

Forum:[Hack Forums](#)

Thread: [Sock5 Class for VB.NET](#)

Dark N3BULA - December 15, 2014, 4:02 pm

Code Imports System.Collections.Generic

Imports System.Net

Imports System.Net.Sockets

Imports System.Runtime.Serialization

Imports System.Text

Namespace Socks5Client

Public Class Socks5Client

Private _socksAddr As String

Private _socksPort As Integer

Private _destAddr As String

Private _destPort As Integer

Private _username As String

Private _password As String

Private _socket As Socket

Private Const SOCKS_VER As Integer = &H5

Private Const AUTH_METH_SUPPORT As Integer = &H2

Private Const USER_PASS_AUTH As Integer = &H2

Private Const NOAUTH As Integer = &H0

Private Const CMD_CONNECT As Integer = &H1

Private Const SOCKS_ADDR_TYPE_IPV4 As Integer = &H1

Private Const SOCKS_ADDR_TYPE_IPV6 As Integer = &H4

Private Const SOCKS_ADDR_TYPE_DOMAIN_NAME As Integer = &H3

Private Const AUTH_METHOD_NOT_SUPPORTED As Integer = &HFF

Private Const SOCKS_CMD_SUCCESS As Integer = &H0

Private Sub New(socksAddress As String, socksPort As Integer, destAddress As String,
destPort As Integer, username As String, password As String)

_socksAddr = socksAddress

_socksPort = socksPort

_destAddr = destAddress

_destPort = destPort

_username = username

_password = password

_socket = New Socket(AddressFamily.InterNetwork, SocketType.Stream,
ProtocolType.Tcp)

End Sub

Public Function Connect() As Socket

_socket.Connect(_socksAddr, _socksPort)

Dim buffer As Byte() = New Byte(3) {}

buffer(0) = SOCKS_VER

buffer(1) = AUTH_METH_SUPPORT

buffer(2) = NOAUTH

buffer(3) = USER_PASS_AUTH

```

_socket.Send(buffer)
_socket.Receive(buffer, 0, 2, SocketFlags.None)
If buffer(1) = AUTH_METHOD_NOT_SUPPORTED Then
    _socket.Close()
    Throw New SocksAuthException()
End If

```

```

If buffer(1) = USER_PASS_AUTH AndAlso (_username Is Nothing OrElse _password Is
Nothing) Then
    Throw New ArgumentException("No username or password provided")
End If

```

```

If buffer(1) = USER_PASS_AUTH Then
    Dim credentials As Byte() = New Byte(_username.Length + _password.Length + 2) {}
    credentials(0) = 1
    credentials(1) = CByte(_username.Length)
    Encoding.ASCII.GetBytes(_username).CopyTo(credentials, 2)
    credentials(_username.Length + 2) = CByte(_password.Length)
    Encoding.ASCII.GetBytes(_password).CopyTo(credentials, _username.Length + 3)

```

```

_socket.Send(credentials, credentials.Length, SocketFlags.None)
buffer = New Byte(1) {}
_socket.Receive(buffer, buffer.Length, SocketFlags.None)
If buffer(1) <> SOCKS_CMD_SUCCSESS Then
    Throw New SocksRefuseException("Username or password invalid")
End If
End If

```

```

Dim addrType As Byte = GetAddressType()
Dim address As Byte() = GetDestAddressBytes(addrType, _destAddr)
Dim port As Byte() = GetDestPortBytes(_destPort)
buffer = New Byte(4 + port.Length + (address.Length - 1)) {}
buffer(0) = SOCKS_VER
buffer(1) = CMD_CONNECT
buffer(2) = &H0
'reserved
buffer(3) = addrType
address.CopyTo(buffer, 4)
port.CopyTo(buffer, 4 + address.Length)
_socket.Send(buffer)
buffer = New Byte(254) {}
_socket.Receive(buffer, buffer.Length, SocketFlags.None)

```

```

If buffer(1) = SOCKS_CMD_SUCCSESS Then
    Return _socket
End If
Throw New SocksRefuseException()
End Function

```

```

Private Function GetAddressType() As Byte
    Dim ipAddr As IPAddress

```

```
Dim result As Boolean = IPAddress.TryParse(_destAddr, ipAddr)
```

```
If Not result Then
```

```
    Return SOCKS_ADDR_TYPE_DOMAIN_NAME
```

```
End If
```

```
Select Case ipAddr.AddressFamily
```

```
    Case AddressFamily.InterNetwork
```

```
        Return SOCKS_ADDR_TYPE_IPV4
```

```
    Case AddressFamily.InterNetworkV6
```

```
        Return SOCKS_ADDR_TYPE_IPV6
```

```
    Case Else
```

```
        Throw New BadDistanationAddrException()
```

```
End Select
```

```
End Function
```

```
Private Function GetDestAddressBytes(addressType As Byte, host As String) As Byte()
```

```
    Select Case addressType
```

```
        Case SOCKS_ADDR_TYPE_IPV4, SOCKS_ADDR_TYPE_IPV6
```

```
            Return IPAddress.Parse(host).GetAddressBytes()
```

```
        Case SOCKS_ADDR_TYPE_DOMAIN_NAME
```

```
            Dim bytes As Byte() = New Byte(host.Length) {}
```

```
            bytes(0) = Convert.ToByte(host.Length)
```

```
            Encoding.ASCII.GetBytes(host).CopyTo(bytes, 1)
```

```
            Return bytes
```

```
        Case Else
```

```
            Return Nothing
```

```
    End Select
```

```
End Function
```

```
Private Function GetDestPortBytes(value As Integer) As Byte()
```

```
    Dim array As Byte() = New Byte(1) {}
```

```
    array(0) = Convert.ToByte(value / 256)
```

```
    array(1) = Convert.ToByte(value Mod 256)
```

```
    Return array
```

```
End Function
```

```
Public Shared Function Connect(socksAddress As String, socksPort As Integer, destAddress  
As String, destPort As Integer, username As String, password As String) As Socket
```

```
    Dim client As New Socks5Client(socksAddress, socksPort, destAddress, destPort,  
username, password)
```

```
    Return client.Connect()
```

```
End Function
```

```
End Class
```

```
<Serializable(> _
```

```
Public Class SocksAuthException
```

```
    Inherits Exception
```

```
    Public Sub New()
```

```
End Sub
```

```
Public Sub New(message As String)
```

```
    MyBase.New(message)
End Sub
```

```
Public Sub New(message As String, inner As Exception)
    MyBase.New(message, inner)
End Sub
```

```
Protected Sub New(info As SerializationInfo, context As StreamingContext)
    MyBase.New(info, context)
End Sub
End Class
```

```
<Serializable> _
Public Class BadDistanciationAddrException
    Inherits Exception
    Public Sub New()
End Sub
```

```
Public Sub New(message As String)
    MyBase.New(message)
End Sub
```

```
Public Sub New(message As String, inner As Exception)
    MyBase.New(message, inner)
End Sub
```

```
Protected Sub New(info As SerializationInfo, context As StreamingContext)
    MyBase.New(info, context)
End Sub
End Class
```

```
<Serializable> _
Public Class SocksRefuseException
    Inherits Exception
```

```
Public Sub New()
End Sub
```

```
Public Sub New(message As String)
    MyBase.New(message)
End Sub
```

```
Public Sub New(message As String, inner As Exception)
    MyBase.New(message, inner)
End Sub
```

```
Protected Sub New(info As SerializationInfo, context As StreamingContext)
    MyBase.New(info, context)
End Sub
```

```
End Class
End Namespace
Enjoy
```

Earn - December 16, 2014, 1:16 am

Haven't tested it yet, but this looks good. Nice work.

PapaBeans - December 16, 2014, 1:28 am

For those who do not know, Code Socket Secure (SOCKS) is an Internet protocol that routes network packets between a client and server through a proxy server.

samo dz - December 16, 2014, 4:27 am

this good packet but i have good packet some code of sock of njq8
gl

taktaktak - December 16, 2014, 1:02 pm

Is this yours? If not please include source to OP. The reason I ask is because in your GDI thread you talk about it being your first program.