

How Can I Monitor My Server's and Networked Device's Connection Status?

Overview.....	1
What is a Networked Device?.....	2
How do networks work?.....	2
Diagram of OSI Model.....	2
Diagram of Simple Network.....	3
What protocol is used for device status?.....	5
What is ICMP?.....	5
What is a Ping?.....	5
How can I use ICMP to conduct a Ping?.....	5
What is Ping Wizard?.....	6
Can you show me an example of how Ping Wizard works?.....	6
[Visual Basic].....	6
[VBScript].....	7
[Active Server Pages].....	8
[Visual C++].....	10
[Visual FoxPro].....	12
Where can I find more information about Ping Wizard?.....	14
Who is Seekford Solutions, Inc.?.....	14

Overview

A lot of time is spent designing and implementing network designs and setting up devices like computers, servers, TCP/IP connected equipment and more. Even with the best plans and architecture there is always the uncertainty of device failure or link failure. When you have distributed applications or a critical servers and devices, it becomes that much more important to have a method of monitoring the devices to see if they are online, alive, and reachable on the network. This article will explain a rough overview of how networks work, how devices are connected and how you can write software to monitor them actively.

What is a Networked Device?

A networked device can be anything from a mainframe server, a printer all the way to custom TCP/IP implementing devices that monitor pump operations and pressure integrity or even display and communication devices. With regards to this article, a Networked Device is any hardware that is logically connected to a network, and uses the TCP/IP communication protocol. This means web servers, personal computers and even your network enabled coffee machine are considered Networked Devices.

How do networks work?

A network is a set of interconnected devices that can all speak the same language. There are many different types of networks, Token-ring, IPX, AppleTalk, TCP/IP, and even proprietary communications used by some PLC's and other hardware. The scope of this article will cover TCP/IP networks, but the physical layout is the same for most types of networks. TCP/IP stands for Transmission Control protocol/Internet Protocol. The standard network is a combination of Ethernet and possibly wireless connected communication devices.

Every device that is connected to the network is considered a node on the network. A device must have a unique address that identifies it when communicating. This is where IP addresses come. When the device boots up, it either assumes a preset IP address or request an open address from a DHCP server. DHCP servers are useful because they assign open addresses to new devices. No two devices can have the same IP address on the network otherwise connection problem will occur.

Devices speak on the network using network interface hardware that has built-in logic for the actual physical communication. In an Ethernet network, communication takes place in a manner that is similar to the way a group of friends talk. Each device waits until the line is quiet, no one is transmitting, and then begins to transmit their own message. It is possible for two devices to start talking at the same time. This is detected by the transmitting devices because they listen to the data as it is sent and if their data is not what they think it should be, they stop transmitting and wait a random amount of time before checking and transmitting again. This random time is important because if they waited the same amount of time, an infinite loop would be created as each of the devices kept colliding.

Diagram of OSI Model

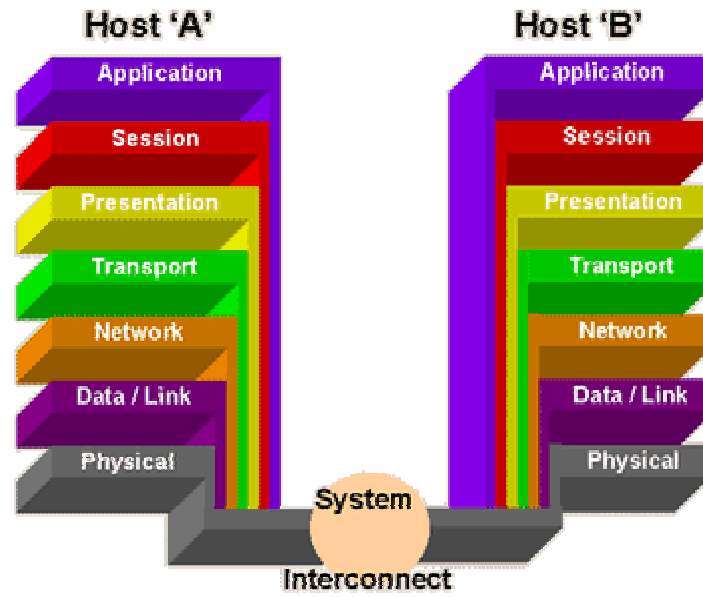
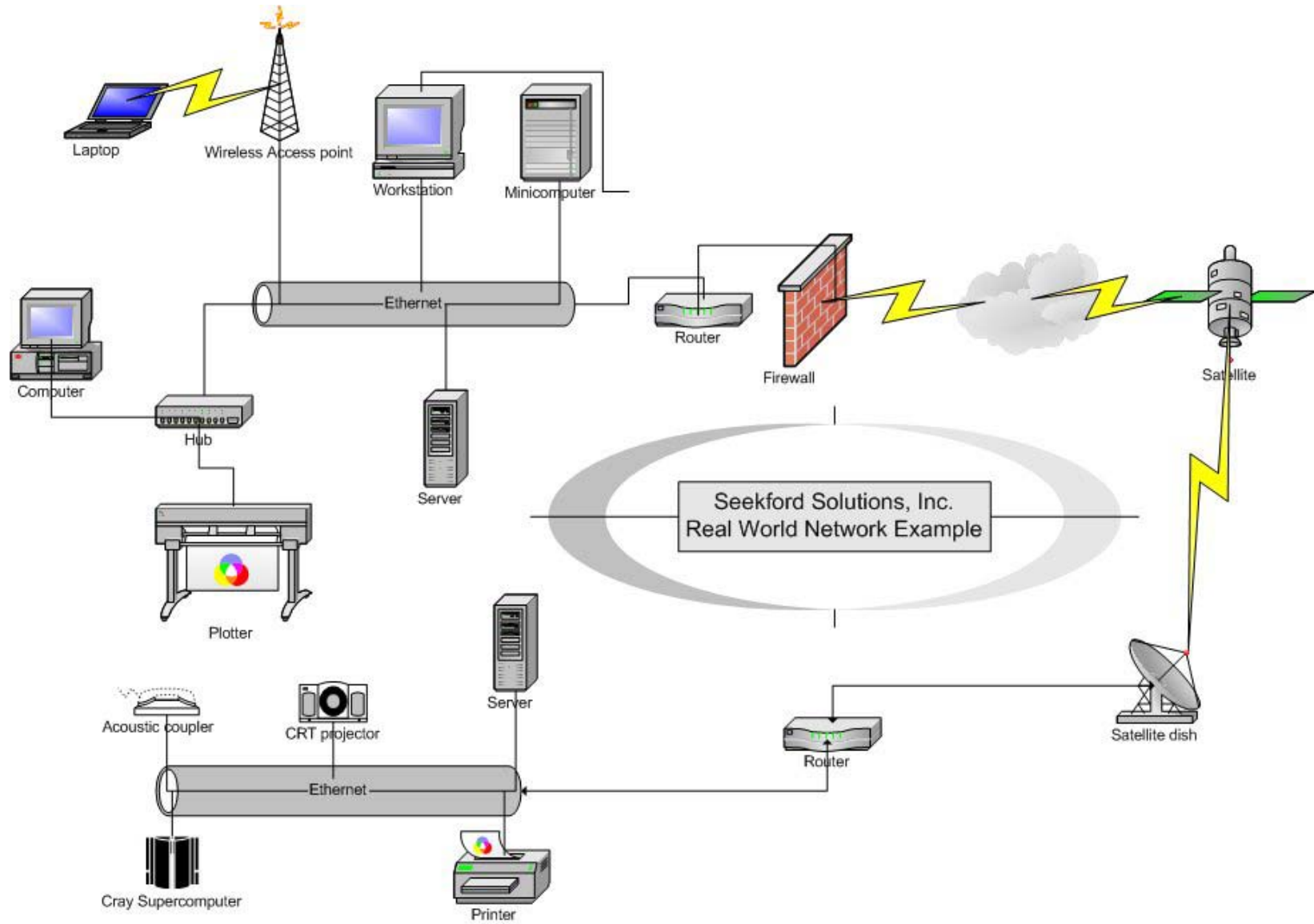


Diagram of Simple Network



What protocol is used for device status?

The network has been described as using TCP/IP as the basis for communication. The relevant part of that is the IP or Internet Protocol. This has a subset, which allows for some standard functionality for communicating with devices. One such functional level is called ICMP or Internet Control Messaging Protocol.

What is ICMP?

ICMP, or Internet Control Messaging Protocol, supports packets containing error, control, and informational messages. It is a critical part of the Internet Protocol module and is required to be implemented in all IP stacks. ICMP is used for transmitting messages about the status of packet sends, network problems and more. The relevant part of the ICMP that is used for checking if a device is online is the ECHO command. Traditionally this is called a Ping, since this the name of the utility from UNIX that utilized the command.

What is a Ping?

Ping is actually the name of a utility written by a programmer for the Unix operating system but now has become synonymous with the ICMP Echo command. A Ping works by sending a special packet of data to a host machine and then waits for the response. The time it takes for this response is called the RTT or Return Trip Time. The longer the time, the worse the network link is between the client and host machines. If there is no response, then it can be assumed that either the network link has been broken or the host is not currently online or active.

It is often believed that "Ping" is an abbreviation for Packet Internet Groper, but Ping's author has stated that the names comes from the sound that a sonar makes. We tend to like the Packet Internet Groper since it seems a better techie term.

How can I use ICMP to conduct a Ping?

To implement the ICMP protocol there are a couple of ways of handling it. The main way is to create a RAW socket and format a special data packet that is sent using an IP packet to the host. This can be complicated since it requires a lot of knowledge about how Sockets work and also about Network byte ordering which is different from the Intel byte ordering used in most PC's memory. The

easy way of handling a Ping is to use Ping Wizard. It is an ActiveX control that works in a normal container application or even in scripted languages like VBScript and Active Server Pages.

What is Ping Wizard?

Ping Wizard is an ActiveX control created by Seekford Solutions, Inc. that makes sending Pings very simple. Only one line of code is needed to actually issue the Ping. It then provides a lot of information from the host such as the RTT or Return Trip Time, the TTL, or Time to Live, and response data provided by the host. Monitoring network connections becomes incredibly easy with this tool and implementation into your application is trivial.

Can you show me an example of how Ping Wizard works?

Below is source code for a variety of different programming languages that can use Ping Wizard. If yours is not listed, then use the code below to provide a guide for your coding.

The source code shows actual use for the product and possible data manipulation, so some of it may be lengthier than needed for the simplest functionality.

[Visual Basic]

```
Dim bOK As String
```

```
Dim sCurrentPING As String
```

```
'PingWizard21 is an Instance of Ping Wizard Version 2 on the Form
```

```
PingWizard21.ResolveNumericIPtoHostName = True 'Set to False for a faster response since the DNS lookup can take some time
```

```
sCurrentPING = "192.168.0.1" 'Address to Ping
```

```
bOK = PingWizard21.PingWithoutRAW(sCurrentPING)
```

```
If bOK Then
```

```

MsgBox CStr(PingWizard21.Address) & " - Connected" & vbCRLF _
      & "Address Name: " & PingWizardv21.AddressName & vbCRLF _
      & "RTT: " & CStr(PingWizard21.RTT)
Else
MsgBox CStr(PingWizard21.Address) & " - FAILED" & vbCRLF _
      & "Error: " & PingWizard21.LastErrorDescription
End If

```

[VBScript]

```

' This is a comprehensive example showing the results like you would see from the PING.EXE program
set MyPINGWizard = CreateObject("PINGWIZARD.PINGWizardCtrl2") 'Create the instance of Ping Wizard v2
MyPINGWizard.UnlockPingWizard("")
Dim sData,bIsOk,iIndex,iPacketsSent ,iPacketsReceived ,iPacketsLost ,dAveragems ,iMinms ,iMaxms
dim sMsg
MyPINGWizard.ResolveNumericIPToHostName = TRUE

For iIndex = 1 To 4
  iPacketsSent = iPacketsSent + 1
  bIsOk = MyPINGWizard.PingWithoutRAW("www.SeekfordSolutions.com")
  If bIsOk Then
    iPacketsReceived = iPacketsReceived + 1
    sData = "Reply from " & MyPINGWizard.Address & "[" & MyPINGWizard.AddressName & "]: bytes=" & _
           CStr(MyPINGWizard.ReplyDataLength) & " time=" & CStr(MyPINGWizard.RTT) & "ms TTL=" & _
           CStr(MyPINGWizard.TTL)
    If (iPacketsSent = 1) Then
      iMinms = MyPINGWizard.RTT
    Else
      iMinms = Iif(iMinms > MyPINGWizard.RTT, MyPINGWizard.RTT, iMinms)
    End If
    iMaxms = Iif(iMaxms < MyPINGWizard.RTT, MyPINGWizard.RTT, iMaxms)

```

```

    dAveragems = (IIf(iPacketsReceived = 1, MyPINGWizard.RTT, dAveragems) + MyPINGWizard.RTT) / 2
Else
    iPacketsLost = iPacketsLost + 1
    sData = "Error! Error number: " & CStr(MyPINGWizard.LastErrorNumber) & " Error Description: " &
MyPINGWizard.LastErrorDescription
End If
    sMSg = smSg & sdata & vbCrLf
Next
    smsg = smsg & vbCrLf & "Ping statistics for " & MyPINGWizard.Address & ":" & vbCrLf
    smsg = smsg & "Packets: Sent = " & cstr(iPacketsSent) & ", Received = " & cstr(iPacketsReceived) & ", Lost = " &
iPacketsLost & " (" & cstr(((Cdbl(iPacketsLost) \ Cdbl(iPacketsSent)) * 100)) & "% loss)," & vbCrLf
    smsg = smsg & "Approximate round trip times in milli-seconds:" & vbCrLf
    smsg = smsg & vbTab & "Minimum = " & cstr(iMinms) & "ms, Maximum = " & cstr(iMaxms) & "ms, Average = " &
cstr(CInt(dAveragems)) & "ms" & vbCrLf

```

```
msgBox sMsg
```

```

function IIF(a,b,c)
    if a then
        iif = b
    else
        iif = c
    end if
end function

```

[Active Server Pages]

<0%

```

dim sAddressToPing
sAddressToPing = "localhost"
set MyPINGWizard = Server.CreateObject("PINGWIZARD.PINGWizardCtrl2")

```

```

Response.write("Ping Timeout:" + CStr(MyPINGWizard.Timeout) + vbCrLf)
Response.write("Ping Packet Size:" + CStr(MyPINGWizard.PacketSize) + vbCrLf)
Response.write("PING address:" + sAddressToPing + " ")
Response.write("<HR>")
'NOTE: YOU WILL NEED A VALID LICENSE TO USE THIS IN ASP.
MyPINGWizard.UnlockPingWizard("")
Response.flush
Dim sData,bIsOk,iIndex,iPacketsSent ,iPacketsReceived ,iPacketsLost ,dAveragems ,iMinms ,iMaxms
MyPINGWizard.ResolveNumericIPToHostName = TRUE 'Set this to False to improve speed
Response.write("Pinging " & sAddressToPing & " with " & CStr(MyPingWizard.PacketSize) & " bytes of data:" & vbCrLf &
vbCrLf)
For iIndex = 1 To 4
    iPacketsSent = iPacketsSent + 1
    bIsOk = MyPINGWizard.PingWithoutRaw(sAddressToPing)
    If bIsOk Then
        iPacketsReceived = iPacketsReceived + 1
        sData = "Reply from " & MyPINGWizard.Address & "[" & MyPINGWizard.AddressName & "]: bytes=" & _
            CStr(MyPINGWizard.ReplyDataLength) & " time=" & CStr(MyPINGWizard.RTT) & "ms TTL=" &
CStr(MyPINGWizard.TTL)
        If (iPacketsSent = 1) Then
            iMinms = MyPINGWizard.RTT
        Else
            iMinms = IIf(iMinms > MyPINGWizard.RTT, MyPINGWizard.RTT, iMinms)
        End If
        iMaxms = IIf(iMaxms < MyPINGWizard.RTT, MyPINGWizard.RTT, iMaxms)
        dAveragems = (IIf(iPacketsReceived = 1, MyPINGWizard.RTT, dAveragems) + MyPINGWizard.RTT) / 2
    Else
        iPacketsLost = iPacketsLost + 1
        sData = "Error! Error number: " & CStr(MyPINGWizard.LastErrorNumber) & " Error Description: " &
MyPINGWizard.LastErrorDescription
    End If

```

```

    Response.write(sData & vbCrLf)
Next
Response.write( vbCrLf & "Ping statistics for " & MyPINGWizard.Address & ":" & vbCrLf)
Response.write("Packets: Sent = " & cstr(iPacketsSent) & ", Received = " & cstr(iPacketsReceived) & _
    ", Lost = " & iPacketsLost & " (" & cstr(((Cdbl(iPacketsLost) \ Cdbl(iPacketsSent)) * 100)) & "% loss)," &
    vbCrLf)
Response.write("Approximate round trip times in milli-seconds:" & vbCrLf)
Response.write(vbTab & "Minimum = " & cstr(iMinms) & "ms, Maximum = " & cstr(iMaxms) & "ms, Average = " &
cstr(CInt(dAveragems)) & "ms" & vbCrLf)
%>

```

[Visual C++]

Assumptions:

m_PingWizard is a member variable for an Instance of Ping Wizard v2 on a dialog
m_Response is a member variable for a textbox on a dialog.

Code:

```

char buff[20];
int iPacketsSent = 0;
int iPacketsReceived = 0;
int iPacketsLost =0;
double dAveragems =0;
int iMinms =0;
int iMaxms =0;
m_PingWizard.UnlockPingWizard("");
m_PingWizard.SetPacketSize(atol(m_PacketSize));
m_PingWizard.SetTimeout(atol(m_TimeOut));
m_PingWizard.SetResolveNumericIPToHostName(m_ResolveNumeric);
m_Response = "Pinging " + m_AddressToPing + " with " + m_PacketSize + " bytes of data:\r\n\r\n";
for(int iIndex =0;iIndex <4;iIndex++)

```

```

    {
        iPacketsSent++;
        if (m_PingWizard.PingWithoutRAW(m_AddressToPing))
        {
            m_Response += "Reply from " + m_PingWizard.GetAddress() + "[" + m_PingWizard.GetAddressName() + "]:
bytes=";

            _itoa(m_PingWizard.GetReplyDataLength(),buff,10);
            m_Response += CString(buff) + " time=";
            _itoa(m_PingWizard.GetRtt(),buff,10);
            m_Response += CString(buff) + "ms TTL=";
            _itoa(m_PingWizard.GetTtl(),buff,10);
            m_Response += CString(buff) + "\r\n";
            iPacketsReceived++;
            if (iPacketsReceived == 1)
            {
                iMinms = m_PingWizard.GetRtt();
            }else
            {
                iMinms = (iMinms > m_PingWizard.GetRtt()) ? m_PingWizard.GetRtt() : iMinms;
            }
            iMaxms = (iMaxms > m_PingWizard.GetRtt()) ? iMaxms : m_PingWizard.GetRtt();
            dAveragems += m_PingWizard.GetRtt();
        }else
        {
            _itoa(m_PingWizard.GetLastErrorNumber(),buff,10);
            m_Response += m_PingWizard.GetLastErrorDescription() + "\r\n";
            iPacketsLost++;
        }
    }
    dAveragems /= iPacketsReceived;
    m_Response += "Ping statistics for " + m_PingWizard.GetAddress() + " :\r\n";

```

```

_itoa(iPacketsSent, buff, 10);
m_Response += "Packets: Sent = " + CString(buff) + ", Received = ";
_itoa(iPacketsReceived, buff, 10);
m_Response += CString(buff) + ", Lost = ";
_itoa(iPacketsLost, buff, 10);
m_Response += CString(buff) + " (";
_itoa(((double)iPacketsLost / (double)iPacketsSent) * 100, buff, 10);
m_Response += CString(buff) + "% loss),\r\n";
m_Response += "Approximate round trip times in milli-seconds:\r\n";
_itoa(iMinms, buff, 10);
m_Response += "\tMinimum = " + CString(buff) + "ms, Maximum = ";
_itoa(iMaxms, buff, 10);
m_Response += CString(buff) + "ms, Average = ";
_itoa((int)dAveragemms, buff, 10);
m_Response += CString(buff) + "ms\r\n";
AfxMessageBox(m_Response);

```

[Visual FoxPro]

Assumptions:

That a dialog has the PingWizardv2 object on it name MyPINGWizard or that the createobject command is used.

Code:

```

&& set MyPINGWizard = CreateObject("PINGWIZARD.PINGWizardCtrl2")
&& Uncomment the above line of code and remove the THISFORM ids if you want to create the object at runtime
THISFORM.MyPINGWizard.UnlockPingWizard("")
THISFORM.MyPINGWizard.ResolveNumericIPToHostName = .t.
local sData,bIsOk,iIndex,iPacketsSent ,iPacketsReceived ,iPacketsLost ,dAveragemms
sData = ""
bIsOk = .f.
iIndex = 0
iPacketsSent = 0

```

```

iPacketsReceived = 0
iPacketsLost = 0
dAveragem = 0
iMinms = 0
iMaxms = 0
sMsg = ""
dPacketLoss = 0
For iIndex = 1 To 4
    iPacketsSent = iPacketsSent + 1
    bIsOk = THISFORM.MyPINGWizard.PingWithoutRaw("www.SeekfordSolutions.com")
    If bIsOk Then
        iPacketsReceived = iPacketsReceived + 1
        sData = "Reply from " + THISFORM.MyPINGWizard.Address + "[" + THISFORM.MyPINGWizard.AddressName + "]:
bytes=" + STR(THISFORM.MyPINGWizard.ReplyDataLength) + " time=" + STR(THISFORM.MyPINGWizard.RTT) + "ms TTL="
+ STR(THISFORM.MyPINGWizard.TTL)
        If (iPacketsSent = 1) Then
            iMinms = THISFORM.MyPINGWizard.RTT
        Else
            iMinms = IIf(iMinms > THISFORM.MyPINGWizard.RTT, THISFORM.MyPINGWizard.RTT, iMinms)
        EndIf
        iMaxms = IIf(iMaxms < THISFORM.MyPINGWizard.RTT, THISFORM.MyPINGWizard.RTT, iMaxms)
        dAveragem = (IIf(iPacketsReceived = 1, THISFORM.MyPINGWizard.RTT, dAveragem) +
THISFORM.MyPINGWizard.RTT) / 2
    Else
        iPacketsLost = iPacketsLost + 1
        sData = "Error! Error number: " + STR(THISFORM.MyPINGWizard.LastErrorNumber) + " Error Description: " +
THISFORM.MyPINGWizard.LastErrorDescription
    EndIf
    sMsg = sMsg + sData + + chr(13)
Next
if (ipacketslost >0) then

```

```

        dPacketloss = ((iPacketsLost) / (iPacketsSent)) * 100
    endif
    msg = msg + chr(13) + "Ping statistics for " + THISFORM.MyPINGWizard.Address + ":" + chr(13)
    msg = msg + "Packets: Sent = " + STR(iPacketsSent) + ", Received = " + STR(iPacketsReceived) + ", Lost = " +
STR(iPacketsLost) + "(" + STR(dPacketLoss) + "% loss)," + chr(13)
    msg = msg + "Approximate round trip times in milli-seconds:" + STR(dAveragem) + chr(13)
    msg = msg + "    " + "Minimum = " + STR(iMinms) + "ms, Maximum = " + STR(iMaxms) + "ms, Average = " +
(STR(dAveragem)) + "ms" + + chr(13)

    MessageBox( msg)

```

Where can I find more information about Ping Wizard?

More information can be found at the product's web link: <http://www.seekfordsolutions.com/Products/PingWizard/>

Who is Seekford Solutions, Inc.?

Seekford Solutions, Inc. is a software development corporation specializing in the design and development of state of the art ActiveX controls and custom projects. Their core product line is focused on Internet technologies primarily in the facilitation of the use of the common Internet protocols. The design philosophy is based on ease of use and quick implementation time. They also handle custom projects for clients who are need of specialty software or who need a framework base to use. The company was founded in early 2001. Their website is <http://www.seekfordsolutions.com/>