The following table lists Phone Codes along with their interpretation.

<code>	<text>	Interpretation
<1>	<positive>	Phone number submitted matches a number returned (last 7 digits).
<2>	<negative>	Phone number submitted does not match any numbers returned.
<3>	<information unavailable>	Phone number was submitted and no phone number was returned.
<11>	<n/a>	No phone number was submitted.

Table 12: Phone Code Interpretation

The third section of the authentication is Social Security Number (SSN). Presented within the <ssn> tag, the content of this tag will tell you whether the SSN information submitted matches the SSN information returned from verifying data. Within the <ssn> tag additional tags are returned which indicate the type of SSN validation being performed. They are: <ssn-9> - Indicates test for full SSN, <ssn-4> - indicates test for the last four digits of the SSN, and <idr-calc> - Indicates IDRCalc test. Each type of SSN with have a return code.

The following table lists SSN Codes along with their interpretation.

<code>	<text>	Interpretation
<1>	<positive>	SSN or component matches a number returned.

<code>	<text>	Interpretation
<2>	<negative>	SSN or component does not match number returned.
<3>	<information unavailable>	SSN or component was submitted and no SSN was returned.
<11>	<n/a>	No SSN or component was submitted.

Table 13: SSN Code Interpretation

AMBIGUOUS RESULTS

Occasionally the response will consist of multiple and distinct identities that match the submitted query. The frequency of an ambiguous result decreases with the submission of additional information in the query. Ambiguous results occur most commonly when a "Jr" and "Sr" reside in the same household.

The following is an example of an Ambiguous Response.

```
<age>
   <code>5</code>
   <text>information is ambiguous</text>
   <deceased>
      <code>5</code>
      <text>information is ambiguous</text>
   </deceased>
   <ambiguous>
      <code>1</code>
      <text>positive</text>
      <person>
         <firstname>John</firstname>
         <lastname> Doe </lastname>
         <name>John Doe </name>
         <age>
            <code>1</code>
            <text>positive</text>
            <deceased>
               <code>2</code>
               <text>negative</text>
            </deceased>
         </age>
         <question>
            <id>age</id>
            <prompt>How+old+are+you%3F<prompt>
            <answer>22</answer>
         </question>
      </person>
      <person>
         <firstname>John</firstname>
         <lastname> Doe </lastname>
         <name>John Doe </name>
         <age>
            <code>1</code>
            <text>positive</text>
```

```
            <deceased>
                <code>2</code>
                <text>negative</text>
            </deceased>
        </age>
        <question>
            <id>age</id>
            <prompt>How+old+are+you%3F<prompt>
            <answer>51</answer>
        </question>
    </person>
</ambiguous>
</age>
```

Within the <age> grouping there are two additional tags: <ambiguous> and <person>. The <ambiguous> encapsulates each of the identities. Each identity and has underlying <age> and <deceased> tags. You may determine that the verification can be completed by using these underlying tags.

In the event that the underlying <age> and <deceased> tags do not supply sufficient detail to complete your verification, a <question> is provided which can be presented and based upon the <answer> can be used to formulate a new query from which the ambiguity would be resolve. This process is described in detail in the next section.

The frequency of an ambiguous result decreases with the submission of additional information in the query. In the example above, if a date of birth was submitted in the query, the ambiguity is avoided if the date of birth entered matches one of the dates of birth found.

IDMatch+PLUS℠ Web Service

The Veratad IDMatch+PLUS Service expands upon the base service provided in the AgeMatch/IDMatch Service by incorporating additional verifying information in the form of historical and biographical, knowledge based challenge questions. Veratad IDMatch+PLUS delivers all the information necessary to present historical, biographical questions and determine the correctness of answers provided to those questions in a single response to the initial query.

THE QUERY

To initiate an IDMatch +Plus response, adjust the query described in The Query section of AgeMatch/IDMatch Section to contain "authenticate4" as the content in the `<function>` tag of the input XML. The following is an example of Function tags which will form an IDMatch +PLUS query.

```
<query>
   <user>username@mydomain.com</user>
   <pass>secret</pass>
   <function>authenticate4</function>
   <output-type>4.3b</output-type>
```

THE RESPONSE

The following section expands upon The Response section of the AgeMatch/IDMatch Web Service Section to include 4 randomly rotated challenge questions and corresponding answers.

The IDMatch+PLUS response contains an additional parent tag entitle `<questions>`. Within the `<questions>` tag there are 4 `<question>` tags. Each will contain a question, 5 multiple choice answers and an indicator for the correct answer. The following is an excerpt of one `<question>` tag in an IDMatch+PLUS response:

```
. . .<questions>
      <question>
         <id>city4</id>
         <prompt>With+which+of+the+following+cities+are+you+or+have+you+been+assoc
         iated%3F+%3Ci%3E%28choose+all+applicable%29%3C%2Fi%3E</prompt>
         <multiple>true</multiple>
         <answers>
            <answer>
               <truth>false</truth>
               <value>Lake Villa</value></answer>
            <answer>
               <truth>false</truth>
               <value>Bolton</value></answer>
            <answer>
               <truth>true</truth>
               <value>Jersey City</value></answer>
            <answer>
               <truth>false</truth>
               <value>Aptos</value></answer>
```

```
            <answer>
               <truth>false</truth>
               <value>None Of The Above</value></answer>
         </answers>
      </question>. . .
. . . <questions>
```

The first tag following each <question> tag is an <id> tag followed by a <prompt> tag. The <prompt> tag contains the textual question for presentation. The following table lists the question IDs currently supported by IDMatch+PLUS along with the question prompt:

<id>	<prompt>
<address 1>	With which of the following addresses are you or have you been associated?
<address_city1>	XXXXX XX is in which of the following cities?
<age1>	Which of the following best represents your age?
<city4>	With which of the following cities are you or have you been associated ?
<college1>	Which of the following colleges have you attended?
<county1>	In which county is XXXXX XX located?
<email1>	Which of the following email addresses are you or have you been associated?
<employer1>	With which of the following companies have you ever been employed?
<housenumber1>	Which of the following house numbers is associated with your address on XXXXX XX?
<maiden1>	What is your mother's maiden name?
<phone 1>	With which of the following phone numbers are you or have you been associated?
<saledate1>	Which of the following best represents the correct year in which the property located as XXXXX XX was purchased?
<school1>	Which of the following schools have you attended?
<state1>	With which of the following states are you or have you been associated?

Table 14: Question ID List

Immediately following each question there is a <multiple> tag the content of which is limited to true/false. A <multiple> tag = "true" indicates there may be more than one correct answer provided. A <multiple> tag = "false" indicates that only a single answer is applicable.

Within each <question> block five multiple choice answers are provided within an <answers> block, each answer is encapsulated within an <answer> block containing a <truth> tag and a <value> tag. The <value> contains a multiple choice answer to be presented while the <truth> tag indicates whether the <value> provide is a correct answer to the question.

The questions and possible answers are delivered within the result for presentation to the user and to be answered by the user, all within your application. The answers provided by the user are not submitted back to Veratad. Instead the answers submitted should be compared to the tags provided within your application.

AMBIGUOUS RESULTS

Although ambiguous results may provide sufficient information within the underlying codes to perform an AgeMatch/IDMatch verification, IDMatch+PLUS questions are not returned with an ambiguous results.

IDMatch+PLUS questions can only be returned if an ambiguity is resolved. The key to resolving the ambiguity is provided in the ambiguous Response. A question is provided within a <question> tag corresponding to each individual returned. Each contains an <id>, <prompt> and <answer> tag. The <prompt> tag contains the textual question for presentation to the user. Once an answer is submitted, it should be compare to the content of the <answers> tag. If the answer submitted does not match any of the answers provided the transactions should be treated as a "person not found".

If on the other hand, the answer submitted matches the <answer> in one of the <person> tags follow these steps to build a requery.

1. Capture the fields submitted in the initial query.
2. Add verifying information as described in the "Verifying Information" section presented earlier in this document by submitted the ambiguous <question> <id> in the <function> tag along with the answer provided.

If an ambiguous response does not contain the <question> tag the transaction should be treated as a "person not found".

If we continue the Ambiguous example provided in the AgeMatch/IDMatch, assuming the end user responded to the question "How Old Are You?" with "22" as the answer, the Query would be built and resubmitted as follows:

```
<query>
    <user>username@mydomain.com</user>
    <pass>secret</pass>
    <function>authenticate4</function >
    <output-type>4.3b</output-type >
    <client-side></client-side>
    <fn>John/fn>
    <ln>Doe</ln>
    <bill>
        <addr>123 W. Long Rd. Apt 5N</addr>
        <city>Big City</city>
        <state>NJ</state>
        <zip>08934</zip>
    </bill>
    <age>22</age>
</query>
```

The inclusion of <age>22<age> in the query will resolve the ambiguity initial experienced.

CustomMatch℠ Web Service

The Veratad CustomMatch℠ Service expands the base service provided in the AgeMatch/IDMatch Service by incorporating the application of custom business rules. Veratad develops custom business rules to meet specific objectives as defined by a Client.

The CustomMatch Service will allow Veratad Clients to customize Responses they receive to Queries based on their pre-defined business rules. CustomMatch Responses can take the form of an "Append" where tags are added to the standard AgeMatch/IDMatch Basic Response or a "Replace", where tags replace the Basic Response.

THE QUERY

To initiate a CustomMatch response, adjust the query described in the Query section of AgeMatch/IDMatch to contain the value provided by Veratad in the `<function>` tag of the input XML. The following is an example of Function tags that form a CustomMatch query:

```
<query>
   <user>username@mydomain.com</user>
   <pass>secret</pass>
   <function>custom</function>
   <output-type>4.3b</output-type>
```

THE RESPONSE - APPEND

The following tags will be added to the XML delivered for AgeMatch/IDMatch and indicate the result of the application of the Client business rules in their entirety:

```
... </function>
<result>
   <code>1</code>
   <text>positive</text>
</result>
```

Additional detail is provided for each tag:

`<code>` – Return code and text indicate whether the Client business rule was met.

<code>	<text>	Interpretation
<1>	<positive>	Client business rules were met, i.e, "PASSED".
<2>	<negative>	Client business rules were not met, i.e., "FAILED"

Table 15: Custom return codes

THE RESPONSE - REPLACE

The tags returned will not be added to the XML delivered for AgeMatch/IDMatch. Instead Veratad will formulate a customized response with a Client. Customized responses can be as brief as a simple "PASS"/"FAIL". Clients can chose to incorporate components of the standard AgeMatch/IDMatch response and/or add unique Client provided tags. Most XML is acceptable, as long as it is well-formed.

The following is an example of a Custom Response:

```
<?xml version="1.0" encoding="UTF-8"?>
<idr-message xmlns="http://idresponse.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://idresponse.com idr_return_4_3b.xsd">
    <transaction-id>239562</transaction-id>
    <time-stamp>2005-12-07T12:28:37-05:00</time-stamp>
    <result>
        <code>1</code>
        <text>positive</text>
    </result>
    <!—client custom tags (if any) -->
</idr-message>
```

OFAC Service

The Veratad OFAC Service extends the response of the Veratad IDMatch and IDMatch+PLUS Services to include a cross reference of the name submitted in the original query to the Office of Foreign Asset Control (OFAC) – Specially Designated Nationals List.

THE QUERY

The input codes are the same as the input codes for Veratad IDMatch Services. To initiate a response containing OFAC cross reference you must first verify your account is set up to provide the Veratad OFAC Service with a Customer Service Representative via email at support@veratad.com. Then verification Request should be submitted to the Standard with OFAC location listed in the Accessing the XML Gateway section.

THE RESPONSE

The following tags will be included in the XML delivered for both AgeMatch/IDMatch and IDMatch+PLUS responses. :

```
... </function>
<ofac>
    <code>1</code>
    <text>positive</text>
    <reference>4321432</reference>
</ofac>
```

Additional detail is provided for each tag:

<code> – Return code and text indicate whether the input first and last name tags were a match in the OFAC database.

<code>	<text>	Interpretation
<1>	<positive>	Input name was found in the OFAC SDNL.
<2>	<negative>	Input name was not found in the OFAC SDNL.
<3>	<information unavailable>	No data was returned

Table 16: OFAC return codes

<reference> – a numeric key to identify the lookup instance against the OFAC SDNL

Out of Band Authentication

Veratad provides "Out of Band" ("OOB") Service strengthening Veratad's Verification Services by providing additional authentication opportunities through secondary communication channels where an independent authentication can be performed. Although Veratad OOB Service can be used in conjunction with Veratad's other Services Veratad OOB Service is a stand-alone service with no dependencies to other Verification API Services.

Veratad OOB messages are expected to take the form of XML messages sent to our web service. Communications are broken into two parts: requests and responses.

THE REQUEST

To form a Veratad OOB request, group the required XML tags and data into a string. The root tag is named "authentication". The message itself is split into a two components: Header and Body.

Request example:

```
<tns: Authentication
xmlns:tns="http://tempuri.org/Authentication"
xmlns:xsi="http:www.w3.org/2001/xmlschema-instance"
xsi:schemaLocation+"http://tempuri.org/authentication authentication.xsd>"
    <header>. . .
    </header>
    <body>. . .
    </body>
</tns:authentication>
```

THE REQUEST HEADER

The Header section conveys the metadata necessary to authenticate and reference the message.

Header example:

```
<header>
    <account>
        <accountid></accountid>
        <accoutpwd></accountpwd>
    </account>
    <custrefnumber></custrefnumber>
</header>
```

Additional details of the component tags are provided below:

<**accountid**> - Veratad-provided user name.

<**accountpwd**> - Password associated with the user name.

<**custrefnumber**> - Customer-assigned reference to the transaction.

THE REQEUST BODY

The Body consists of three required blocks of tags and optional tags. The required tag are <authdetail>, <usercredentials> and <phonedetails>. The optional tags are <messagedetails> and <phonedetails>

Body example:

```
<body>
    <request>
        <auth>
            <authdetail>
                <mode>2</mode>
                <authtype>
                    <type>1</type>
                    <asynch>0</asynch>
                </authtype>
            </authdetail>
        </auth>
        <usercredentials>
            <password>NNNN</password>
            <username>some_username<username>
        </usercredentials>
        <phonedetails>
            <countrycode>1</countycode>
            <phonenumber>NNNNNNNNNN</phonenumber>
        <messagedetails>{0}</messgaedetails>
        <voiceprompt></voiceprompt>
    </request>
</body>
```

The <**authdetail**> Tag specifies metadata regarding the nature of the request, which OOB service you are attempting to run, whether your inquiry is going to a phone and whether your request is to be handled synchronously or asynchronously. Additional details of the <**authdetail**> component tags are provided below:

<**mode**> - Defines what OOB services is being run. Mode options are listed in Table 15:

Mode	OOB Service
2	Phone Input
7	SMS Text Message
17	Phone Password Delivery

Table 17: List of OOB Services

<**type**> - always = 1

<**asynch**> - Indicates whether the transaction will be handled synchronously or asynchronously. synchronous = 0, asynchronous = 1

The <**usercredentials**> Tag contains information about the authentication you are conducting with your customer. It contains two of three tags. Additional details of the <**usercredentials**> component tags are provided below:

<**username**> - Contains some means of indicating which user in your system is being authenticated. Any string is allowed in this tag and need not be meaningful to Veratad, though the tag itself is required.

<**password**> - Used in conjunction with <mode> = 2 (phone input). Since this will require input into a phone via the phone's keypad, should contain only numbers.

<**onetimepassword**> - Used in conjunction with <mode> = 7 or 17.

The <phonedetails> Tag bears the information needed to identify a phone to call. Additional details of the <phonedetails> component tags are provided below:

<**coutrycode**> - Default = 1 for USA

<**phonenumber**> - 10 digit US phone number, no punctuation.

The <**messagedetails**> Tag is optional, and describes the format and content of a text message, and is therefore only applicable to the SMS Text Message service (mode 7). It contains only one tag, <**sms**>, which is the text to be sent out as a message. SMS messages are by their nature limited to 160 characters. Please take the length of the one-time password and timestamp into account when planning your message (not the length of the entities.)

Message Details example:

```
<messagedetails>
   <sms>Your password is {0}</sms>
</messagedetails>
```

Two entities can be inserted into the message, which will be replaced by other appropriate data. If either appears in the body of the message, they will be replaced by the appropriate content. , they are:

Entity	Meaning (replaced by)
{0}	onetimepassword
{1}	timestamp

Table 18: List of Message Details

The <voiceprompt> Tag is optional, and controls the message delivered to a phone via call, and is therefore only applicable to the Phone Message service (mode 2 or 17). It contains two tags, <**callerid**> and <**promptid**>.

<**callerid**> - content to be displayed as the originating number of the phone call and is limited to a 10 digit numeric string.

<**promptid**> - identifies a prerecorded message that will be played to the received of the call. Such message must be arranged with Veratad beforehand.

Voice Prompt example:

```
<voiceprompt>
   <callerid>2011231234</callerid>
   <promptid></promptid>
</voiceprompt>
```

THE RESPONSE

You will received one of two responses depending upon the value of the <**asynch**> flag you submitted in your request. The message itself is split into a two components: Header and Body.

Response example:

```
<tns: Authentication
xmlns:tns="http://tempuri.org/Authentication"
xmlns:xsi="http:www.w3.org/2001/xmlschema-instance"
xsi:schemaLocation+"http://tempuri.org/authentication authentication.xsd>"
   <header>. . .
   </header>
   <body>. . .
   </body>
</tns:authentication>
```

THE RESPONSE HEADER

The Response Header will contain the customer reference number.

Header example:

```
<header>
   <account>
      <accountid></accountid>
      <accoutpwd></accountpwd>
   </account>
   <custrefnumber>XXXXXXX</custrefnumber>
</header>
```

Additional details of the component tags are provided below:

<**accountid**> - Not used at this time.

<**accountpwd**> - Not used at this time.

<**custrefnumber**> - Customer-assigned reference to the transaction which was submitted with the corresponding Request.

THE RESPONSE BODY- SYNCHRONOUS

If your request was synchronous, then the entire transaction will be completed before a response is returned. In this case, you will immediately be presented with results. The <results>Body consists of three required blocks of tags. These are <authdetail>, <usercredentials> and <phonedetails>. The results are contained within a block call <resultset>. Within the <result> block :

Sychronous Body example:

```
<body>
   <response>
      <result>
         <resultset>
            <resultcode>0</resultcode>
            <transactionid>XXXXXXXX</transactionid>
            <custrefnumber>XXXXXX</custrefnumber>
            <error>
               <errorcode>NN</errorcode>
               <errormessage>XXXXXXXX</errormessage>
            </error>
         </resultset>
      </result>
   </response>
</body>
```

Additional details of the <resultset> component tags are provided below

<**resultcode**> - will be 0 in the event of a successful authentication and 1 in the case of a failure.

<**transactionid**> - Integer denoting Veratad's reference number assigned to the transaction.

<**custrefnum**> - String submitted by client with the request.

<**errorcode**> - Integer denoting Veratad's error code. Please refer to following table for a list of possible errors. This tag will be empty if no errors are encountered.

<**errormessage**> - String describing the nature of the error. Please refer to the following table for a list of possible errors. This tag will be empty if no errors are encountered.

Error Code	Error Message

Table 19: List of OOB Error Codes

THE RESPONSE BODY- ASYNCHRONOUS

If your request was asynchronous, the system will provide an immediate return with a means to reference the transaction response. It is left to you to check back at some future time to determine the end result of the transactions.

Asychronous Body example:

```
<<body>
    <response>
        <status>
            <ticket>********************************</ticket>
        </status>
    </response>
</body>
```

Additional details of the <status> component tags are provided below

<**ticket**> - A numeric hash referring to a transaction-in-progress, which is required to submit a follow-up status request.

THE STATUS REQUEST AND RESPONSE - ASYNCHRONOUS

To follow up on an asynchronous transaction, refer to the header and body format of a Request and substitute a <status> block in place of the <auth>block.

Status Request Body example -

```
<body>
    <request>
        <status>
            <ticket>********************************</ticket>
        </status>
    </request>
</body>
```

In the event the transaction is not yet resolved, another asynchronous response will be returned. On the other hand, if the transaction has resolved, the response will follow the same form as a synchronous response.

Helpful Hints

1. Always disable the "submit" button on click to prevent unintended verification requests by double and impatient clicking.
2. Create name, address and personal information field rules to filter invalid information prior to submission.
3. Add First Name, Last Name and Reference information into the Client Side Tags for reporting cross reference.
4. Consider setting client side sessions to limit the number of attempts a user is allowed.
5. Formulate content to be retained for audit trail and reporting.
6. Provide privacy statement acknowledging 3rd party verification for users accept prior to query.
7. For more information contact support@veratad.com