WinFrame Fail-Over Strategies

BY GUY C. YOST

This article focuses on strategies such as fault tolerance and redundancy that are geared toward ensuring that your highly visible WinFrame environment will not be caught "dead" while fulfilling its mission.

DUE to the rising interest in thin-client/server computing and the success of Citrix's WinFrame product, over the past few months, John Johnston and I have written several articles that explore various aspects of this technology. For a WinFrame overview, refer to the article, "Application Serving Takes Shape: Understanding WinFrame Servers" (***Technical Support***, February 1998). Subsequent issues examined deploying load balancing (April 1998) as well as application installation and remote access (June 1998).

This article focuses on fail-over strategies that are geared toward ensuring that your highly visible WinFrame environment will not be caught "dead" while fulfilling its mission.

The oldest rule in fault tolerance is redundancy. The bet is simple: If one server dies, then another will take over. This also applies to components within a server configuration.

WINFRAME SERVER TYPES

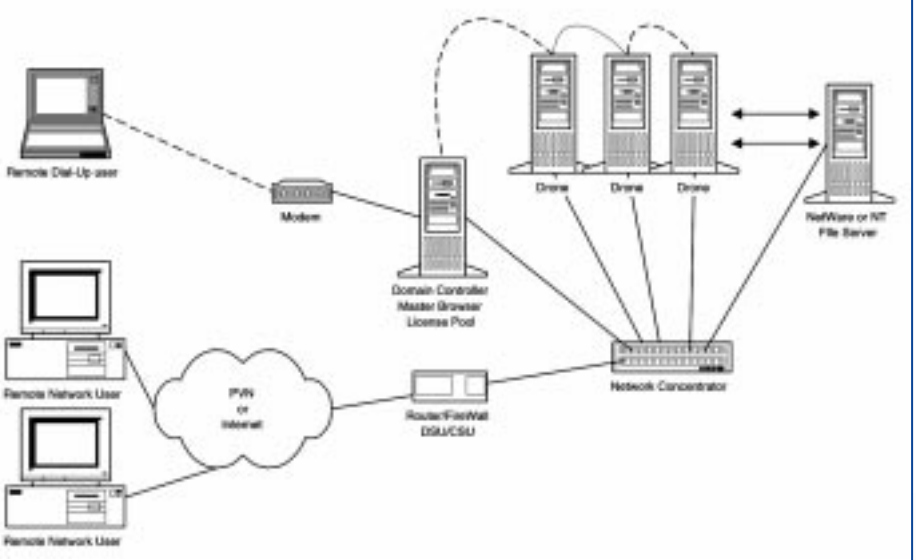
Before we explore redundancy in a WinFrame environment, it is helpful to know the different roles that servers can assume. Depending on the number of users supported, a WinFrame farm may consist of one or more of the following:

"Drone" servers: These are the worker bees in the farm that ultimately host the client sessions. Remember that each drone session acts as a client on the network, even though the users connect to a WinFrame terminal server. Each user session is supported in a separate memory space, so drones will need to be well equipped with respect to memory and CPU power. Recall from the February issue that the drones used for a recent project each have 2GB RAM and four 200 MHz Pentium II processors. This allows each server to comfortably support its operating system and 30 users, assuming 32MB RAM per user.

Domain Controllers (DCs): Standard MS domain controllers are needed to support load balancing and authentication to the farm domain. As with standard NT networking, a single domain may have both primary and backup domain controllers. These servers can be configured modestly compared to the drones, and can either be a stock Microsoft NT server or an actual Citrix WinFrame server. The advantages of each are explained in this article.

Master Browser (MB): To improve the efficiency of load balancing and to simplify client configuration options, all client traffic can be directed to the farm's Master Browser(s). The MB acts as the traffic cop, directing connections to the drones based on the number of users connected to each drone. The MB is also the logical location for pooling WinFrame client licenses.

Figure 1: Network Infrastructure Supporting Multiple Server Types



Application File Server (optional): For larger installations where numerous drones are needed to support the user base, a standard file server can be used to host a central copy of the applications clients will access. The drones simply access the applications as would a normal client; however, drones host several concurrent users. Alternatively, all applications would need to be installed onto each drone. This might not create a great deal of administrative overhead if using only a few drones, but it would be an administrative nightmare to support across numerous servers, with or without help from third-party server-cloning software. A proposed environment that uses all server types and functions is shown in Figure 1. Note the following about Figure 1:

Technical Note

An ICA Master Browser is similar to a Windows NT Master Browser in that they both list available servers and resources on the network. The ICA Master Browser, however, must be a Citrix server because it knows how to advertise and publish ICA applications and can manage license pools for other Citrix servers. To ensure that a Citrix server is configured to be Master Browser, use REGEDT32 to set the following registry entries found in the HKEY_LOCAL_MACHINE\SYSTEM\CURRENTCONTROLSET\SERVICES\ICABROWSER\PARAMETERS hive key:

```
ISMASTERBROWSER=1
NOTMASTERBROWSER=0
```

The settings combination logic is as follows:

```
ISMASTERBROWSER=0
NOTMASTERBROWSER=0
```

means that the Citrix server is not currently acting as an ICA Master Browser, but it could if asked (i.e., it is eligible for an election.)

```
ISMASTERBROWSER=0
NOTMASTERBROWSER=1
```

means that the Citrix server is not currently acting as an ICA Master Browser, nor should it be eligible in a browser election.

```
ISMASTERBROWSER=1
```

means that the Citrix server is currently acting as an ICA Master Browser for the Citrix domain, so the value of NOTMASTERBROWSER is not important.

You can configure the ICA client to direct its attach request to a specific list of ICA Master Browsers in two ways:

1. If you are using the Remote Application Manager, choose "Options," then "Settings" from the main menu bar, then select the "Server Location" tab. On this screen, you can add any number of IP or IPX server addresses which are searched in the order as listed. Enter the network addresses of the ICA Master Browsers here.

2. If you are using the "ICA file" option to launch the ICA client, then you can add a list of ICA Master Browsers. Right-button click on the "ICA" icon and choose "Properties." Next, click on the "Address" tab and then click on the "Server Location" button. Add the addresses of the ICA browsers here, and in the text of the ICA file it will create similar lines as:

```
[WFCLIENT]

Version=2

TcpBrowserAddress=123.45.67.89

TcpBrowserAddress2=123.45.67.90
```

Using this interface requires that the ICA file editor is installed on the workstation. If the ICA client editor is not installed on the workstation, then the entries shown can be manually inserted using a text editor such as Notepad. The client will send the initial request to the browser(s), which then directs the client session to one of the available server "drones."

- The file server can be either NT or NetWare, depending on the NOS used in your normal network environment. By accessing the applications from an existing LAN file server, this proposed design leverages the existing resources in your network.

- The Domain Controller also acts as the Master Browser and license-pooling repository. To support all of these functions, this server would need to be a Citrix WinFrame server rather than a standard NT server because NT doesn't support Citrix License pooling.

- The user community would configure their ICA clients to point to the IP address of the DC/MB server, which would then direct the connection request to the least utilized drone. If the DC/MB is not part of the advertised server farm application (or desktop), then it won't host user sessions.

FAULT TOLERANCE 101

The oldest rule in fault tolerance is redundancy. The bet is simple: If one server dies, then another will take over. This also applies to components within a server configuration. Redundant NICs, power supplies, CPUs, fans, and drives (via RAID) will add reliability to your mission-critical servers, and when configured correctly, servers won't skip a beat when a component fails. Most hardware vendors offer server-class systems that support these features, and although that topic is not WinFrame-specific, it is noteworthy during the hardware planning stage.

Taking a broader view of redundancy, several techniques can be applied to a WinFrame environment thereby providing fault-tolerance using:

- WinFrame Server load balancing across drones
- multiple Domain Controllers
- multiple Master Browsers
- client license-pooling
- multiple file servers
- geographically dispersed server resources

Keep in mind that any of these techniques can be used in a small, medium, or large-scale WinFrame environment. To cover all bases, this article assumes a large-scale deployment and you can omit items where appropriate.

FAULT TOLERANCE THROUGH LOAD BALANCING

Although configuring WinFrame load balancing was covered in detail in the April issue, I'll review the key aspects of this technology for completeness.

By having multiple servers host the same applications, clients can attach to and use any available server in the "farm." Each server is configured identically with the same drive contents so users won't need to know or care which server they are using, as long as the hosting environment is consistent across servers.

When a user connects to a server farm using the ICA client, the session is sent to the most available server (based on an overall score calculated using the number of connections, CPU and disk utilization, and amount of free RAM.) The session is "hard" established between the client and the drone so that if the drone goes down, then the user will lose his session. The user must then reconnect to the server farm and establish another session on an "up" server. Although this doesn't sound ideal, it does offer a quick turnaround so downtime for the user is minimal. As mentioned in February's article, seamless fail-over is possible using server cluster technology from other vendors but at a considerably higher expense.

There's a danger when losing the connection if the user is storing or accessing manipulative data on the drone. It is therefore not recommended that data be stored on the drone; instead, use the user's home directory or local drives. However, there's one exception: The client session will require a defined TEMP environment variable but you can assign this to a volatile directory on the drone because the data won't be used from one session to another. WinFrame normally takes care of this by automatically creating unique subdirectories off of the server's defined TEMP directory based on the client's connection ID. To ensure this feature is active, issue the following command from a DOS window while logged in as Administrator:

```
FLATTEMP /DISABLE
```

Here's a tip: During the login process, set the user's session TMP and TEMP environment variables to T:\, or another free drive letter. Then use the SUBST command to associate the T:\ drive with the currently allocated temp directory as follows:

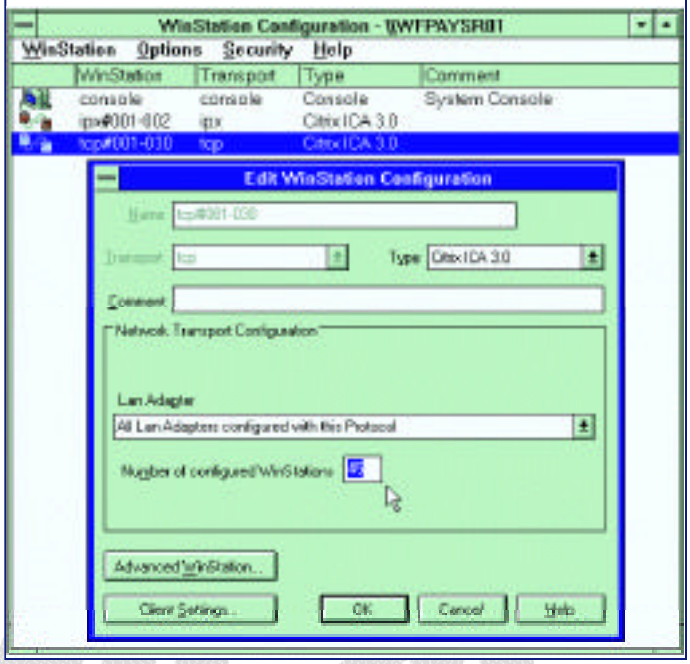
```
SUBST T:\ %TEMP%
```

This way, applications that use explicit temporary directory assignments in the application setup or INI file can use "T:\" as their temp directory, while the system will still keep track of the actual \TEMP\XX directory for its own housekeeping. Application and system temp files will go to the same place and be automatically discarded when the session ends.

MULTIPLE DOMAIN CONTROLLERS AND MASTER BROWSERS

As with standard NT domains, a WinFrame server farm will need at least one DC to authenticate users and to support load balancing. As

Figure 2: Adjusting License Pooling



mentioned previously, it is possible to use a standard NT 4 or 3.51 server as the DC. A popular choice for a DC is a desktop computer equipped with NT 4, 128MB to 1GB RAM (depending on the number of users; count on 1MB per user with a minimum of 128MB), and a single 200 MHz or faster CPU. Alternatively, you can combine the following functionality into one WinFrame server:

- Domain Controller (Primary or Backup)
- Master Browser (Primary or Secondary)
- license pooling repository (split licenses between two servers)

The advantage of using a WinFrame server as the DC is that the server can also be used as the farm's Master Browser and license-pooling repository. Using this approach, you would most likely want to configure the WinFrame DC not to host actual user sessions by simply not including the DC server name in the published application list (explained in the April 1998 issue).

By having two DCs, (a PDC and BDC) and by configuring each to be a Master Browser and host pooled licenses, you will have redundancy of all functions in the event that one DC goes down. While both DCs are active (which should be 99 percent of the time) they will share in acting as DCs, MBs, and available licenses. This allows a user's initial connect request to be answered by the BDC if the PDC is busy. The only drawback is that you will temporarily lose a number of pooled licenses while a DC is down, but it won't disconnect sessions already established to drone servers before the DC became unavailable. This leads us to a discussion of license pooling strategies.

CLIENT LICENSE POOLING STRATEGIES

When a drone server is installed, you must provide a base-license serial number and then activate that serial number. If, for example, you plan on allowing 30 users per server as a comfortable maximum, then you could install a 15-user based license and then another 15-user "bump-pack" for a total of 30 licenses active on a server. In that case, all licenses would be local to the server and not pooled. If the server goes down, you lose 30 licenses for the duration of down-time.

If you kept 15 licenses local and pool the other 15, then if a drone went down you'd only lose 15 licenses while the other 15 are pooled (on the PDC or BDC) for use on another server as shown in Figure 2.

The pooled 15 user licenses could then be used on "up" servers, even if they are each currently serving 30 users. Here's how: To allow 45 (or more) users on a drone, login to the drone as Administrator and run the Winstation Configuration utility in the Administration Tools program group. To allow any number of remote users to connect to the server, you will need to define a Winstation profile that allows you to specify the number of supported concurrent users. We set that number to 45.

When we tested the user capacity of the drone, we targeted 45 users as a hard maximum on each box while keeping the CPU average under 60 percent, and with plenty of available RAM. In the event that one or more drones go down, other drones can accept up to 45 users, drawing from the license pool.

The license pool also allows for a likely occurrence: Assume that drone A has 15 users on it and those 15 users are running linear regression algorithms in some statistical program. It is probable that the CPU utilization is very high on that server, but drone B can host 15 additional users even though it already has 30 users running lighter applications.

At first we thought that by pooling half of the total licenses on the DCs we would be too vulnerable to user support loss if either DC went down. To mathematically verify our concern, we built a "what-if" Excel spreadsheet that allowed us to plug in various licensing scenarios and then the obvious hit us! If you have a server farm comprised of 22 drones, two DC/MBs, and two file servers, then what are the statistical odds that when any one machine fails, it would be a drone rather than a DC or a file server? The odds are 11 to 1 it would be a drone.

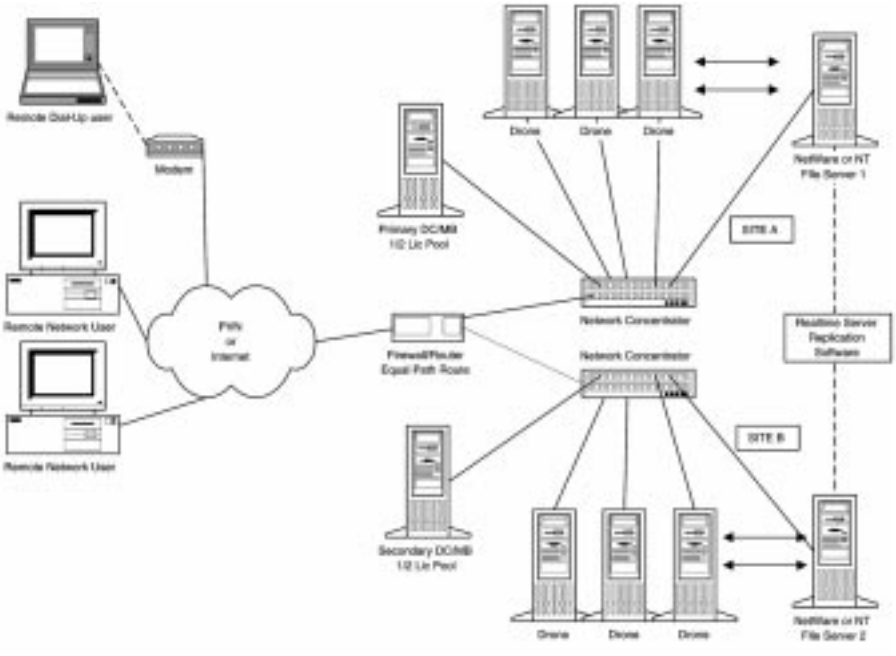
With all things considered, we devised a license deployment system that would support 30 users per server (under normal operating conditions), yet optimize the number of supported users in the event that one or two servers failed. In a nut shell, we purchased 690 total licenses, installed 15 on each of the 22 drones (330 users), and split the remaining pool of 360 licenses between the two DC/MBs. (If you're doing the math, the extra 30 licenses come from the base 15-user licenses installed on each DC/MB.)

MULTIPLE FILE SERVERS

If you are using a separate file server to host the users' home drives and applications (as shown in Figure 1) then you will want to protect your network in the event of a file server failure. Like the DC, a centralized file server is a critical component and without redundancy you will be very open to a single point of failure.

If using NT as your NOS, the NT file servers can be synchronized using products such as Octopus Technologies' Super Automatic Switch Over software (SASO). The file servers would be configured for standard NT file and print services, and would

Figure 3: Geographic Redundancy



therefore require an appropriate number of NT server licenses in addition to the WinFrame licenses. If using Novell or NDS as your NOS, the file servers can be synchronized using Novell's SFTIII or NetWare Replication Services (NRS), and naturally you would need an appropriate number of NetWare licenses.

In either case, the critical information that should be synchronized is the user's data as well as customized application files. The network topology that supports multiple file servers is shown in Figure 3. Note the following about Figure 3:

- Both sites share one common logical segment and domain. All IP addresses used are part of the same subnet.
- The firewall/router connection to the outside world can be located at either site A or site B — it doesn't matter as long as the route is configured as equal cost.
- Drones and licensing are evenly distributed between sites.
- Rather than support modem banks at either or both sites, dial-up users connect to the farm using a layer 2 forwarding service from our WAN connection provider. The provider manages the modem bank and merges dial-in traffic to a frame-relay connection that is passed to the server farm along with WAN and PVN traffic.
- Dial-up authentication is resolved using RADIUS services between the provider and the server farm.

GEOGRAPHICALLY DISPERSED SERVER RESOURCES

As part of the inherent fail-over design, consider the physical location of your redundant resources. For a current project, we have evenly split the number of drones, DC/MBs, and file servers across two sites about a half-mile apart. They are connected by redundant 200MB/s fiber channels. Each building is independently powered so that if lightning strikes building A, building B will keep on ticking. The network topology to support all redundancy discussed in this article is shown in Figure 3.

SUMMARY

There is a lot to planning a robust WinFrame environment. The next challenge faced by administrators is to streamline the NOS aspects of the design to make on-going maintenance of the farm not too consuming. In future articles,techniques for simplifying user and farm maintenance will be introduced. In particular, tips on integrating WinFrame into an enterprise Novell NDS environment will be presented. ts

NaSPA member Guy C. Yost is the owner of Redstone Consulting, an IT management consulting firm in New York. He has authored several books on networking for Que Publishing, including Learning NetWare 4.1, and NetWare 4.1 SmartStart, and contributes to Technical Support magazine as an author, columnist and technical editor. Guy also develops and conducts seminars on networking with Windows NT, UNIX, NetWare and Internet/intranet technologies across the United States and Canada. He can be reached at (518) 674-5606 or gyost@logical.net.