

Remote Management of Win2K Servers: Three Secure Solutions

Mark Burnett 2002-09-25

It's a common scenario: your company has an IIS Web server sitting 300 miles away at a high-bandwidth, air-conditioned and power-regulated co-location center. The network is stable and the price is right, but you must completely manage the server remotely; you can't just go sit down at the console whenever you want. Remote management presents several problems, the most obvious being that the traffic between you and the server is travelling across the public Internet, available for others to sniff. Another problem is that remote administration normally involves installing software and opening ports, both of which increase the attack surface of your server. The goal when selecting a remote administration solution is to make sure that you (and only you) can do your job without exposing the server to additional risk.

In particular, the concerns when administering a remote server are:

- Access Control
- Integrity
- Confidentiality
- Auditing

Access Control

Access control is making sure that only you can remotely administer the server. This means that the remote administration software should only accept connections from a small range of IP addresses and should prompt for a username and password. Access control can be further strengthened through the use of smart cards and client certificates. There are also obscurity techniques that may provide additional layers of protection such as using non-standard TCP ports or suppressing service banners.

Integrity

Integrity ensures that the data received by the server is the same data that you sent. You also want to be sure that a packet is authentic and cannot be replayed at a later time.

Confidentiality

Perhaps the greatest concern with remote administration is that sensitive data is travelling across a public network. Confidentiality ensures that this traffic cannot be intercepted and viewed by others. Confidentiality means using strong, accepted encryption algorithms with a sufficiently large encryption key.

Auditing

Auditing is the ability to log all access to a server for later analysis. It is important to remember that a server could very well become a crime scene and it is essential that your remote access solution keep sufficient information about every connection to the server. Furthermore, the logs should be moved off the server itself to ensure their integrity.

Remote Management Methods

Although there are a variety of ways to remotely manage a Win2K server, not all products provide the security requirements listed above. But that doesn't mean we cannot use them. By combining different products we can come up with some very secure solutions that provide features we need to administer remotely.

Below are some examples of what can be done using built-in or third-party open source solutions. While there is no one best way to remotely administer a server, these are good examples of what can be done when combining solutions.

Option 1: Terminal Services Through Zebedee

Terminal Services is a built-in service in Windows 2000 that provides admins with a remote desktop for managing a server. Terminal Services is the most obvious way to remotely manage a server because it is built-in, easy to get running, uses built-in Windows accounts for authentication, and allows for strong encryption. But there are some limitations: there is no mechanism to limit access by IP address, it is not obvious how to change the default listening port, and it has no logging facility. Based on the list of requirements at the beginning of this article, Terminal Services alone does not score well on security.

But Terminal Services can be made more secure by tunneling the traffic through another tool called [Zebedee](#). Zebedee is an open source program that allows you to redirect TCP or UDP traffic over encrypted, compressed tunnels. Zebedee has a small footprint and offers

encryption, authentication, IP address filtering, tunneling, and logging. Zebedee can give Terminal Services the boost it needs to make it a very secure remote administration solution.

Zebedee works by listening on a local port and then encrypting and compressing that traffic by sending it to another copy of Zebedee running on the server. The result is a tunnel that can handle multiple TCP or UDP connections over a single TCP port.

To begin, you must configure Zebedee to listen for new connections on a non-standard port (note that your firewall should not allow outside connections to port 3389). This can be done with the following command:

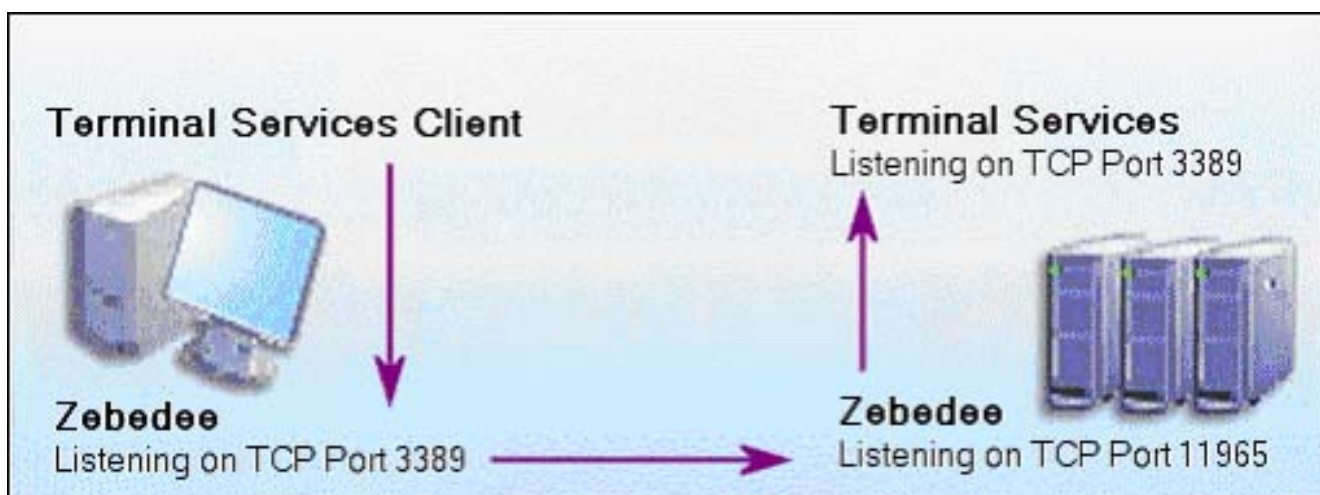
```
c:\>zebedee -s -o server.log
```

Next, you run Zebedee on your client and have it listen on port 3389 and redirect that traffic to the server port you selected.

```
C:\>zebedee 3389:serverhost:3389
```

This causes Zebedee to listen to port 3389 on the local system and forward that traffic to the remote system (in this case named serverhost), which will forward the traffic on to the local port 3389, where Terminal Services is listening (see Figure 1 Below). Note that by default the communication between the server and client will be through TCP port 11965, but this should be changed when you put it into production.

Figure 1: Terminal Services on Zebedee



At this point, you can open your Terminal Services client and enter localhost as the server. The

client will connect to the local Zebedee client which will forward it to the Zebedee server, which will in turn then forward to the remote Terminal Services port. Terminal Service just thinks it is connecting locally, but in fact all traffic is being tunneled through a secure channel.

Zebedee has many options in its configuration file for authentication, encryption, IP address filtering, and logging. Be sure to explore these options and implement as many as possible to ensure your configuration is secure.

Since Terminal Services does not have any option for transferring files, you will need to consider other options. One option is to use an FTP server. Although FTP is normally not secure, you can also tunnel this through the same Zebedee connection. Doing so is a bit tricky but the Zebedee documents have detailed instructions on how to do so.

There are also two other third party solutions for transferring files directly over Terminal Services. One is a free tool called [TSDropCopy](#) from [Analogx](#) and the other is a shareware tool called [WTS-FTP](#) from [Ibexsoftware.com](#).

Overall, Terminal Services is a convenient way to manage a server but in itself is not secure enough to be used remotely. But if it is tunnelled through Zebedee, it can be a very secure solution.

Option 2: VNC On SSH

[VNC](#) is a remote desktop tool very similar to Terminal Services, providing remote desktop access to the server. There are, however, some key differences, such as:

- VNC works with the existing desktop on the server rather than creating multiple virtual desktops;
- VNC clients are available on many platforms, including Windows CE and Java;
- VNC is open source;
- VNC can restrict access by IP address; and,
- Traffic between the client and server is not encrypted;

While VNC does have some benefits, it is not secure enough to use by itself. The most significant problem is the lack of encryption. But VNC traffic, like Terminal Services, can be tunnelled to make up for its shortcomings. In this case, we will match it up with SSH (a

Windows installation of OpenSSH is available at <http://www.networksimplicity.com/openssh>). OpenSSH is in concept similar to Zebedee but it is a much broader application that also allows for a remote command prompt, secure file copy (SCP), and secure FTP (SFTP). Like Zebedee, it can tunnel traffic over a single port; however, it is limited to TCP traffic. SSH supports strong public key encryption and is a widely-used protocol with strong user support.

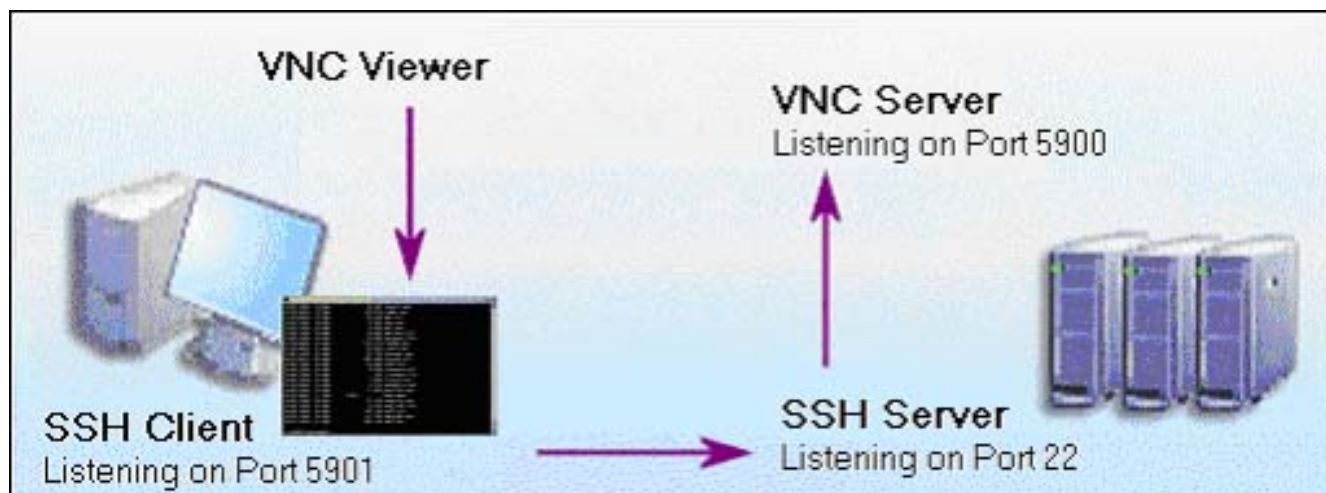
The concept of forwarding traffic over SSH is very similar to using Zebedee. You configure the server to listen on a single port (TCP port 22 by default), then connect to that port using an SSH client. An SSH client is essentially an encrypted telnet client that gives you command prompt access to the server. But SSH also allows you to use port forwarding to allow other protocols to work over the same encrypted connection. To instruct your SSH client to forward the VNC ports through the connection, use the following command:

```
C:>ssh -L 5901:serverhost:5900 serverhost
```

This will create a local listener that appears to be a VNC listener on the local system that forwards to the traffic to serverhost. To connect, point your VNC viewer to localhost:1 as follows:

```
C:\>vncviewer:1
```

Figure 2: VNC on SSH



VNC will think it is connected to an unencrypted session on the local system while in fact the traffic is encrypted and tunnelled to the remote VNC listener.

VNC on SSH is an excellent solution if you must manage the server from multiple client

platforms. Both VNC and SSH are supported on most major operating systems. Further, both applications are open source and have been around long enough to be quite stable and secure.

Option 3: Windows over VPN

If you are on an all-Windows network, you may wish to simply use the built-in administrative tools to manage your server. For example, you may wish to map a drive to the server, and take advantage of the many Windows networking services available on Windows 2000. You certainly can accomplish this by opening up port 445 on the server's firewall and only allowing your IP address to connect to that port. However, all traffic between you and the server will not be encrypted and can be sniffed. Again, we must tunnel the traffic using another technology. In this case a good match is L2TP tunnelling using the built-in Windows VPN server and client.

To do this, you must enable the Routing and Remote Access, the Server, and the Workstation services. Next, open the Routing and Remote Access administrative tool and right-click on your server. Select the Configure and Enable Routing and Remote Access option and follow the instructions to create a new Virtual Private Network server.

Note that the VPN server will listen on TCP port 1723. If you do not wish for others to see this port open, it is important to restrict access to this port by only allowing a limited range of IP addresses to connect to it.

On the client end, you must now create a new connection by right-clicking on My Network Places and selecting Properties. From there, click on Make New Connection and select Connect to a Private Network through the Internet. Follow the prompts to configure the connection for your server. When finished, you will have a new icon for your VPN connection. Double-click on the icon and you will be prompted for a username and password to connect to the server. Once connected, you will have a new network connection, complete with a new IP address, that connects directly to the server just as if you had installed a new network adaptor and ran a cable directly to the server. The only difference is that this connection travels through an encrypted tunnel across the Internet.

As with any network connection, you should be sure to only allow the network protocols you need and you should consider using packet filtering. Keep in mind that this connection could allow an attacker to use your computer to get to your web server or use your web server to get to your internal network. Either way, you must consider the risks involved.

Once connected through a VPN, you can map drives or manage the server as if it were connected to the local network. While this option does introduce considerable risks, it is by far the most convenient solution. If properly secured on both ends, this solution can be sufficiently secure for remote server management.

Other Options

This article has discussed three possibilities but there are many variations to these solutions. The point isn't so much to showcase any particular solution, but to show what it takes to make remote administration secure. If you have another remote administration tool, look at the requirements at the beginning of this article and see if it passes. If it doesn't, you may consider the tunnelling solutions mentioned here or one of the many others available. You may need to mix and match products to find the one solution that works best for you. Sometimes all it takes is a little extra help of an encrypted tunnel or a good firewall configuration to make a remote management solution very secure.

[Privacy Statement](#)

Copyright 2006, SecurityFocus