Why the need for MetaFrame?

TSE currently has three main shortcomings (lack of scalability, support for heterogeneous environments, and management and configuration tools) that hinder its ability to be a complete enterprise thin-client solution. MetaFrame’s ability to provide a solution to these shortcomings makes it a necessary component of any large-scale enterprise TSE implementation.

Independent Computing Architecture (ICA)

Citrix’s Independent Computing Architecture (ICA) is the foundation upon which both MetaFrame and WinFrame are built. It is the distributed presentation services protocol used to deliver a number of MetaFrame’s key features.

MetaFrame management tools

The tools that come with MetaFrame provide robust support for scaling both Terminal Servers and their users within the enterprise.

Client desktop integration

MetaFrame delivers an impressive assortment of desktop integration features that make it difficult at times to determine whether you are actually running remotely or not.

MetaFrame licensing

MetaFrame licensing differs from the standard Terminal Server licensing in that you are purchasing concurrent or server-based user licenses instead of client-based licenses.

Why the Need for MetaFrame?

Although Terminal Server provides complete multiuser functionality on its own, there are currently three main shortcomings in TSE that hinder its ability to be a complete enterprise thin-client solution. MetaFrame’s ability to provide a solution to these shortcomings (in addition to other features and
utilities) makes it a necessary component of any large-scale enterprise TSE implementation. The added features that MetaFrame provides are as follows:

- Scalability
- Support for heterogeneous computing environments
- Management and configuration tools

**Scalability**

As we mentioned when discussing scalability in Chapter 1, "The Features of Terminal Server," the Terminal Server product is targeted at small businesses or corporate departments where the number of users and the complexity of the environment are relatively low.

The drawback to TSE is that it is not really scalable. As the number of servers in a TSE environment increases, so does the time required to maintain both the servers and the clients that use them. In Chapter 1, we also talked about MetaFrame and how it overcomes this scalability issue through a feature known as **load balancing**.

Although there are many references to Terminal Server being able to service 100 or more users on a single server, there are many reasons that more than a single Terminal Server would be required for an environment with 100 or fewer people. Issues such as fault tolerance and availability are the most common. Although you could run a hundred users off a single Terminal Server, would many businesses be comfortable with such a scenario? A problem with that server would impact all the users and would keep them from working until the problem was resolved. What if their job function is considered mission-critical? In this case, the option to use a single large server is not a viable one.

A logical alternative to this would be to introduce additional servers for redundancy and to distribute the user load to reduce the impact of a server failure. Two servers with a capacity of 50 each would be better than all users on a single server. A failure would leave half of the users instead of all users out of operation until the problem was resolved. Two servers with a capacity of 100 users each would allow you to distribute your 100 users across two servers, but would still be able to accommodate the entire environment on a single server if one is lost. The downside to this is that you are paying double the cost to help improve availability. With TSE alone, you are also creating a more complex environment to manage. MetaFrame’s load balancing allows you to introduce more servers of smaller capacity while maintaining more manageability. Consider four servers, each with a capacity of 50. You would now be able to lose two servers and still run your 100 users on the remaining two. It is not until the third is lost that you will be impacted.

The issue with TSE alone is that the more terminal servers you add to your environment, the more difficult it becomes to manage them effectively. The scalability threshold for TSE will be the limiting factor in how many servers you have in your environment. MetaFrame is not restricted by this threshold.

---

**Author’s Note**

The actual number of TSE servers you can have before reaching the scalability threshold will vary slightly, depending mainly on the applications in use, the redundancy requirements, and how many TSE users there are. In general, anything beyond three or
four servers becomes a client management issue, and anything beyond 8-10 becomes a system maintenance issue.

Really, the limiting factor to scalability with TSE is client management. As soon as the size of your environment begins to have an impact on the users’ ability to work, then you have lost the advantages that thin client has to offer. MetaFrame’s load balancing helps to overcome this limitation.

Heterogeneous Computing Environments

Server and client manageability are not the only issues to overcome if you want to be able to scale TSE to run within a corporate enterprise. The issue of the heterogeneous computing environment is in many ways an even larger obstacle to overcome. There are many enterprise computing environments that are not completely standardized with a Windows operating system and a TCP/IP network implementation, both of which are currently required to run the TSE client. Few organizations have the luxury of being able (or wanting) to standardize their entire company on these requirements simply to introduce Terminal Server. MetaFrame’s support for a wide range of clients and network protocols provides Terminal Server access to a heterogeneous computing environment.

Management and Configuration Tools

MetaFrame also provides a number of management and configuration tools that help simplify the administration of a large-scale TSE/MetaFrame implementation. Functionality such as session shadowing, seamless windows, and application linking and embedding (which allows an application to be executed through a Web page) provide you with additional flexibility in deciding how you want to deploy and support your thin-client environment.

In this chapter, we will discuss the specific features of MetaFrame that fall into the categories we have just discussed. We will revisit these features later in the book when we talk about planning and performing your Terminal Server implementation (in Chapter 8, "Defining Policies and Procedures," and Chapter 11, "Security and Auditing").

Independent Computing Architecture (ICA)

Citrix’s Independent Computing Architecture (ICA) is the foundation upon which both MetaFrame and WinFrame are built. ICA is the distributed presentation services protocol used to deliver a number of MetaFrame’s features including screen information, keyboard and mouse input, and even sound. There are three components that make up the ICA environment:

Author’s Note

Many of the ICA features that exist for MetaFrame also exist in Citrix’s WinFrame product, such as session shadowing and application publishing. WinFrame and MetaFrame servers can coexist in the same environment and be managed using the same tools.

- MetaFrame server--Any information that is to be sent to or received from the client is transmitted using the ICA protocol running on top of a network protocol such as TCP/IP, IPX, SPX, or NetBEUI. This includes screen, mouse, and keyboard information in addition to other features such as sound and client and printer mappings. The ICA server uses a technology
called SpeedScreen to decrease the time required to update the graphics on a user’s MetaFrame desktop. We will talk about SpeedScreen shortly.

- **Presentation Services Protocol**—The ICA protocol bundles and transports information such as keystrokes, mouse actions, and screen updates to the ICA client and also receives and processes information that the clients have sent back.

- **ICA client**—The client receives the ICA information, translating it into the application’s interface, which is displayed to the user on his or her computer screen. It also takes actions performed by the users and bundles that information into an ICA packet to be sent back to the MetaFrame server for processing. The interface information is presented to the user while 100 percent of the application processing is happening on the server. The ICA client can be any device that supports the ICA protocol.

### ICA SpeedScreen

*SpeedScreen* is a server-side component of ICA that controls how screen information is selected and queued for sending. Figure 3.1 demonstrates the SpeedScreen components and how they function.

**FIGURE 3.1 ICA SpeedScreen.**

When screen information is updated, the Screen Evaluator will determine whether the updated information should be queued to send. This component can significantly reduce the amount of retransmitted information, particularly when an application repaints all of its screen data regardless of the change. It also helps by not sending screen information that is currently hidden behind another window. Once the Evaluator has determined what will be queued, it is then passed on to the Packet Builder that bundles the packets and sends them to the Session. SpeedScreen can make a noticeable improvement, particularly when users are accessing the system through dialup.

### ICA Browsing

Citrix’s ICA is managed by the ICA Browser service that runs on every MetaFrame (and WinFrame) server. This service maintains information on all of the available MetaFrame servers in the environment. A separate set of data is maintained for each of the supported network protocols (TCP/IP, IPX, and NetBEUI).

In addition to server information that is used by the ICA client to connect to a MetaFrame server, the ICA Browser also maintains server information such as MetaFrame licenses, administrative user information, and performance statistics. The ICA Browser services on each MetaFrame server work together to create an environment consisting of the following components:

- A master browser
- All of the member (non-master) browsers
- The ICA client (on the desktop)

### Master and Member Browsers
In a physical network, only one master browser will exist for each network protocol in use. For example, if you have both TCP/IP and IPX connections, you will have a separate master browser for each. Master browsers for different protocols do not have to be on the same MetaFrame server.

Figure 3.2 shows two networks separated by a router, each containing MetaFrame servers. The one network has a master browser each for TCP/IP and IPX, whereas the other only has a master browser for TCP/IP. Because the networks are physically separated, a TCP/IP master browser will exist in each.

A master browser is selected through a process known as a master browser election. Once the master browser has been chosen, all other browsers automatically become member browsers.

**FIGURE 3.2 ICA master browsers.**

---

**Tip**

A master browser election is initiated when:

- A MetaFrame or WinFrame server is started up.
- The current master browser does not respond to the requests of a member browser or another master browser through an ICA gateway.
- The current master browser does not respond to a MetaFrame/WinFrame client request.
- Two or more master browsers are detected on the same network.

When an election does occur, the master browser is selected based on the following criteria, which are shown from highest to lowest in precedence:

1. The browser has been configured to run as a master browser in the Registry.

2. The ICA browser has the highest version number.

3. The MetaFrame/WinFrame server is an NT domain controller.

4. The ICA browser service has been running longer than any other.

5. The name of the MetaFrame/WinFrame server is alphabetically lower than all others.

When an election is initiated, the initiating browser will broadcast its complete set of election criteria. If another browser has higher criteria, it will broadcast its own. When no higher criteria can be broadcast, the last browser to send the broadcast is elected the master browser.

Consider this example. There are three MetaFrame servers (MF1, MF2, MF3) and one WinFrame server (WF1) in a small environment. The four servers have the following criteria, with the three MetaFrame servers already running and MF2 currently the
**master browser:**

<table>
<thead>
<tr>
<th>Criteria</th>
<th>MF1</th>
<th>MF2</th>
<th>MF3</th>
<th>WF1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configured as master browser?</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ICA Browser version number</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>A domain controller?</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Uptime</td>
<td>21</td>
<td>23</td>
<td>7</td>
<td>0</td>
</tr>
</tbody>
</table>

When the WinFrame server WF1 is started, it will immediately initiate an election and broadcast its election criteria. Immediately after that, MF1 broadcasts its criteria, which is higher than WF1 because of the browser version number. MF3 will not broadcast because MF1’s uptime was higher. Finally MF2 responds because its uptime is higher than MF1’s. MF2 remains the master browser and WF1 becomes a member browser in the environment.

The master browser will maintain the following information:

- All available MetaFrame servers
- All available published applications
- All pooled MetaFrame licenses
- Load information used for load balancing

This information is updated periodically by all member browsers on the same network. Referring back to Figure 3.2, the member browsers on each network will send update information to the master browser on their network, but not to the master browser on the other network. Information maintained by the two masters can be shared with each other using something known as an ICA gateway, which we will discuss shortly.

**The ICA Client**

In order for an ICA client to get the address of a server or published application to connect to, it must be able to locate the master browser, either directly or through a MetaFrame server on the same network as the master browser. The client can do this in two ways:

- By sending broadcast packets--The client will broadcast a Get Master Address packet and wait for the first MetaFrame server to respond.
- By sending the request directly to a MetaFrame server--The server is specified in the Server Location (under Options, Settings in the Remote Application Manager) for the ICA client or in the ICA file currently being used. (See Chapter 14, "Client Installation and User Configuration," for information on how to configure this.)

When the address of the master browser is known (either through a broadcast or directly), the client can request the following from this server:
A list of all available MetaFrame servers

A list of all available published applications

The address of the server or published application that the client wants to connect with

**ICA Gateways**

Figure 3.2 shows the following scenario. You have a user on Network A who wants to run a published application on Network B. Because the user is on Network A, he or she will not be able to locate the master browser for Network B with broadcasts because they will not be passed by the router across from A to B. The other option is to configure the client to list one or more servers in Network B in the Server Location list. The problem with this is that now the user will not be able to access published applications in Network A because all requests for a master browser will be passed across to the server listed in Network B, where that master browser cannot see published applications in Network A.

Citrix provides a solution to this problem using what is known as an *ICA gateway*. An ICA gateway is established between the two networks to allow the master browsers in each to share information with each other. The master browsers in each network can then maintain server and published application information about the other network. With an ICA gateway in place between Network A and Network B, the user in Network A will be able to access the published application in B without having to specify any servers in the Server Location. This way he or she will also be able to access the published application in Network A.

**Tip**

To eliminate broadcasts in your MetaFrame environment, you should specify multiple servers in the Server Location for a client. By providing multiple servers, you ensure that if one is down, they will be able to contact another. We discuss configuring client information in Chapter 14.

An ICA gateway is configured using the Application Configuration utility and is discussed in Chapter 13, "MetaFrame Installation and Configuration." When a gateway is created, the local server is responsible for establishing the connection between the master browsers on the two networks. The two MetaFrame servers selected to establish the ICA gateway do not have to be master browsers. These servers will be responsible for creating the link between the two master browsers. Multiple gateway pairs can be created to provide redundancy.

**Tip**

For ICA gateways to function properly, the network router must pass ICA Browser traffic between the networks as follows:

<table>
<thead>
<tr>
<th>Network Protocol</th>
<th>Traffic to Route</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP/IP</td>
<td>UDP datagrams</td>
</tr>
<tr>
<td>IPX Raw</td>
<td>IPX packets</td>
</tr>
</tbody>
</table>
MetaFrame Management Tools

We now look at the MetaFrame management tools that supplement those delivered with Terminal Server. These tools provide robust support for scaling both Terminal Servers and their users within the enterprise.

Automatic Client Update

The key component to accessing both a Terminal Server and a MetaFrame server is the client software. When you are dealing with a small environment, manually having to install and update the client software when necessary may not be that much of a problem, but it can quickly become one as the number of users increases to enterprise levels.

Author's Note
Whenever we refer to a MetaFrame server, we are talking about a Terminal Server with the MetaFrame component installed.

Within an enterprise environment, an upgrade of the client can be a major project. An upgrade may be required for a number of reasons such as to resolve a problem or add new functionality. In either case, until the upgrade has been distributed to all users, the problem may still exist, or the functionality may not be available. There is also the possibility that someone may be missed during the upgrade and as a result may have problems the next time he or she tries to access the server.

This support issue has a direct impact on two of the benefits of TSE, which are centralized management and reduced maintenance costs. MetaFrame addresses this issue with the Automatic Client Update (ACU) tool.

Author's Note
To use the ACU utility, the ICA client must be version 581 or higher. Older ICA clients, such as those that ship with WinFrame, will not be recognized by this utility.

This utility provides the ability to automatically update the Citrix ICA client on a user’s desktop when he or she logs on to a MetaFrame server. The administrator is able to schedule the automatic download and installation of the client software to the user’s device when that user next logs on to the MetaFrame server. This provides a simple solution to the issue of keeping the version of the ICA client up-to-date, and administrators no longer have to worry whether a client device is running the proper version.

Although other software update tools such as Microsoft’s Systems Management Server (SMS) also provide the ability to automatically update clients, this MetaFrame utility allows the automatic update of all ICA clients, including supported ICA Windows-based terminals.

Tip
Support for ICA terminals is dependent on the terminal manufacturer.

In Chapter 14, we will talk more about ACU and how to configure it for use in your environment.
The ReadyConnect Client Installation Set

More a process than a utility, the *ReadyConnect client installation sets* are a way of creating preconfigured client software distribution files that can be used for first-time deployment of ICA clients. After this initial installation, all future updates can be performed using the ACU utility described earlier. The ReadyConnect installation files can help to greatly simplify the deployment of the ICA client in your organization.

In Chapter 14, we will describe how to create a ReadyConnect set of ICA client installation files to be used both for new installs as well as updating existing clients using ACU.

Application Publishing

Although Terminal Server provides the Client Connection Manager (CCM), which allows you to specify a single application for a user to run on a Terminal Server, Citrix allows even easier access to applications installed on a MetaFrame server through *Application Publishing*. Unlike the CCM, which requires you to configure the connection information for the application on the client, Application Publishing specifies the executable and the working directory on the server so that the client never has to know where the program is actually being run from. The application is then "published," so that the ICA clients will be able to see the advertised application and connect and execute it.

Figure 3.3 shows a connection to a published application being created in the Remote Application Manager (RAM) tool that comes with the ICA clients. Notice that instead of a server being selected, a list of published applications is being requested. All applications currently being published are listed in the drop-down list box. As we discussed earlier in this chapter, the master browser maintains and provides this list of published applications to clients that request it.

**FIGURE 3.3** A published application connection being created in the Remote Application Manager on a MetaFrame client.

Once an entry exists in the RAM, you can create a shortcut to the published application on the desktop or the Start menu. This allows a user to immediately launch the published application instead of running the RAM first. We provide a detailed explanation on how to create published applications and their desktop shortcuts in Chapter 18, "Application Integration."

Application Launching and Embedding (ALE)

It certainly isn’t news when we say that the Internet has undergone phenomenal growth in the last five years and continues to do so today. The use of the Web browser as a tool to access information not only on the Internet, but also on internal corporate intranets, is becoming more and more common with each passing day.

The *Application Launching and Embedding (ALE)* features of MetaFrame provide the tool for integrating Windows-based applications and Web computing. With ALE, you can publish applications on a MetaFrame server, while having a Web server provide the Web pages from which the published application can be launched or embedded.
To launch an application from a Web page, you just click on a hyperlink to an ICA file that starts up the application window, which can then be run as if it had been installed locally on your computer. If an application is *embedded*, it is placed within the boundaries of the Web browser and does not have a separate window on your desktop. Figure 3.4 shows the standard calculator application that comes with Terminal Server embedded in a Web page.

**FIGURE 3.4 The published Calculator application embedded in a Web page.**

The easiest way to launch or embed an application in a Web page is to use the MetaFrame ALE Wizard. This wizard will guide you through the steps required to generate the HTML page. The ALE Wizard is accessed from the same Application Configuration utility that is used to configure your published applications. In Chapter 16, "MetaFrame Web Computing," we will discuss ALE in detail.

**Anonymous User Accounts**

In many situations, a person does not require an explicit logon to access an application. This is most often the case when dealing with the Internet, where potentially thousands of people may visit your site. It would not be possible to provide some user ID or password for each person, but at the same time you want to grant them access that is as restrictive as possible.

Citrix’s solution to this problem is to incorporate the idea of the *anonymous user*, allowing people to access published applications (most often embedded in a Web page) without requiring an explicit logon. The benefit is that it allows a way of enforcing restricted access without providing the person with a logon prompt. By forcing all users to access your environment using an anonymous account, what you lose in auditing abilities, you gain through restricted access. By not presenting a logon window to an Internet user, you are denying potential hackers an easy means of attempting unauthorized access to your environment.

Anonymous user accounts are local accounts created on the MetaFrame server that have special guest permissions by default. For security reasons, these accounts should not be granted any additional permissions, or made a member of any other group besides the Anonymous group. If a MetaFrame server has been created as a Domain Controller, anonymous accounts cannot be created on that server. As an added security (and privacy) feature, any desktop settings, user-specific files, or other resources owned by that anonymous user are discarded at the end of that user’s session.

When a person starts an application as an anonymous user, the MetaFrame server will select a user ID from the pool of anonymous accounts that is not currently logged on. During the installation of the MetaFrame software, an anonymous account is created for each user license that exists. The accounts are named Anon000, Anon001, Anon002, and so on, as shown in Figure 3.5. If additional licenses are added later, the additional anonymous accounts are *not* created, and you will need to add them manually using User Manager.

**FIGURE 3.5 Anonymous accounts on a MetaFrame server.**

---

*Tip*

The easiest (and safest) way to create new anonymous user accounts is simply to copy one that already exists. You should ensure that the account you are copying does not have any extra permissions assigned that you are not aware of. After the accounts have
been created, they will not be usable by the MetaFrame server until it has been rebooted. By usable we mean that the MetaFrame server will not be able to automatically assign one of them to an anonymous logon session.

The following restrictions are also set on anonymous accounts by default:

- A ten-minute idle timeout. After 10 minutes of inactivity, the account is disconnected.

- On a broken or timed-out connection, log off. This must be enforced to ensure that anonymous sessions are not being tied up by users who are no longer connected. Typically, a broken or timed-out connection will occur when a person exits the browser without properly exiting the published application he or she is accessing.

- Users cannot change the password. This guards against denial-of-service attacks where someone could change the password on an anonymous connection and in so doing render it unavailable for anonymous logons.

**Warning**

If you are going to change the permissions on an anonymous account, make sure you make the change for all of them. Users have no control over which anonymous account they are assigned to when they connect, so there is no way to enable certain permissions for some anonymous accounts but not others. You have no way of controlling access to logon by using one of these accounts.

Never make a security change on an anonymous account unless you are 100 percent certain that it will not expose your environment to any unnecessary risk.

**Load Balancing**

As we mentioned earlier when discussing published applications, in order to publish an application across multiple servers, you need to have a Citrix load-balancing license installed on each server that will participate. Load balancing is an essential part of providing scalability in your MetaFrame environment.

Load balancing enables you to create server farms from which users access applications without having to explicitly identify the server to run it on. When a published application has been configured to participate in a load-balanced environment, and it is launched from an ICA client, load balancing will determine which server is least loaded and automatically direct the user to run the application from that server. This allows multiple users to be spread across the available servers in the farm, and to make the best use of the resources available. We discuss the configuration and tuning of load balancing in Chapter 13.

**Session Shadowing**

In Chapter 2, "What Is NT Server, Terminal Server Edition?" we discussed the Terminal Server Administration utility and how it allowed you to view information such as users or sessions on any of the available terminal servers. Citrix adds additional functionality to this application as well as providing their own customized version called *MetaFrame Administration*. The added components are as follows:
• License tab--Displays the total number of available licenses for the MetaFrame servers on the network. See "Licensing" later in this chapter for more information on MetaFrame licensing.

• Session shadowing--This allows an administrator or other authorized person to view another user’s session in real time. It also provides them with the ability to take control of the keyboard and mouse in that user’s session.

• Published Application Management--This allows you to view all of the published applications in the environment, what servers they are configured on, and what users are currently accessing them. We will provide more information on this in Chapter 18.

By default, only members of the Administrator’s group have access to shadow another user. This access can be granted to any group or individual user. In Chapter 11, we describe how to enable shadowing permissions on ICA Terminal Server connections through the Terminal Server Connection Configuration utility as well as User Manager for Domains.

You shadow a user through the Terminal Server Administration utility. You do so by clicking the User tab in the right pane, right-clicking the user you want to shadow, and selecting the Shadow option. You will then be asked to select the Hotkey Reminder. This is the key combination that you press to terminate shadowing and return to your own session. Figure 3.6 shows the MetaFrame Administration utility and the shadow option highlighted for a user.

FIGURE 3.6 Shadowing through the MetaFrame Administration utility.

If notification is turned on, the user will be informed that you want to shadow him or her, and he or she has to accept to allow shadowing to begin. If notification is turned off, you will begin shadowing without notifying the user. On the user’s terminal, he or she will notice a momentary flicker of the display as MetaFrame synchronizes the client’s session screen with the administrator’s screen.

Here are some factors to note regarding shadowing a user’s session:

• You cannot shadow a session that has a larger window resolution or higher color depth than your own. For example, if you have an 800¥600 resolution with 16 colors, you cannot shadow any session that is running higher than 800¥600, or that has greater than 16 colors. This is an important point to remember, since you will need to ensure that the support staff has sufficient screen resolution and color depth to be able to shadow users to help with any issues they may have.

• Shadowing is not supported on RDP clients, only on ICA clients. This means that an RDP client cannot shadow an ICA client, and vice versa.

• You cannot shadow the Terminal Server console. You also cannot shadow another session from the console. In order to shadow from the console, you must install the ICA client on the server and then log on to the server using this client.

Client Desktop Integration
MetaFrame delivers its own ICA client, which includes an impressive assortment of desktop integration features that make it difficult at times to determine whether you are actually running remotely or not. Some of these features include audio support, client printer and driver mappings, and local/remote clipboards.

**Seamless Windows**

The most impressive integration feature is the *seamless windows connection*. Seamless windows is available only for the 32-bit Windows ICA Client. Seamless windows allows you to run multiple published applications on your client, each displaying its application window directly on the local desktop. Each window is completely resizable and behaves like a local application, appearing on the taskbar and responding to the Alt+Tab application-switching keystrokes.

When you start an application running in a seamless window on your desktop, the following will happen:

- If it’s not already running, the ICA Seamless Connection Center (ISCC) is started. This appears as a little red computer icon at the far right side of the taskbar. This application manages your seamless windows connections.

- The connection with the MetaFrame server publishing your application is made. The ISCC will determine whether you have already been authenticated on this server, and if so, you will immediately start the application. If you have not been authenticated already and if the connection type is explicit, you will be prompted for your user ID and password. If the connection type is anonymous, an anonymous user account will be assigned to you, and you will log on.

---

**Author’s Note**

Because a user must be authenticated on each MetaFrame server that he or she connects to, if the user is accessing multiple published applications all located on different servers, he or she may be required to enter his or her ID and password for each (if authentication is explicit). Users will very quickly become annoyed with having to continuously enter their ID and password whenever they start an application.

The fact is that there isn’t an easy solution to this problem. Providing a single sign-on across multiple MetaFrame servers will have to be approached by Citrix very carefully to ensure that it does not compromise the security of the environment. Citrix is conscious of the fact that administrators want the single sign-on capabilities and will hopefully have a solution available in the next release of MetaFrame (or possibly sooner). In the meantime, you will have to look at your published application configuration carefully. You may want to look at using the anonymous logon feature to give users access to an application that has its own built-in security. Again, you will want to approach this carefully to ensure that you are not compromising the security of your environment.

If pooled licenses are available, one of them will be taken for your seamless windows session. Regardless of the number of seamless windows you have open, only one pooled Citrix license will be used. If pooled licenses have not been configured in the environment, one license will be taken for each of the MetaFrame servers that you are connected to.
Once you have been authenticated and a license assigned, the application will be started and will appear on your desktop.

Figure 3.7 shows two seamless windows applications open on the desktop (Notepad and Calculator) along with ISCC. Notice how the ISCC lists all the servers that you are currently connected to and the application(s) running on each. It also displays the user ID that you used to log on.

**FIGURE 3.7** The ICA Seamless Connection Center for managing seamless windows sessions.

**Local/Remote Clipboard**

MetaFrame provides a seamless integration of the Clipboard in your TSE session, and the Clipboard on your desktop. Using the standard cut-and-paste functions, you can move data between the two environments as if you were moving it between local applications. This functionality is available only between ICA clients and MetaFrame sessions. RDP clients do not currently support this function.

**Client Mappings**

To further integrate the ICA client into your standard desktop, MetaFrame provides robust client device mapping functionality. This allows for easy transfer of information between the client device and the terminal server. The following sections list the devices that MetaFrame maps.

**Drives**

Depending on the MetaFrame installation, when users connect to a Terminal Server, they will either have their local drives map to the corresponding drive letters in their MetaFrame session or their local drives will map to alternate drive letters such as X:, Y:, and Z:.

*Tip*

We recommend always reassigning your MetaFrame OS and data drive letters to the higher drive letters so that client drives map to the same drives in their MetaFrame session. For example, the C: drive in the MetaFrame session maps back to the local C: drive on the user’s desktop. This helps to eliminate confusion and is much more intuitive for most users. In Chapter 13, we discuss how to configure MetaFrame to allow the proper client drive mappings.

Figure 3.8 shows the My Computer window with the appropriate drive mappings in a Terminal Server session. You can see that the drive letters A through G have been assigned to the corresponding drive letters on the local machine. The letters W: through Z: represent physical drives on the terminal server itself. These have been remapped to avoid conflict with the client drive mappings.

The automatic mapping of client drives can be turned on or off from either the Connection Configuration utility or from User Manager for Domains.

*Warning*

Although automatic mapping of client drives can be turned off, clients are still able to manually map to their local drives unless additional steps are taken. We discuss client
drive mappings and security in Chapter 11.

FIGURE 3.8 My Computer showing the client drive mappings in a MetaFrame session.

Printers

Printer mappings behave very similarly to client drive mappings and are configured in the same location in CCM or User Manager for Domains. There are actually two options:

- One for mapping the client printers
- The other for using the client default as the TSE default

When mapping of client printers is enabled, it allows a user to connect to a MetaFrame server and automatically have their locally configured printers available for printing. This includes locally attached printers and network printers that are configured on the local client.

Figure 3.9 shows the Printers folder in a TSE session and the list of printers available. The only listed printer is actually the client printer that was autocreated by MetaFrame when the user logged on. The user was not required to go in and add a new printer manually. This is a network printer that has been set up on the local computer.

Of interest is the actual name for the printer. As you can see, the actual printer name contains a pound sign (#) before the double backslashes. The name before the pound sign is the client’s ICA session name. This is configured through the MetaFrame Remote Application Manager program on the client.

It is important that all MetaFrame clients have unique client names. This unique name is used to access the client devices that have been established for the user’s session. We provide information on printer configuration in Chapter 14.

FIGURE 3.9 A Terminal Server Printers dialog box containing a client printer that was autocreated by MetaFrame.

Warning

In an enterprise environment with a large number of users, it is more bandwidth-efficient to have the user’s printers configured on the server and not through local client printer mappings. Printers can be configured on the server in one of three ways:

- Users map their own printers manually--This is done by selecting Add Printer from the Printers folder.
- Through user profiles--Users are assigned NT profiles that already contain their printer configurations.
- Through logon scripts--NT logon scripts are run when the user logs on to map drives.

When printers are configured in the MetaFrame session, it allows the print jobs to be
spooled on the MetaFrame server and then sent across the network to the printer. When printers are accessed through client mappings, the print job is sent from the MetaFrame server to the client where it is spooled and then sent across the network again to the printer. This essentially causes the print job to make an extra trip on the network. This may cause bandwidth issues in a large organization with many MetaFrame users.

Ports

Both COM and LPT port mapping is supported. This allows devices attached to a local client’s COM/LPT port to be accessible during a MetaFrame session. The mapping of these devices can be disabled using the Terminal Server Connection Configuration utility and changing the client settings for a connection. Issuing the command NET USE from a command prompt in TSE will list the client-mapped ports in addition to all drive mappings, as shown in Figure 3.10.

FIGURE 3.10 Port mappings listed using the NET USE command.

Client Audio Support

MetaFrame also supports client audio. Like the other mapping functions, it can be disabled in the Terminal Server Connection configuration utility as well as being turned on or off from within the connection properties of Remote Application Manager.

Tip
In order to use client audio, there must be a SoundBlaster-compatible sound card in the client computer. Unless sound is a true necessity, we recommend not using it because of the added network bandwidth.

MetaFrame Licensing

MetaFrame licensing is different from the standard Terminal Server licensing in that you are purchasing concurrent or server-based user licenses instead of client-based licenses. When you install 15 Citrix user licenses on a MetaFrame server, this allows up to 15 concurrent ICA connections at any one time. They do not have to be the same 15 users every time. You could install the ICA client on 100 desktops and any 15 out of those 100 would be able to log on at any given time.

Warning
MetaFrame licenses are not a replacement for the required Microsoft licenses. You are still required to purchase all necessary Terminal Server, NT Workstation, and client access licenses.

Citrix also allows you to carry these licenses across multiple MetaFrame servers, using what is known as license pooling. With license pooling, you could purchase 30 licenses for five MetaFrame servers, which then allows any combination of 30 users to log on to the available MetaFrame servers. License pooling allows you to implement redundancy into your environment by not having to tie licenses to each of your MetaFrame servers.

Tip
The ICA Master browser on the network maintains the pooled license counter.

Author’s Note
Licenses can be pooled between MetaFrame and WinFrame 1.7 servers. This allows any existing WinFrame licenses to be available in your MetaFrame environment. You are still required to purchase the base MetaFrame product with the 15-user license. For information on MetaFrame pricing, contact a Citrix authorized reseller or go to the Citrix Web site (http://www.citrix.com/csn/pricing.htm).