

IP2Proxy™ Web Service

~~ Detect Anonymous Proxy by IP Address ~~

1. Overview	3
1.1 Overview of the IP2Proxy Web Service	4
1.2 Front End of IP2Proxy Web Service	4
1.2.1 Integration with In-house System	4
1.2.2 Direct Access to Hosted Web Service	6
1.3 Back End of IP2Proxy Web Service	7
1.3.1 IP Address to IP Number Conversion	7
1.3.2 Record Matching	8
1.4 Process Flow Overview	8
2. Implementation	9
2.1 Implementation Details	10
2.2 Basic Parameters - Input	12
2.3 Basic Parameters - Output	12
2.4 List of possible value for MESSAGE field	12
2.5 Structure of the Request and Response	12
2.5.1 Request by SOAP	12
2.5.2 Response by SOAP	13
2.5.3 Request by HTTP-GET	13
2.5.4 Response by HTTP-GET	13
2.5.5 Request by HTTP-POST	13
2.5.6 Response by HTTP-POST	14
3. Design Information	15
3.1 Placement of IP2Proxy Web Service	15
4. Appendix I : ISO3166 Country Code	16
5. Appendix II : Sample Code	22

1. Overview

This documentation provides a basic understanding and information to help you get started with our products. Look over this documentation to gain a high-level understanding of the process flow that underlies the IP2Proxy Web Services.

For more information, please visit <http://www.fraudlabs.com> or contact your FraudLabs representative:

Email : sales@fraudlabs.com

1.1 Overview of the IP2Proxy Web Service

IP2Proxy Web Service is the proprietary anonymous IP detection service that enables you to check proxy servers for anonymity in order to reduce fraud for online merchants. As sophisticated fraudsters can confuse online merchants by sending them a fake IP address and bypass IP Geolocation clarification along with their requests, IP2Proxy becomes a critical component and offers an additional layer of protection for your business. Merchants can make use of our service in order to automate order process according to the results that we provide.

IP2Proxy Web Service is hosted on redundant servers, redundant Internet connections, and 24x7x365 monitoring. IP2Proxy Web Service is using platform independent XML format to exchange data between systems. All customers can integrate our web service regardless of their web server and shopping cart solutions.

Key Features Include:

- Protect from IP address spoofing
- Verify whether the online buyer is behind anonymous proxy servers
- Prevent and reduce fraud by authenticating online visitors
- Provides a complete XML-based Web Services API
- Contains sample code examples for ease integration

1.2 Front End of IP2Proxy Web Service

The general idea is that front end is responsible for collection input from the user and conforms to some specification that the back end can use. Front end of IP2Proxy Web Service is rather simple to understand. As we are using platform independent XML format to exchange data between systems, you may either integrate our web service to your in-house system, or direct access to our hosted web service.

1.2.1. Integration with In-house System

1. Our sample codes in different languages are available at: <http://www.fraudlabs.com/ip2proxysamplecodes.aspx> and download sample codes that you need (For sample codes

in different languages please refer to Appendix II). Below are links to get sample codes:

- i. Microsoft ASP.NET - VB.NET:
<http://www.fraudlabs.com/ip2proxysamplecode/IP2ProxyWebServiceClientVB.zip>
- ii. Microsoft ASP.Net - C#:
<http://www.fraudlabs.com/ip2proxysamplecode/IP2ProxyWebServiceClientCSharp.zip>
- iii. Microsoft ASP 3.0:
<http://www.fraudlabs.com/ip2proxysamplecode/IP2ProxyWebServiceClientAsp.zip>
- iv. Java/ Apache:
<http://www.fraudlabs.com/ip2proxysamplecode/IP2ProxyWebServiceClientJava.zip>
- v. PHP:
<http://www.fraudlabs.com/ip2proxysamplecode/IP2ProxyWebServiceClientPHP.zip>
- vi. Python:
<http://www.fraudlabs.com/ip2proxysamplecode/IP2ProxyWebServiceClientPython.zip>
- vii. Perl:
<http://www.fraudlabs.com/ip2proxysamplecode/IP2ProxyWebServiceClientPerl.zip>
- viii. ColdFusion MX:
<http://www.fraudlabs.com/ip2proxysamplecode/IP2ProxyWebServiceClientColdFusion.zip>
- ix. Access 2000:
<http://www.fraudlabs.com/ip2proxysamplecode/IP2ProxyWebServiceClientAccess.zip>
- x. Excel 2000:
<http://www.fraudlabs.com/ip2proxysamplecode/IP2ProxyWebServiceClientExcel.zip>

II. Please go through 'readme' file that we provide together with the sample codes for more set up information

III. In order to use our service, you need to get your own license key – *How?*

- i. log on to <http://www.fraudlabs.com>
- ii. sign up as our registered member
- iii. check your email to complete user activation
- iv. get license key :
 - a) Free License Account:

- 1) at the left menu bar, under category of IP2Proxy™ Proxy Detection, click "free license "
- 2) view Terms of Use (*please note that you must agree with our Terms of Use before proceed)
- 3) click on "Get Free License Now"
- 4) the license key will be sent to your email
- b) Premium Subscription:
 - 1) at the left menu bar, under category of IP2Proxy™ Proxy Detection, click "subscribe now"
 - 2) fill in required field in the secure payment form
 - 3) click on "make payment"
 - 4) the license key will be sent to your email after your payment is confirmed by our online payment merchant

IV. Now fill in the required field

V. Click on "Submit" and result is provided

1.2.2. Direct Access to Hosted Web Service

- I. Log on to <http://ws.fraudlabs.com/ip2proxywebservice.aspx>
- II. Click on "IP2Proxy"
- III. In order to use our service, you need to get your own license key – *How?*
 - i. log on to <http://www.fraudlabs.com>
 - ii. sign up as our registered member
 - iii. check your email to complete user activation
 - iv. get license key :
 - a) Free License Account:
 - 1) at the left menu bar, under category of IP2Proxy™ Proxy Detection, click "free license"
 - 2) view Terms of Use (*please note that you must agree with our Terms of Use before proceed)
 - 3) click on "Get Free License Now"
 - 4) the license key will be sent to your email
 - b) Premium Subscription:
 - 1) at the left menu bar, under category of IP2Proxy™ Proxy Detection, click "subscribe now"
 - 2) fill in required field in the secure payment form
 - 3) click on "make payment"

- 4) the license key will be sent to your email after your payment is confirmed by our online payment merchant

IV. Now you fill in the particular field

V. Click on “Invoke” and result is provided

1.3 Back End of IP2Proxy Web Service

Back end is the part that processes the input from the front end. The process is seamless to the end-user and works by interaction with a SOAP API through IP2Proxy Web Service.

The process works as follows:

1. User submits a form containing the parameters (inputs) that to be analyzed
2. System verify the license key before proceed
3. If the license key is valid, it will proceed and check the credits availability. If the remaining credits still available, it will start to analyze the parameters
4. The results then will be submitted for IP address converting and anonymity checking process
5. Returns and display the results

1.3.1. IP Address to IP Number Conversion

If the provided IP address is 161.132.13.1, then the IP number is 2709785857. - *How to get this conversion done?*

In general, the formula to convert an IP Address to IP Number is given as below:

Assume that the IP Address is in this format: A.B.C.D.

$$\text{IP Number} = A * (256 * 256 * 256) + B * (256 * 256) + C * (256) + D$$

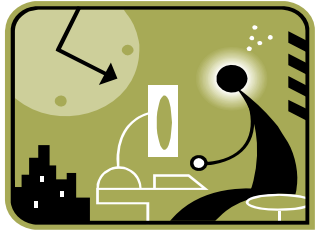
So for the IP address 161.132.13.1, the IP number is:

$$\begin{aligned} \text{IP Number} &= 161 * (256 * 256 * 256) + 132 * (256 * 256) + 13 * \\ &\quad (256) + 1 \\ &= 2709785857 \end{aligned}$$

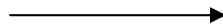
1.3.2. Record Matching

Next, our web service will look through our daily updated database to search a record that matches the anonymous IP address.

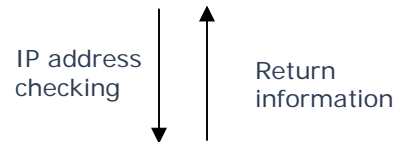
1.4 Process Flow Overview



Step 1
Visitors visit websites



Step 2
Visitor's IP address
captured by merchant



Step 3
IP2Proxy Web Service
(Proxy checking)

2. Implementation

This section provides basic information of the process of integrating the web service into your website. Look over this section to gain a high-level understanding of requesting IP2Proxy Web Service.

The implementation section covers the following topics:

- IP2Proxy Function

For those with solid SOAP programming knowledge, integration basics are included below. POST all requests to:

<http://ws.fraudlabs.com/ip2proxywebservice.asmx>

For more information about IP2Proxy Web Service implementation, please visit <http://www.fraudlabs.com> or contact your IP2Proxy representative:

Email : sales@fraudlabs.com

2.1 Implementation Details

This section was created for those who wish to develop applications that make use of IP2Proxy Web Service.

SOAP - IP2Proxy Function Request

```
POST /ip2proxywebservice.asmx HTTP/1.1
Host: ws.fraudlabs.com
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://ws.fraudlabs.com/IP2Proxy"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <IP2Proxy xmlns="http://ws.fraudlabs.com/">
      <IP>string</IP>
      <LICENSE>string</LICENSE>
    </IP2Proxy>
  </soap:Body>
</soap:Envelope>
```

SOAP - IP2Proxy Function Response

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <IP2ProxyResponse xmlns="http://ws.fraudlabs.com/">
      <IP2ProxyResult>
        <IPADDRESS>string</IPADDRESS>
        <ANONYMOUSPROXY>string</ANONYMOUSPROXY>
        <CREDITSAVAILABLE>int</CREDITSAVAILABLE>
        <MESSAGE>string</MESSAGE>
      </IP2ProxyResult>
    </IP2ProxyResponse>
  </soap:Body>
</soap:Envelope>
```

HTTP GET- IP2Proxy Function Request

```
GET /ip2proxywebservice.asmx/IP2Proxy?IP=string&LICENSE=string HTTP/1.1
Host: ws.fraudlabs.com
```

HTTP GET- IP2Proxy Function Response

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<IP2PROXY xmlns="http://ws.fraudlabs.com/">
  <IPADDRESS>string</IPADDRESS>
  <ANONYMOUSPROXY>string</ANONYMOUSPROXY>
  <CREDITSAVAILABLE>int</CREDITSAVAILABLE>
  <MESSAGE>string</MESSAGE>
</IP2PROXY>
```

HTTP POST- IP2Proxy Function Request

```
POST /ip2proxywebservice.asmx/IP2Proxy HTTP/1.1
Host: ws.fraudlabs.com
Content-Type: application/x-www-form-urlencoded
Content-Length: length

IP=string&LICENSE=string
```

HTTP POST- IP2Proxy Function Response

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<IP2PROXY xmlns="http://ws.fraudlabs.com/">
  <IPADDRESS>string</IPADDRESS>
  <ANONYMOUSPROXY>string</ANONYMOUSPROXY>
  <CREDITSAVAILABLE>int</CREDITSAVAILABLE>
  <MESSAGE>string</MESSAGE>
</IP2PROXY>
```

A WSDL is available at:

<http://ws.fraudlabs.com/ip2proxywebservice.asmx?wsdl>

A description of the IP2Proxy operation is available at:

<http://ws.fraudlabs.com/ip2proxywebservice.asmx?op=IP2Proxy>

2.2 Basic Parameters - Input

Field	Format	Description
IP	Required	IP address of credit card holder.
LICENSE	Required	License key for free license and premium users.

2.3 Basic Parameters - Output

Field	Format	Description
IPADDRESS	String	IP address that to be checked.
ANONYMOUSPROXY	YES or NO	Whether IP address is from Anonymous Proxy Server.
CREDITSAVAILABLE	Integer	Number of queries remaining in your account, can be used to alert you when you may need to add more queries to your account.
MESSAGE	String	Web Service Message Response

2.4 List of possible value for MESSAGE field

IP2Proxy Web Service Error Message
• Invalid license key
• Invalid IP address
• IP address and License Key cannot be blank
• No credit available
• License key was expired

2.5 Structure of the Request and Response

2.5.1. Request by SOAP

A valid request posted from user should look something like this:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <IP2Proxy xmlns="http://ws.fraudlabs.com/">
      <IP>68.142.197.65</IP>
      <LICENSE>xx-xxxx-xxxx</LICENSE>
    </IP2Proxy>
  </soap:Body>
</soap:Envelope>
```

2.5.2. Response by SOAP

After posting a valid request, IP2Proxy will return several parameters. An example of a response returned by IP2Proxy is shown below:

```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <IP2ProxyResponse xmlns="http://ws.fraudlabs.com/">
      <IP2ProxyResult>
        <IPADDRESS>68.142.197.65</IPADDRESS>
        <ANONYMOUSPROXY>NO</ANONYMOUSPROXY>
        <CREDITSAVAILABLE>79</CREDITSAVAILABLE>
        <MESSAGE />
      </IP2ProxyResult>
    </IP2ProxyResponse>
  </soap:Body>
</soap:Envelope>
```

2.5.3. Request by HTTP-GET

A valid request posted by user:

```
GET /ip2proxywebservice.asmx/IP2Proxy?IP=68.142.197.65&LICENSE=XX-XXXX-
XXXX HTTP/1.1
Host: ws.fraudlabs.com
```

2.5.4. Response by HTTP-GET

An example of a response returned by IP2Proxy is shown below:

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<IP2PROXY xmlns="http://ws.fraudlabs.com/">
  <IPADDRESS>68.142.197.65</IPADDRESS>
  <ANONYMOUSPROXY>NO</ANONYMOUSPROXY>
  <CREDITSAVAILABLE>79</CREDITSAVAILABLE>
  <MESSAGE />
</IP2PROXY>
```

2.5.5. Request by HTTP-POST

A valid request posted by user:

```
POST /ip2proxywebservice.asmx/IP2Proxy HTTP/1.1
Host: ws.fraudlabs.com
Content-Type: application/x-www-form-urlencoded
Content-Length: length
```

IP=68.142.197.65&LICENSE=XX-XXXX-XXXX

2.5.6. Response by HTTP-POST

An example of a response returned by IP2Proxy is shown below:

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<IP2PROXY xmlns="http://ws.fraudlabs.com/">
  <IPADDRESS>68.142.197.65</COUNTRYMATCH>
  <ANONYMOUSPROXY>NO</ANONYMOUSPROXY>
  <CREDITSAVAILABLE>79</CREDITSAVAILABLE>
  <MESSAGE />
</IP2PROXY>
```

3. Design Information

This section provides suggestion regarding the placing of IP2Proxy Web Service into Web pages. Look over this section to get a general idea for adding this service to your website.

3.1 Placement of IP2Proxy Web Service

As IP2Proxy web service provides intensive proxy checking and provides you a very easy and quick description, this service can be placed on ordering form or transaction form on your website, offering a seamless integration.

Description: Proxy checking is when visitor attempts to access to certain websites, especially when it involves online transactions.

Level of security: High

Site Type(s):

- e-Commerce Solutions
- Internet Retailers
- Online Credit Card Merchants
- Web Analytics
- e-Commerce Software Developers

Benefits:

- Protect from IP address spoofing
- Verify whether the online buyer is behind anonymous proxy servers
- Prevent and reduce fraud by authenticating online visitors
- reduces chargeback

For detecting online fraud, we strongly recommend using the FraudLabs Web Service since that it examines an online transaction from various angles with the integration of full and critical modules, and provides fraud score as a guideline.

4. Appendix I : ISO3166 Country Code

This table lists all valid ISO3166 two characters country codes that returns from component API query and describe the country names behind these country codes.

Country Code	Country Name
AD	ANDORRA
AE	UNITED ARAB EMIRATES
AF	AFGHANISTAN
AG	ANTIGUA AND BARBUDA
AI	ANGUILLA
AL	ALBANIA
AM	ARMENIA
AN	NETHERLANDS ANTILLES
AO	ANGOLA
AP	ASIA PACIFIC
AQ	ANTARCTICA
AR	ARGENTINA
AS	AMERICAN SAMOA
AT	AUSTRIA
AU	AUSTRALIA
AW	ARUBA
AZ	AZERBAIJAN
BA	BOSNIA AND HERZEGOWINA
BB	BARBADOS
BD	BANGLADESH
BE	BELGIUM
BF	BURKINA FASO
BG	BULGARIA
BH	BAHRAIN
BI	BURUNDI
BJ	BENIN
BM	BERMUDA
BN	BRUNEI DARUSSALAM
BO	BOLIVIA
BR	BRAZIL
BS	BAHAMAS
BT	BHUTAN
BV	BOUVET ISLAND
BW	BOTSWANA
BY	BELARUS
BZ	BELIZE
CA	CANADA
CC	COCOS (KEELING) ISLANDS
CD	CONGO, THE DEMOCRATIC REPUBLIC OF THE
CF	CENTRAL AFRICAN REPUBLIC
CG	CONGO

Country Code	Country Code
CH	SWITZERLAND
CI	COTE D'IVOIRE
CK	COOK ISLANDS
CL	CHILE
CM	CAMEROON
CN	CHINA
CO	COLOMBIA
CR	COSTA RICA
CS	CZECHOSLOVAKIA (FORMER)
CU	CUBA
CV	CAPE VERDE
CX	CHRISTMAS ISLAND
CY	CYPRUS
CZ	CZECH REPUBLIC
DE	GERMANY
DJ	DJIBOUTI
DK	DENMARK
DM	DOMINICA
DO	DOMINICAN REPUBLIC
DZ	ALGERIA
EC	ECUADOR
EE	ESTONIA
EG	EGYPT
EH	WESTERN SAHARA
ER	ERITREA
ES	SPAIN
ET	ETHIOPIA
EU	EUROPEAN UNION
FI	FINLAND
FJ	FIJI
FK	FALKLAND ISLANDS (MALVINAS)
FM	MICRONESIA, FEDERATED STATES OF
FO	FAROE ISLANDS
FR	FRANCE
FX	FRANCE, METROPOLITAN
GA	GABON
GB	GREAT BRITAIN
GD	GRENADA
GE	GEORGIA
GF	FRENCH GUIANA
GH	GHANA
GI	GIBRALTAR
GL	GREENLAND
GM	GAMBIA
GN	GUINEA
GP	GUADELOUPE
GQ	EQUATORIAL GUINEA
GR	GREECE

Country Code	Country Code
GS	SOUTH GEORGIA & SOUTH SANDWICH ISLANDS
GT	GUATEMALA
GU	GUAM
GW	GUINEA-BISSAU
GY	GUYANA
HK	HONG KONG
HM	HEARD ISLAND AND MCDONALD ISLANDS
HN	HONDURAS
HR	CROATIA
HT	HAITI
HU	HUNGARY
ID	INDONESIA
IE	IRELAND
IL	ISRAEL
IN	INDIA
IO	BRITISH INDIAN OCEAN TERRITORY
IQ	IRAQ
IR	IRAN, ISLAMIC REPUBLIC OF
IS	ICELAND
IT	ITALY
JM	JAMAICA
JO	JORDAN
JP	JAPAN
KE	KENYA
KG	KYRGYZSTAN
KH	CAMBODIA
KI	KIRIBATI
KM	COMOROS
KN	SAINT KITTS AND NEVIS
KP	KOREA, DEMOCRATIC PEOPLE'S REPUBLIC OF
KR	KOREA, REPUBLIC OF
KW	KUWAIT
KY	CAYMAN ISLANDS
KZ	KAZAKSTAN
LA	LAO PEOPLE'S DEMOCRATIC REPUBLIC
LB	LEBANON
LC	SAINT LUCIA
LI	LIECHTENSTEIN
LK	SRI LANKA
LR	LIBERIA
LS	LESOTHO
LT	LITHUANIA
LU	LUXEMBOURG
LV	LATVIA
LY	LIBYAN ARAB JAMAHIRIYA
MA	MOROCCO
MC	MONACO
MD	MOLDOVA, REPUBLIC OF

Country Code	Country Code
MG	MADAGASCAR
MH	MARSHALL ISLANDS
MK	MACEDONIA, THE FORMER YUGOSLAV
ML	MALI
MM	MYANMAR
MN	MONGOLIA
MO	MACAU
MP	NORTHERN MARIANA ISLANDS
MQ	MARTINIQUE
MR	MAURITANIA
MS	MONTSERRAT
MT	MALTA
MU	MAURITIUS
MV	MALDIVES
MW	MALAWI
MX	MEXICO
MY	MALAYSIA
MZ	MOZAMBIQUE
NA	NAMIBIA
NC	NEW CALEDONIA
NE	NIGER
NF	NORFOLK ISLAND
NG	NIGERIA
NI	NICARAGUA
NL	NETHERLANDS
NO	NORWAY
NP	NEPAL
NR	NAURU
NU	NIUE
NZ	NEW ZEALAND
OM	OMAN
PA	PANAMA
PE	PERU
PF	FRENCH POLYNESIA
PG	PAPUA NEW GUINEA
PH	PHILIPPINES
PK	PAKISTAN
PL	POLAND
PM	SAINT PIERRE AND MIQUELON
PN	PITCAIRN
PR	PUERTO RICO
PS	PALESTINIAN TERRITORY, OCCUPIED
PT	PORTUGAL
PW	PALAU
PY	PARAGUAY
QA	QATAR
RE	REUNION
RO	ROMANIA

Country Code	Country Code
RU	RUSSIAN FEDERATION
RW	RWANDA
SA	SAUDI ARABIA
SB	SOLOMON ISLANDS
SC	SEYCHELLES
SD	SUDAN
SE	SWEDEN
SG	SINGAPORE
SH	SAINT HELENA
SI	SLOVENIA
SJ	SVALBARD AND JAN MAYEN
SK	SLOVAKIA
SL	SIERRA LEONE
SM	SAN MARINO
SN	SENEGAL
SO	SOMALIA
SR	SURINAME
ST	SAO TOME AND PRINCIPE
SU	RUSSIAN FEDERATION
SV	EL SALVADOR
SY	SYRIAN ARAB REPUBLIC
SZ	SWAZILAND
TC	TURKS AND CAICOS ISLANDS
TD	CHAD
TF	FRENCH SOUTHERN TERRITORIES
TG	TOGO
TH	THAILAND
TJ	TAJIKISTAN
TK	TOKELAU
TM	TURKMENISTAN
TN	TUNISIA
TO	TONGA
TP	EAST TIMOR
TR	TURKEY
TT	TRINIDAD AND TOBAGO
TV	TUVALU
TW	TAIWAN, PROVINCE OF CHINA
TZ	TANZANIA, UNITED REPUBLIC OF
UA	UKRAINE
UG	UGANDA
UK	UNITED KINGDOM
UM	UNITED STATES MINOR OUTLYING ISLANDS
US	UNITED STATES
UY	URUGUAY
UZ	UZBEKISTAN
VA	HOLY SEE (VATICAN CITY STATE)
VC	SAINT VINCENT AND THE GRENADINES
VE	VENEZUELA

Country Code	Country Code
VG	VIRGIN ISLANDS, BRITISH
VI	VIRGIN ISLANDS, U.S.
VN	VIET NAM
VU	VANUATU
WF	WALLIS AND FUTUNA
WS	SAMOA
YE	YEMEN
YT	MAYOTTE
YU	YUGOSLAVIA
ZA	SOUTH AFRICA
ZM	ZAMBIA
ZW	ZIMBABWE

5. Appendix II : Sample Code

IP2Proxy Web Service sample code is available in several different programming languages. Below are some examples, for more different programming languages please log on to:

<http://www.fraudlabs.com/ip2proxysamplecodes.aspx>

i. ASP.NET – VB.NET (SOAP)

```
Private Sub IP2ProxyWebService()  
    Dim x_IP2Proxy As New IP2ProxyWebService.IP2ProxyWebService  
    Dim oIP2Proxy As New IP2PROXY  
    Try  
        oIP2Proxy = x_IP2Proxy.IP2Proxy( _  
            Me.txtIP.Text, Me.txtLicense.Text)  
  
        Me.txtResult.Text = "IP Address:" & oIP2Proxy.IPADDRESS &  
vbNewLine  
        Me.txtResult.Text += " Anonymous Proxy:" & oIP2Proxy.  
ANONYMOUSPROXY & vbNewLine  
        Me.txtResult.Text += "Credits Available:" &  
oIP2Proxy.CREDITSAVAILABLE & vbNewLine  
        Me.txtResult.Text += "Message:" & oIP2Proxy.MESSAGE &  
vbNewLine  
        Catch ex As Exception  
            Response.Write(ex.Message)  
        End Try  
    End Sub
```

ii. ASP.NET – VB.NET (HTML)

```
Private Sub Lookup()  
    Dim x_IP2Proxy As New IP2ProxyWebService.IP2Proxy  
    Dim oIP2Proxy As New IP2PROXY  
    Try  
        ' create the request URL  
        Dim x_builder As String  
        ' add the host element of the URL  
        x_builder =  
"http://ws.fraudlabs.com/IP2ProxyWebService.asmx/IP2Proxy?" & _  
            "IP=" & Me.txtIP.Text & _  
            "&LICENSE=" & Me.txtLicense.Text  
        ' create the web client and obtain the response data  
        ' as a byte array  
        Dim x_web_client As WebClient = New WebClient  
        Dim x_response() As Byte =  
x_web_client.DownloadData(x_builder.ToString())  
        ' process the string result to obtain a validation result  
        processResultString(Encoding.Default.GetString(x_response))  
        Catch ex As Exception  
            Response.Write(ex.Message)  
        End Try
```

```
End Sub

Private Sub processResultString(ByVal p_result As String)
Dim indexOpen As Integer
Dim indexClose As Integer
Dim strParam As String
    Me.txtResult.Text = p_result

    strParam = "<IPADDRESS>"
    indexOpen = p_result.IndexOf(strParam) + strParam.Length
    indexClose = p_result.IndexOf("</IPADDRESS>")
    Me.txtResult.Text = "IP Address:"
    If indexClose <> -1 Then
        Me.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen)
    End If

    strParam = "<ANONYMOUSPROXY>"
    indexOpen = p_result.IndexOf(strParam) + strParam.Length
    indexClose = p_result.IndexOf("</ANONYMOUSPROXY>")
    Me.txtResult.Text += vbNewLine & "Anonymous Proxy:"
    If indexClose <> -1 Then
        Me.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen)
    End If

    strParam = "<CREDITSAVAILABLE>"
    indexOpen = p_result.IndexOf(strParam) + strParam.Length
    indexClose = p_result.IndexOf("</CREDITSAVAILABLE>")
    Me.txtResult.Text += vbNewLine & "Credits Available:"
    If indexClose <> -1 Then
        Me.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen)
    End If

    strParam = "<MESSAGE>"
    indexOpen = p_result.IndexOf(strParam) + strParam.Length
    indexClose = p_result.IndexOf("</MESSAGE>")
    Me.txtResult.Text += vbNewLine & "Message:"
    If indexClose <> -1 Then
        Me.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen)
    End If
End Sub
```

iii. ASP.NET – C#.NET (SOAP)

```
private void IP2ProxyWebService()
{
    IP2ProxyWebService x_IP2Proxy = new IP2ProxyWebService();
    IP2PROXY oIP2Proxy = new IP2PROXY();
    try
    {
```

```

        oIP2Proxy = x_IP2Proxy.IP2Proxy(
            this.txtIP.Text,
            this.txtLicense.Text);
        this.txtResult.Text = "IP Address:" + oIP2Proxy.IPADDRESS + "\n";
        this.txtResult.Text += " Anonymous Proxy:" + oIP2 ANONYMOUSPROXY
+ "\n";
        this.txtResult.Text += "Credits Available:" +
oIP2Proxy.CREDITSAVAILABLE + "\n";
        this.txtResult.Text += "Message:" + oIP2Proxy.MESSAGE + "\n";
    }

    catch (Exception ex)
    {
        Response.Write(ex.Message);
    }
}

```

iv. ASP.NET – C#.NET (HTML)

```

private void Lookup()
{
    IP2ProxyWebService.IP2ProxyWebService x_IP2Proxy = new
IP2ProxyWebService.IP2ProxyWebService();
    IP2PROXY oIP2Proxy = new IP2PROXY();
    try
    {
        //create the request URL
        string x_builder;
        //add the host element of the URL
        x_builder =
"http://ws.fraudlabs.com/IP2ProxyWebService.aspx/IP2Proxy?";
        x_builder += "IP=" + this.txtIP.Text;
        x_builder += "&LICENSE=" + this.txtLicense.Text;
        // create the web client and obtain the response data
        // as a byte array
        WebClient x_web_client = new WebClient();
        byte [] x_response =
x_web_client.DownloadData(x_builder.ToString());
        // process the string result to obtain a validation result

        processResultString(Encoding.Default.GetString(x_response));
    }
    catch (Exception ex)
    {
        Response.Write(ex.Message);
    }
}

private void processResultString(String p_result)
{
    int indexOpen;
    int indexClose;
    string strParam;

    strParam = "<IPADDRESS>";
}

```

```

    indexOpen = p_result.IndexOf(strParam) + strParam.Length;
    indexClose = p_result.IndexOf("</IPADDRESS>");
    this.txtResult.Text = "IP Address:";
    if (indexClose != -1)
    {
        this.txtResult.Text += p_result.Substring(indexOpen,
indexClose - indexOpen);
    }

    strParam = "<ANONYMOUSPROXY>";
    indexOpen = p_result.IndexOf(strParam) + strParam.Length;
    indexClose = p_result.IndexOf("</ANONYMOUSPROXY>");
    this.txtResult.Text += "\rAnonymous Proxy:";
    if (indexClose != -1)
    {
        this.txtResult.Text += p_result.Substring(indexOpen,
indexClose - indexOpen);
    }

    strParam = "<CREDITSAVAILABLE>";
    indexOpen = p_result.IndexOf(strParam) + strParam.Length;
    indexClose = p_result.IndexOf("</CREDITSAVAILABLE>");
    this.txtResult.Text += "\rCredits Available:";
    if (indexClose != -1)
    {
        this.txtResult.Text += p_result.Substring(indexOpen,
indexClose - indexOpen);
    }

    strParam = "<MESSAGE>";
    indexOpen = p_result.IndexOf(strParam) + strParam.Length;
    indexClose = p_result.IndexOf("</MESSAGE>");
    this.txtResult.Text += "\rMessage:";
    if (indexClose != -1)
    {
        this.txtResult.Text += p_result.Substring(indexOpen,
indexClose - indexOpen);
    }
}

```

v. ASP 3.0

```

<%
Sub sendRequest(strip, strLICENSE)

dim SoapBody

if((strIP <> "") or (strLICENSE <> "")) then
SoapBody = xmlSoap(strIP, strLICENSE)
end if

<table border="0" cellspacing="1" cellpadding="3" rules="rows">
<%
if SoapBody = "" then
%><tr><td><b>Empty Soap Body Response</b></td></tr><%
else

```

```
dim xml
on error resume next
set xml = Server.CreateObject("Microsoft.XMLDOM")
    if(err.number = 0) then
        xml.async = False
        xml.loadxml(SoapBody)
        dim oNode : set oNode =
xml.selectSingleNode("soap:Envelope/soap:Body/" & SoapAction &
"Response/" & SoapAction & "Result")
            if
(oNode.selectSingleNode("Error").nodeTypedValue = "") then
%>
<tr>
<th colspan="2" align="left"><b>Result</b></th>
</tr>
<% if(TypeName(oNode.selectSingleNode("IPADDRESS")) = "IXMLDOMElement")
then %>
<tr>
<td align="left">IP ADDRESS:</td>
<td><%=oNode.selectSingleNode("IPADDRESS").nodeTypedValue%></td>
</tr>
<% end if %>
<% if(TypeName(oNode.selectSingleNode("ANONYMOUSPROXY")) =
"IXMLDOMElement") then %>
<tr>
<td align="left">ANONYMOUS PROXY:</td>
<td><%=oNode.selectSingleNode("ANONYMOUSPROXY").nodeTypedValue%></td>
</tr>
<% end if %>
<% if(TypeName(oNode.selectSingleNode("CREDITSAVAILABLE")) =
"IXMLDOMElement") then %>
<tr>
<td align="left">CREDITS AVAILABLE:</td>
<td><%=oNode.selectSingleNode("CREDITSAVAILABLE").nodeTypedValue%></td>
</tr>
<% end if %>
<% if(TypeName(oNode.selectSingleNode("MESSAGE")) = "IXMLDOMElement")
then %>
<tr>
<td align="left">MESSAGE:</td>
<td><%=oNode.selectSingleNode("MESSAGE").nodeTypedValue%></td>
</tr>
<% end if %>
<%
else ' error element
%>
<tr>
<td align="left" colspan="2"><b>Error</b></td>
</tr>
<tr>
<td align="left">Description:</td>
<td><%=oNode.selectSingleNode("MESSAGE").nodeTypedValue%></td>
</tr>
<%
end if 'error element does not exist
set xml = nothing
else
```

```
%>
<tr>
<td align="left">This objects requires Microsofts XML Parser 3.0 SP2 or
greater.</td>
</tr>
<tr>
<td>Download here: <a
href="http://download.microsoft.com/download/xml/SP/3.20/W9X2KMeXP/EN-
US/msxml3sp2Setup.exe">http://download.microsoft.com/download/xml/SP/3.
20/W9X2KMeXP/EN-US/msxml3sp2Setup.exe</a></td>
</tr>
<%
    Response.Write("Error: " & err.number)
    end if 'DOM object is valid
    end if 'soapbody is <> "" empty string
%>

<%
function xmlSoap(strIP, strLICENSE)

' Instantiate objects to hold the XML DOM and the HTTP/XML
communication:
' Instantiate object for HTTP/XML communication:

Dim xmlhttp, strSoap
Set xmlhttp = Server.CreateObject("Msxml2.ServerXMLHTTP")

' Build XML document:

strSoap = "<?xml version=""1.0"" encoding=""utf-8""?>" & _
"<soap:Envelope xmlns:xsi=""http://www.w3.org/2001/XMLSchema-instance""
xmlns:xsd=""http://www.w3.org/2001/XMLSchema""
xmlns:soap=""http://schemas.xmlsoap.org/soap/envelope/">" & _
"<soap:Body>" & _
"<" & SoapAction & " xmlns="" & SoapNamespace & "">" & _
"    <IP>" & strIP & "</IP>" & _
"    <LICENSE>" & strLICENSE & "</LICENSE>" & _
"  </soap:Body>" & _
"</soap:Envelope>"

'Build custom HTTP header:
xmlhttp.Open "POST", "http://" & SoapServer & SoapPath, False
' False = Do not respond immediately
xmlhttp.setRequestHeader "Man", "POST " & SoapPath & " HTTP/1.1"
xmlhttp.setRequestHeader "Host", SoapServer
xmlhttp.setRequestHeader "Content-Type", "text/xml; charset=utf-
8"

xmlhttp.setRequestHeader "SOAPAction", SoapNamespace & SoapAction

'Send it using the header generated above:
xmlhttp.send(strSoap)
if xmlhttp.Status = 200 then
'Response from server was success
xmlSoap = xmlhttp.responseText
'note xmlSoap is the function name hence the return value is a
string Variant
```

```

else
  ' Response from server failed

xmlSoap = ""
  ' Tell administrator what went wrong - maybe not users though
Response.Write("Server Error.<br>")
Response.Write("status = " & xmlhttp.status)
Response.Write("<br>" & xmlhttp.statusText & "<br>" &
xmlhttp.responseText)
  Response.Write("<br><pre>" & Request.ServerVariables("ALL_HTTP")
& "</pre>")
  end If
  Set xmlhttp = nothing
end function
%>

```

vi. PHP

```

<?php

// Get your own license key from http://www.fraudlabs.com
$license = "<Enter license key>";

// Replace these parameters to your values
$ip = "68.142.197.65";

require_once('./lib/nusoap.php');

// Call the service with this message
$msg = '<IP2Proxy xmlns="http://ws.fraudlabs.com/">
<IP>'.$ip.'</IP>
<LICENSE>'.$license.'</LICENSE>
</IP2Proxy>';

// Create a new SOAP object
$soapclient = new
soapclient('http://ws.fraudlabs.com/ip2proxywebservice.asmx?wsdl','wsdl
');
if (!$soapclient->fault) {

$result = $soapclient->call('IP2Proxy', array($msg));
if(!isset($result['Error'])) {
// No Error. Retrieve results.
echo "IPADDRESS = " . $result["IPADDRESS"] . "<br>";
echo "ANONYMOUSPROXY = " . $result["ANONYMOUSPROXY"] . "<br>";
echo "CREDITSAVAILABLE = " . $result["CREDITSAVAILABLE"] . "<br>";
echo "MESSAGE = " . $result["MESSAGE"] . "<br>";
}
else {
// Error
echo "Error Description = " . $result["Error"]["Desc"] . "<br>";
echo "Error Number = " . $result["Error"]["Number"] . "<br>";
}
}
else {
echo "Error = {$soapclient->faultcode}<br>";
}
}

```

```
    echo "String = {$soapclient->faultstring}";  
    } // end if $soapclient->call is success  
?>
```

vii. JAVA / APACHE

```
import java.io.*;  
  
//Web Service Client Class  
public class Client  
{  
    //Entry Point to this Application  
    public static void main(String[] args)  
    {  
        try  
        {  
            //Create a proxy  
            ApacheSoapProxy proxy = new ApacheSoapProxy ();  
  
            //Invoke IP2ProxyCheck() over SOAP and get the new OID  
            //Require Parameter IP2ProxyCheck(String IP, String LICENSE)  
  
            String result = proxy.IP2Proxy("68.142.197.65", "<Enter License  
Key>");  
  
            //Print out the value  
            System.out.println (result);  
        }  
        catch (java.net.MalformedURLException exception)  
        {  
            exception.printStackTrace ();  
        }  
        catch (org.apache.soap.SOAPException exception)  
        {  
            exception.printStackTrace ();  
        }  
    }  
}
```