

Emulando NO-IP DUC

- 1.-Introducción
- 2.-Código en c
- 3.-Código en vb
- 4.-Código en Delphi
- 5.-Encriptación del DUC (Incompleto)
- 6.-Despedida

1.-Introducción

En este pequeño artículo veremos como emular al NO-IP DUC en nuestros programas. Para el que no lo sepa No-IP es un DNS gratuito que resuelve nuestro nombre de dominio a la ip dinámica que tengamos en ese momento. Se puede decir que este artículo está escrito a medias por MazarD y Thor. Yo encontré las urls, lo programé en visual basic y C. Y Thor encontró un problema en las peticiones lo solucionó y lo programó en Delphi.

La idea me surgió al leer en el DUC que disponia de un sistema para detectar la ip pública, nada más lejos, lo único que hace es conectar a un servidor externo y éste le devuelve la ip desde la que se ha conectado, menudo sistema no? Como todos los troyanos desde el inicio de los tiempos. Hechándole un ojo al DUC con un debugger encontré las url que nos permiten averiguar la ip pública, listar los dominios de nuestra cuenta y actualizarlo. Thor detectó que cuando se hacen las peticiones al servidor php sólo con get algunas veces no nos devuelve el resultado.

El DUC envia las peticiones encriptadas, he intentado sacarla pero me ha sido imposible, aunque creo que me he quedado bastante cerca así que en el último punto explicaré lo que he conseguido hasta el momento por si alguien se anima a intentarlo.

Los hosts que nos devuelven nuestra ip pública son los siguientes:

<http://ip1.dynupdate.no-ip.com:8245/>

<http://ip2.dynupdate.no-ip.com:8245/>

Para listar los nombres de dominio de nuestra cuenta:

<http://dynupdate.no-ip.com:8245/list-hosts.php?email=Email&pass=Pass>

Para actualizar un nombre de dominio de nuestra cuenta a una ip:

[http://dynupdate.no-](http://dynupdate.no-ip.com:8245/ducupdate.php?username=Email&pass=Pass&h[]=Dom&ip=Ip)

[ip.com:8245/ducupdate.php?username=Email&pass=Pass&h\[\]=Dom&ip=Ip](http://dynupdate.no-ip.com:8245/ducupdate.php?username=Email&pass=Pass&h[]=Dom&ip=Ip)

Para actualizar un grupo de dominios:

[http://dynupdate.no-](http://dynupdate.no-ip.com:8245/ducupdate.php?username=Email&pass=Pass&g[]=Grupo&ip=Ip)

[ip.com:8245/ducupdate.php?username=Email&pass=Pass&g\[\]=Grupo&ip=Ip](http://dynupdate.no-ip.com:8245/ducupdate.php?username=Email&pass=Pass&g[]=Grupo&ip=Ip)

P

En general si nos responde con un "status=3" el username o password son incorrectos si es correcto nos devolverá lo que hemos pedido o "dominio:1" en el caso de que se haya actualizado correctamente. Lo mejor es que introduzcas estas peticiones en el navegador y veas cómo funciona el tema.

Al automatizar el proceso deberemos hacer las peticiones del siguiente modo para que siempre devuelva el resultado:

```
GET / HTTP/1.0
Accept: */*
User-Agent: DUC v2.2.1
Host: Host al que consultemos
Pragma: no-cache
```

Y nos devolverá algo de este estilo:

```
HTTP/1.1 200 OK
Date: Mon, 13 Mar 2006 12:40:50 GMT
Server: Apache/1.3.27 (Unix) PHP/4.3.2
Connection: close
Content-Type: text/html
```

83.32.x.x

Ahora será sólo cuestión de parsear la salida y listo :)

2.-Código en c

```
//Emulador de No-ip DUC by MazarD
//MazarD@gmail.com
```

```
#include <stdio.h>
#include <windows.h>
#include <string.h>
```

```
#pragma comment(lib,"ws2_32.lib")
```

```
void Uso();
```

```
int main(int argc,char *argv[])
{
    WSADATA wsaData;
    SOCKET wSck;
    sockaddr_in Direccion;
    int Brec=1;
    char Buffer[30];
    struct hostent* ResIp;
    char HostNoip[24];
```

```

char Consulta[300];
if (argc<2 || argc>6) Uso();

switch (atoi(argv[1])) {
case 0:
    sprintf(HostNoip,"ip1.dynupdate.no-ip.com");
    //ip2.dynupdate.no-ip.com también se puede utilizar
    sprintf(Consulta,"GET /");
    break;
case 1:
    if (argc!=4) Uso();
    sprintf(HostNoip,"dynupdate.no-ip.com");
    sprintf(Consulta,"%s%s%s%s%s",
        "GET /list-hosts.php",
        "?email=",argv[2],
        "&pass=",argv[3]);
    break;
case 2:
    if (argc!=6) Uso();
    sprintf(HostNoip,"dynupdate.no-ip.com");
    sprintf(Consulta,"%s%s%s%s%s%s%s%s%s",
        "GET /ducupdate.php",
        "?username=",argv[2],
        "&pass=",argv[3],
        "&h[]=",argv[4],
        "&ip=",argv[5]
    );
    break;
default:
    Uso();
}
strcat(Consulta,"\n\r\n\r");
strcat(Consulta," HTTP/1.0\n\rAccept: */*\n\rUser-Agent: DUC
v2.2.1\n\rHost: ");
strcat(Consulta,HostNoip);
strcat(Consulta,"\n\rPragma: no-cache\n\r\n\r");

if (WSAStartup(MAKEWORD(2,2),&wsaData)!=NO_ERROR) {
    printf("[-]Error al inicializar el socket!");
    exit(-1);
}

if
((wSck=socket(AF_INET,SOCK_STREAM,IPPROTO_TCP))==INVALID_SOCKET)
{
    printf("[-]%s",WSAGetLastError());
    WSACleanup();
}

```

```

        exit(-1);
    }

    Direccion.sin_family=AF_INET;

    if ((ResIp=gethostbyname(HostNoip))==0) {
        printf("[-]No se puede resolver el nombre");
        WSACleanup();
        exit(-1);
    }

    Direccion.sin_addr=((struct in_addr *)ResIp->h_addr);
    Direccion.sin_port=htons(8245);

    if (connect(wSck,(SOCKADDR*)
&Direccion,sizeof(Direccion))==SOCKET_ERROR) {
        printf("[-]No se ha podido conectar\n");
        WSACleanup();
        exit(-1);
    }

    send(wSck,Consulta,strlen(Consulta),0);
    while (Brec!=SOCKET_ERROR) {
        Brec = recv(wSck, Buffer, sizeof(Buffer), 0 );
        printf(Buffer);
        if (Brec<=0) {
            printf("Cerrada la conexión\n");
            break;
        }
    }

    WSACleanup();
    return 0;
}

void Uso(void) {
    printf("\n[Emulador de No-Ip DUC by MazarD
(mazard@gmail.com)]\n");
    printf("Uso:\n");
    printf("NoIpMD 0                --Devuelve tu ip pública\n");
    printf("NoIpMD 1 <user> <pass>        --Devuelve la lista de
hosts de tu cuenta\n");
    printf("NoIpMD 2 <user> <pass> <host> <ip> --Actualiza el host a
la ip\n");
    printf("Ej:\nNoIpMD 2 mazard@gmail.com passnoip mazard.no-ip.org
66.66.66.66\n");
    exit(1);
}

```

3.-Código en Visual Basic

Esta funcion hace las consultas:

'Opcion=0 --> Consulta la ip pública
'Opcion=1 --> Consulta la lista de dominios de la cuenta no-ip
'Opcion=2 --> Actualiza el dominio a la ip
'-1 se ha producido un error de usuario
'-2 no se ha podido conectar con el host
' 1 se ha realizado la consulta correctamente

Public Function NoIpMD(ByRef wSck As Winsock, ByVal Opcion As Byte,
Optional Usuario As String, Optional Password As String, Optional Dominio As
String, Optional UpIp As String) As Integer

Dim Consulta As String
Dim rHost As String

Select Case Opcion

Case 0

rHost = "ip1.dynupdate.no-ip.com"
Consulta = "GET"

Case 1

If IsMissing(Usuario) Or IsMissing>Password) Then
NoIpMD = -1
Exit Function

End If

rHost = "dynupdate.no-ip.com"

Consulta = "GET /list-hosts.php?email=" & Usuario & "&pass=" &

Password

Case 2

If IsMissing(Usuario) Or IsMissing>Password) Or IsMissing(Dominio) Or
IsMissing(UpIp) Then

NoIpMD = -1

Exit Function

End If

rHost = "dynupdate.no-ip.com"

Consulta = "GET /ducupdate.php?username=" & Usuario & "&pass=" &

Password & "&h[]=" _

& Dominio & "&ip=" & UpIp

Case Else

NoIpMD = -1

Exit Function

End Select

Consulta = Consulta & " HTTP/1.0" & vbCrLf & "Accept: */*" & vbCrLf & "User-
Agent: DUC v2.2.1" _
& vbCrLf & "Host: " & rHost & vbCrLf & "Pragma: no-cache" & vbCrLf & vbCrLf

```
If wSck.State <> sckClosed Then wSck.Close  
wSck.Connect rHost, 8245
```

```
While wSck.State <> sckConnected And wSck.State <> sckError  
    DoEvents  
Wend  
If wSck.State = sckError Then  
    NoIpMD = -2  
    Exit Function  
End If
```

```
wSck.SendData Consulta  
NoIpMD = 1  
End Function
```

La función que introduciríamos en el Data_Arrival para parsear la salida:

```
Public Function ExtraerSalida(ByVal Buffer As String) As String  
    Dim pos As Long  
  
    pos = InStr(Buffer, vbCrLf & vbCrLf)  
    If pos = 0 Then  
        ExtraerSalida = ""  
    Else  
        ExtraerSalida = Mid$(Buffer, pos + 2, Len(Buffer) - pos + 1)  
    End If  
  
End Function
```

4.-Código en Delphi

En delphi vamos a usar 3 botones, los cuales realizan estas acciones:

- 1 - Obtiene una lista de dominios del email con el password dado.
- 2 - Obtiene nuestra IP pública
- 3 - Actualiza el dominio que hallamos elegido de los disponibles con nuestra IP pública.

En la Form tendremos:

- Tres EditText's, Edit1 para el email, Edit2 para el password y Edit3 para la IP pública.
- Un listbox donde añadiremos los dominios disponibles.
- Tres botones (uno para cada acción antes comentadas)
- Tres sockets, uno para cada boton.

Los codigos de los tres botones son respectivamente:

```
procedure TForm1.Button1Click(Sender: TObject);
```

```
begin
  ClientSocket1.Host:='ip1.dynupdate.no-ip.com';
  ClientSocket1.Port:=8245;
  ClientSocket1.Open;
end;
```

```
procedure TForm1.Button2Click(Sender: TObject);
begin
  ClientSocket2.Host:='dynupdate.no-ip.com';
  ClientSocket2.Port:=8245;
  ClientSocket2.Open;
end;
```

```
procedure TForm1.Button3Click(Sender: TObject);
begin
  ClientSocket3.Host:='dynupdate.no-ip.com';
  ClientSocket3.Port:=8245;
  ClientSocket3.Open;
end;
```

En el evento OnConnect de estos sockets enviaremos las peticiones GET antes vistas:

```
procedure TForm1.ClientSocket1Connect(Sender: TObject;
  Socket: TCustomWinSocket);
begin
  Socket.SendText('GET / HTTP/1.0'+#13#10+
    'Accept: */*'+#13#10+
    'User-Agent: DUC v2.2.1'+#13#10+
    'Host: '+ClientSocket1.Host+#13#10+
    'Pragma: no-cache'+#13#10#13#10);
end;
```

```
procedure TForm1.ClientSocket2Connect(Sender: TObject;
  Socket: TCustomWinSocket);
begin
  Socket.SendText('GET /list-
hosts.php?email='+Edit1.Text+'&pass='+Edit2.Text+' HTTP/1.0'+#13#10+
    'Accept: */*'+#13#10+
    'User-Agent: DUC v2.2.1'+#13#10+
    'Host: '+ClientSocket2.Host+#13#10+
    'Pragma: no-cache'+#13#10#13#10);
end;
```

```
procedure TForm1.ClientSocket3Connect(Sender: TObject;
  Socket: TCustomWinSocket);
begin
  Socket.SendText('GET
/ducupdate.php?username='+Edit1.Text+'&pass='+Edit2.Text+'&h[]='+
```

```

    ListBox1.Items[ListBox1.ItemIndex]+'&ip='+Edit3.Text+'
HTTP/1.0'+#13#10+
        'Accept: */*'+#13#10+
        'User-Agent: DUC v2.2.1'+#13#10+
        'Host: '+ClientSocket3.Host+#13#10+
        'Pragma: no-cache'+#13#10#13#10);
end;

```

Por último nos faltan los eventos OnRead de los sockets, donde trataremos la información que nos responde el servidor:

```

procedure TForm1.ClientSocket1Read(Sender: TObject;
    Socket: TCustomWinSocket);
var
    Texto, IP: String;
begin
    Texto:=Socket.ReceiveText;
    IP:=Copy(Texto, AnsiPos(#13#10#13#10, Texto)+4, Length(Texto)-
AnsiPos(#13#10#13#10, Texto)-3);
    Edit3.Text:=IP;
end;

procedure TForm1.ClientSocket2Read(Sender: TObject;
    Socket: TCustomWinSocket);
var
    Dominios: String;
begin
    Dominios:=Socket.ReceiveText;
    Dominios:=Copy(Dominios, AnsiPos(#13#10#13#10, Dominios)+4,
Length(Dominios)-AnsiPos(#13#10#13#10, Dominios)-3);
    While Dominios <>" do
        begin
            Dominios:=Copy(Dominios, 3, Length(Dominios)-2);
            ListBox1.Items.Add(copy(Dominios, 0, AnsiPos('|', Dominios)-1));
            Dominios:=Copy(Dominios, AnsiPos('|', Dominios)+1, Length(Dominios)-
AnsiPos('|', Dominios));
        end;
    end;
end;

```

El del tercer socket no nos interesa lo que nos responda el servidor ya que suponemos que se ha actualizado bien.

5.-Encriptación del DUC

Esto era lo que a mí me parecía mas interesante de todo el tema, pero no lo he podido resolver aunque me he quedado bastante cerca, supongo que mis conocimientos sobre ingeniería inversa són bastante justos :p. De todos modos pongo aquí lo que he sacado y si te ánimas ya tendrás bastante trabajo hecho.

La versión con la que he tratado es la 2.2.1 para otras los offsets cambiarán y puede que la encriptación entera.

La encriptación siempre se realiza sobre la petición entera, luego le añade un /request=Encriptado

Es decir si loggearnos directamente en nuestra cuenta no-ip el DUC lo que haría sería encriptar

email=Email&pass=Pass

y hacer la petición con

http://dynupdate.no-ip.com:8245/dns/?request=TextoEncriptado

Vamos a ello:

Inicialmente le dá la vuelta a nuestro login, por ejemplo

email=mazard@gmail.com&pass=NoIpPass quedaría:

ssaPpIoN=ssap&moc.liamg@drazam=liame

Además tenemos un vector tan largo como nuestro texto a encriptar con valores que desconozco como calcula(esto es lo que me ha faltado) y hace lo siguiente:

Coje el primer valor del vector, lo desplaza 2 bytes a la derecha le aplica un AND 3F y el resultado lo utiliza para decidir de la cadena

"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/"

que valor debe cojer para sustituir por el primer elemento del texto a encriptar.

El mismo valor lo desplaza dos bytes a la izquierda y le aplica un AND 30.

Coje el siguiente valor del vector.

Lo desplaza 4 bytes a la derecha y le aplica una mascara AND para quedarse con el byte de menor peso.

Al resultado de esto le aplica una OR con el último cálculo del valor anterior y vuelve al vector del alfabeto cojiendo la letra que le corresponde y sustituyendola por el segundo del texto a encriptar.

Ahora eleva al cubo el segundo valor del vector y le aplica una AND 3C.

Coje el tercer valor del vector.

Lo desplaza 6 bytes a la derecha y le aplica una AND 3, el resultado de esto con una OR con el último resultado del valor anterior, con esto vuelve a decidir con el vector del alfabeto que letra debe cojer.

Con el último resultado le aplica un and 3F y elige la última letra de las 4 que sustituirá del vector del alfabeto a nuestro texto a encriptar.

A partir de aquí el proceso se repite hasta terminar con la cadena.

Seguir esto puede resultar un poco complicado pero con el debugger se vé muy sencillo. El inicio del algoritmo de encriptación está en 0047070E y el vector que no consigo descubrir cómo se calcula en A32BE8 cuando se ha llegado al inicio de la encriptación.

6.-Despedida

Espero que te haya gustado el artículo, para dudas, sugerencias o cualquier cosa mandame un email a mazard en gmail punto com.

Para información más detallada sobre la implementación en delphi:

<http://horzum.no-ip.biz/usuarios/indetectables/foro/viewtopic.php?t=281>

Artículo para elhacker.net.

Por MazarD y Thor