Global Information Assurance Certification Paper

Secure Foundations for Microsoft Terminal Services

By: Matthew General

GSEC Practical V1.4b
Option 1
January 11th, 2004

Table Of Contents

Abstract

Today's competitive business climate requires organizations to deliver applications and information through ever increasingly complex channels. Mobile workforces require access to information from the field; branch offices demand applications from the corporate site; and IT Departments are being challenged to produce compliant, sound, and cost-conscious disaster recovery solutions. With a myriad of options, many organizations are turning towards Terminal Services to solve these access and deployment problems.

What is Terminal Services? According to Microsoft,

> "Terminal Services provides a multisession environment that allows client devices to access a virtual Windows 2000 Professional desktop session and Windows-based programs running on the server. Terminal Services allows multiple users to be connected interactively to a computer and to the display of desktops and applications to remote computers." [1]

Terminal Services can be utilized to rapidly deploy applications, provide access to information over low bandwidth links, prolong the life of older hardware, centralize administration, and much more. However, this work-horse technology is not a quick fix panacea. Terminal Services is a complex operating environment that is both powerful and robust, exposing access to many resources, applications, and data. Simply put, Terminal Services creates both solutions and risks.

What is a risk in the Terminal Services environment? Failure in a Terminal Services environment can lead to several types of vulnerabilities. An improperly configured Terminal Server could allow malicious users to gain access to the system. These malicious users could then gain access to sensitive information on that system, or neighboring systems. They could create a denial of service against legitimate users, preventing them from working. A failure could lead to a complete compromise that would allow a malicious user to run code of his choice and effectively own the server. Ultimately, a malicious user could establish a platform from which they can have a very clear, unobstructed view of the rest of the network, aided by the convenience of a complete Windows desktop.

Terminal Servers create risks. This document is intended to establish a collection of information in order to successfully design a secure Terminal Services deployment which minimizes risk exposure. This document covers most of the concerns an administrator should have to evaluate when deploying terminal services to their organization by discussing key concepts and providing references to the subject matter for further, in-depth analysis.

Summary

This document will go over the key concepts to designing a secure Terminal Services deployment. It is broken down in sections that generally align with setting up the hardware, installing the system, and configuration of the components. However, much of the information discusses concepts, as there isn't a single correct answer, but rather, many different ways to build a secure system. The core topics I will cover are as follows:

1. System Setup & Operating System Installation – Some brief notes on the requisites and recommended Hardware and Operating System criteria.

2. Network Design – A discussion on the physical network and its associated impact.

3. Remote Desktop Protocol – The native Terminal Services protocol for client/server communications and its security.

4. Terminal Services Connection Configuration – An in-depth look at the key tool used to configure a Terminal Server.

5. The USRLOGON.CMD Script – A default component of a Terminal Server that could pose a risk and how to deal with it.

6. Active Directory Security Group: Terminal Server Users – How to create a group in Active Directory which logically groups and grants access to your users.

7. Active Directory Group Policy Objects – Perhaps the single most important tool to configure security on your Terminal Server.

8. Group Policy Software Restrictions – A component of policies that contributes substantially to maintaining the integrity of the environment.

System Setup & OS Installation

To begin with, some basic principals of Windows Server installation and configuration apply to Terminal Servers as much as with any Windows Server installation. Many of the principals that dictate any good server setup or installation should apply. A few of the following items are mandatory, while others are more or less optional, but highly recommended. This isn't, however, to be interpreted as a guide to securing servers in general, so I've only covered a few areas, in brief, that have some impact on the Terminal Services design.

The following list is the requisite configuration.

1. The server should be built on a solid, reliable platform. Enough cannot be said about the importance of a well built, well architected server platform. I suggest a name brand system with top of the line components. Short change yourself, and it will cost you in the end. "How does this impact security," you might ask? One example is that a poorly built system will lead to faults, faults will lead to extensive troubleshooting, and troubleshooting invariably leads to sacrificing configuration settings in favor of diagnosing problems. Even for short periods of time, a vulnerable server only takes seconds to penetrate. It never pays to skimp on mission critical equipment.

2. The server should be located in a secure facility. A data center, a computer room, or even a secured closet; any place that limits physical access. In the words of Microsoft Corporation, "If a bad guy has unrestricted physical access to your computer, it's not your computer anymore."[2]

3. When it comes time to install the Windows operating system, a Terminal Server should always be installed onto a system with NTFS partitions. FAT, or FAT32 partitions simply don't have the security required to prevent unauthorized access and auditing. In some environments, it may be considered acceptable or even practiced, to install a FAT partition on the server for the sake of management, recovery, or deployment flexibility. Even these reasons are unacceptable for a Terminal Servers. Users of Terminal Servers have very intimate access with the server. It is important that all local drives contain the highest level of security.

The following items are the recommended list.

1. The server should be a member of an Active Directory Domain. Active Directory is the network administration system for a Microsoft Windows network. Keep in mind, this is not required for a Terminal Server installation, but it really should be. Many of the topics discussed in this document rely on Active Directory, and for the sake of this document, it is presumed that you will be working with it. Terminal Services without membership in an Active Directory Domain can be done, but I would only recommend it for a limited number of very specific reasons.

2. During installation of Terminal Services, you are prompted to select Permission Compatibility. Your options are **Full Security** or **Relaxed Security**. The options affect some default permissions applied to the file system and registry. Full Security is recommended. However, from time-to-time, some legacy applications will not work correctly in a Full Security deployment, and the administrator must choose Relaxed Security in order for the application to run correctly. As more and more applications are becoming Windows 2000 and 2003 aware, I find that the need for Relaxed Security has been diminished to only a handful of applications. Never the less, every

installation should begin with Full Security, and only resort to Relaxed Security if required. Even though this setting is presented during the installation of Terminal Services, it can be changed at a later date without having to reinstall Terminal Services.

Network Design and Security

Physical Network Architecture

Networking a Terminal Server has several considerations of which the administrator must be aware. First is where the Terminal Server is logically located within the network in proximity to the other servers and the clients. Second, at a deeper layer, is the security of the protocols between the client and the server.

Obviously, requirements vary for every environment, so our scenarios will be limited. However this overview of Terminal Services networking considerations should support our desire to establish a foundation for any environment.

In any case, no matter what environment you are working in, clients must be able to reach the server, and the server must be able to reach the data. (data being defined as our File Servers, Database Servers, or other secure hosts) So herein lies our basis: Users, Terminal Servers, and Data Servers.

Common best practices in the security arena support isolation of resources. I support the theory of isolation as well. Isolation allows us to create a layer of defense among the users, the servers and the data or in other words, Defense in Depth. With that in mind, the best design for a terminal services deployment would be similar to the common three-tiered web services designs. Three-tiered isolation dictates that a firewall should be located between the users and the applications, and another firewall between the applications and the data. As such, a good Terminal Services architecture should also have a firewall between the users and the Terminal Servers. Furthermore, it is best that another firewall lie between the Terminal Servers and the data.

Terminal Servers are intended to host applications. They should reside in a DMZ or similar security layered segment of the network. Terminal servers can pose an element of risk, allowing authenticated users to get very near and intimate with the data and as such, should be isolated from the data. Clients, or end users, should be treated as un-trusted and the interaction between them and the servers should be limited.

Of course, this theoretical design is ideal, and may not always be practical. Often, terminal servers may have to be located in direct proximity to the data, but it is less secure and poses more risks.

The Data Center Model

The ultimate solution in this design is what I term, the "Data Center Model." This model assumes that all clients come from a wide variety of sources, local, branch, dial-up, and Internet based locations, or in other words, are universally connected. The Data Center Model distinctly separates the servers from the users. In effect, the Data Center Model establishes an internal network for data and a DMZ network for applications. The DMZ network interfaces with the Internet where your users reside and nothing but servers are connected to either the DMZ or Internal networks. The advantages to this design are:

1. Users are never directly attached to the protected network.

2. The protected network is highly accessible to users, no matter their location.

3. It simplifies end-user access by offering only a single destination regardless of their location.

4. The DMZ and protected networks are very portable.

The Data Center Model clearly has advantages, but requires careful planning and forethought. It returns the benefits of network control and reduces the inherent risks of client/server-based computing.

The location of the Terminal Server in proximity to the users and the data is something that must be analyzed carefully. There isn't one correct method but rather many different ways to design a deployment. I've given one example of how a system can be deployed utilizing isolation and tiered access. However, there are other ways to achieve similar results. No matter what the situation is, it is important to understand the impact and risks a Terminal Server will pose within a network.

Two Factor Authentication

Guaranteeing the authenticity of a user is a challenge in any environment. However, often a Terminal Server will be exposed to the Internet or other potentially hostile environments and simple username/password combinations are simply not secure enough. In these environments, I highly recommend some variety of two factor authentication. Two factor authentication is accomplished through the use of token or biometric based authentication systems. There are too many options to choose from to discuss the specifics of each. Nevertheless, for any server that is publicly connected, two factor authentication substantially reduces the chances of a compromise due to brute force or similar attacks.

The Remote Desktop Protocol

The Terminal Services native protocol between the clients and the server is called, “Remote Desktop Protocol,” or RDP. RDP, in simple terms, sends screenshots to the client, and returns keystrokes and mouse movements to the server. While it does much more than that in the background, for all intents an purposes, that is the majority of the payload. How it gets from the source to the destination and what is inside is our next concern.

RDP works like most standard TCP protocols, establishing an acknowledged, session oriented connection. But most administrators are concerned about what exactly is transported in the payload. Authentication occurs, keystrokes are sent, screens full of data are being transmitted; is this data safe?

> RDP uses RSA Security's RC4 cipher, a stream cipher designed to efficiently encrypt small amounts of data. RC4 is designed for secure communications over networks. Beginning with Windows 2000, administrators can choose to encrypt data using a 56- or 128-bit key.[3]

The short answer is reasonably, yes. Microsoft has enabled the RDP protocol to be encrypted in several different manners. Even the most basic encryption is based on the standard RSA RC4 encryption scheme, while the most complex is FIPS compliant encryption. By default, Terminal Servers are configured for “Client Compatible” encryption. In essence, Client Compatible means that the Terminal Server will adjust its encryption during connection establishment based on the maximum key strength supported by the client. Essentially, this is an auto-negotiating protocol that starts from the highest level of encryption and negotiates downward for compatibility. However, in certain circumstances, you may want to ensure everyone is using the highest level of compatible encryption. In such a case, you can adjust the Client Compatible setting to the desired level and only clients that can handle the required level will be allowed to connect.

I don’t recommend changing the default value unless you must guarantee connections meet a certain encryption value. The “Client Compatible” option should suffice for most installations and create a sufficient level of security. For those environments that require the highest level of encryption, I recommend using IPSEC encryption through a gateway, or VPN host.

It should also be noted that FIPS Compliancy encryption was introduced in RDP 5.1. This encrypts and decrypts the RDP data stream according the the Federal Information Processing Standards algorithm. This is the highest level of encryption supported by the RDP protocol suite. It provides a sound, 128 bit encryption scheme, appropriate for, and compliant with most government installations.

Finally, for the ultimate layer of security, you can implement an IPSec Policy to secure your RDP communications.

> Many organizations use standardized Internet Protocol security (IPSec) for network security. You can configure IPSec policies on Terminal servers to force all Terminal Services communications to be protected by IPSec.[4]

This ensures the most secure encryption between the client and the server, however it comes at a cost of performance. It should be thoroughly tested, as there is a greater impact on performance due to the overhead of the protocol, in addition to the burden of encryption that is placed on the processor. In general, I wouldn't recommend this unless it were a requirement for communications.

Terminal Services Connection Configuration

After the terminal server has been set up, the administrator must configure the basic connection configuration options. The connection configuration occurs in the Terminal Services Configuration tool. This tool is used to configure a substantial number of connection oriented options from user permissions to connection time-outs, and much more. I'll go over the application and some of the important settings an administrator should be concerned about.

To begin with, the interface displays two folders which contain distinct components of the environment. A list of connection profiles in the **Connections** folder and server specific configuration options in the **Server Settings** folder.

Connections

Connections make up a list of profiles. These profiles are bound to either a specific network card, or all network cards. There may only be one listener per network card, or a single listener bound to all network cards. The listener is the central object around most of the configuration settings within the Terminal Services Configuration tool. By default, you will only find one listener defined, bound to all network adapters, titled **RDP-tcp**. That profile will be where the majority of your options are configured. It should be noted that many of the settings discussed below may be set on a user-by-user basis, and is the default. However, Terminal Services Configuration allows you to override any user-specific settings.

General. The General tab allows you to set the **Encryption Level**. Discussed previously, the **Encryption Level** defaults to **Client Compatible** which sets the encryption between the client and the server to the highest key strength supported by the client.

Logon Settings. In the **Logon Settings** property tab, you can specify a specific set of credentials to always authenticate with during each session, or allow the logon credentials to be provided by the client. The default is **Use Client-Provided Logon Information**, which is also the recommended setting. Only under special circumstances would you want to use the **Always use the**

following logon information setting. Examples would be either kiosk or service provider type installations. I recommend that each logon be performed by a unique user. **Always use the following logon information** only authenticates using the specified credentials, which in turn causes each session to be running under the same user account context. This can create problems with some applications and printing, and is therefore not recommended. Finally, an option is also provided to **Always Prompt For Password**. This setting causes the password prompt to be forced on a user even if they set their password in the client application.

Sessions. The **Sessions** property sheet is used to configure timeout and reconnection settings. There are several types of time out's to be concerned with. There is a disconnected time-out, an active session limit, and finally an idle timeout. The active session limit measures the total allowed time on the server. If you wish to limit a single session to a fixed amount of time, say perhaps four hours, then you would set a four hour session time-out. Alternatively, if you are concerned about someone logging in and walking away from the computer, you could set an idle session limit. Idle time is measured from the last interaction with the server. Finally, there is the disconnected time out. If you get disconnected due to an interruption in your connection, or are disconnected as a result of an idle time-out, your session goes into a state of "Disconnected." In this state, your applications are still processing, but without a terminal to display the session. For a variety of reasons, an administrator may want to limit the time a session may remain disconnected. For which case, the disconnected time-out exists. The only recommendation for time-outs is that you assess your risks and tolerances. Regarding Reconnection settings, an administrator can control whether or not a user can reconnect from any client or only from the previous client. This is useful in certain circumstances to control where users are allowed to reconnect from. There isn't a single recommended setting for timeouts and reconnection other than adjusting them according to policy and tolerance.

Environment. Use the **Environment** settings to launch a specific program at the time the user logs on. This can be useful in kiosk mode, or when deploying a single application and access to a desktop isn't necessary. When in use, the application that is launched when the user logs on becomes the shell, and when they close the application, the session is terminated.

Remote Control. Terminal Services comes with a powerful feature that allows an administrator to remotely control or observe another active session. This can be an incredible tool to administer and support a Terminal Services environment. However, not all organizations may want to grant the administrator this power, or they may wish to further control how it works. The **Remote Control** properties allow an administrator to define specific options of the Remote Control feature.

Client Settings. The **Client Settings** property page allows you to configure what access the server has and can interact with the client device. It is important to know that by default, the Terminal Server connects a drive letter to the client

drives and creates printers and attaches them to the client device upon successful user logon. This could represent a security risk and therefore it is recommended that if this function isn't needed, it be disabled.

Network Adapter. As discussed previously, the connection configuration properties are based on the network adapter interface and can be set on a per-adapter basis, or grouped into all network adapters. This is where you can select how the configuration properties are bound. Additionally, you can set the maximum number of connections allowed through this adapter, or accept the default of unlimited.

Permissions. Primarily, connection permissions set which users or groups are allowed to login into the server and what auxiliary actions on the server they are allowed to perform. There are many important settings that should be modified or at least analyzed before putting the server into production. Most of the auxiliary permission settings can only be modified through this interface, as there aren't any group policy objects to set these options across the domain. These auxiliary settings can be found by selecting the **Advanced** option and then editing the permissions. The following is a list of permissions that may be administered:

- Query Information – Query sessions and servers for information.
- Set Information – Configure connection properties.
- Remote Control – View or actively control another user's session.
- Logon – log on to a session on the server.
- Logoff – Log of a user from a session.
- Message – Send a message to another user's session.
- Connect – Connect to another user's session.
- Disconnect – Disconnect a session.
- Virtual Channels – Use virtual channels.

By default, the Remote Desktop Users group should only be granted Allow access to **Logon**. All other settings grant a typical user more access to the system than what is generally required to perform day-to-day access.

Server Settings

The second component of the Terminal Services Configuration tool is the **Server Settings** component. It's a brief list of six server specific options to control certain

options on the Terminal Server. The following list is what is available to configure under **Server Settings**:

- **Delete Temporary Folders On Exit** – Deletes a user's temporary folders during logoff. By default this is enabled and should remain that way. This feature can and should only be adjusted in Group Policies.

- **Use Temporary Folders Per Session** – Controls whether or not the Terminal Server creates session specific temporary folders. By default this is enabled and should remain that way. This feature can and should also only be set in Group Policies.

- **Licensing** – The licensing mode sets how Terminal Services Licenses are issued. The options are **Per Device**, or **Per User**. There isn't a correct setting, and should be determined by how the users will use the terminal server. If you have many users sharing a few terminals, then per device is the best option, on the other hand, if you have many users that use more than one computer, then per user makes better sense.

- **Active Desktop** – Controls whether or not the **Active Desktop** shell is available. Generally, to conserve resources and increase security, it is best to leave this disabled.

- **Permission Compatibility** – This option is presented during the installation of Terminal Services and was discussed previously. I recommend that this be set for **Full Security** and only be adjusted to **Relaxed Security** if all attempts to obtain application compatibility have been exhausted.

- **Restrict Each User To One Session** – Restricting users to one session helps manage resources and assists users in reconnecting to disconnected sessions. It is recommended to leave this option in the default state of enabled, unless multiple simultaneous connections are desired.

The USRLOGON.CMD Script

The earlier version of Terminal Services, Windows NT 4.0 Terminal Server Edition, was an offshoot of Windows NT 4.0. While it was based on the same underlying architecture, it nevertheless created some compatibility problems with applications that weren't really designed for a multi-user environment. To compensate for this deficiency, Microsoft distributes a variety of "compatibility scripts" for particular widely adopted applications which slightly alter the environment for these applications. The compatibility scripts alter registry settings, modify environment variables, and add a new drive letter directed (or rooted) in the

users profile. As Terminal Services were incorporated into Windows 2000 as a single product, and subsequently Windows 2003, Microsoft has continued to include these compatibility scripts in order to maintain backward compatibility with older applications, and to maintain consistency during in-place upgrades.

The compatibility scripts are dependent upon one command file to perform scripted procedures during session initialization each time a user logs on. The USRLOGON.CMD file is the command file that processes registry analysis and changes, as well as setting some environment variables and map the additional drive letter.

Today it is becoming increasingly common to bypass the use of compatibly scripts because most new applications are Terminal Services aware or at least more compatible with Windows. As a side note, in order for an application to be granted the "Made for Windows" designation from Microsoft, it must work flawlessly in a Terminal Services installation.

It should be determined during the pilot phases of a terminal service deployment project whether or not compatibility scripts will be required. If not, leaving the USRLOGON.CMD file enabled could lead to a security risk and at the very least, add to the complexity of configuring software restrictions. I recommend that the USRLOGON.CMD script be disabled and any scripting or logon procedures should be enabled through Windows shell scripts initiated by Group Policy Logon scripts.

The USRLOGON.CMD file can be disabled by editing the registry in the location HKLM\Software\Microsoft\WindowsNT\CurrentVersion\WinLogon and removing the reference to USRLOGON.CMD in the **AppSetup** property key. If, at a later time, it is determined that this functionality will be required for some reason, it can be replaced.

Creating the Domain Security Group: Terminal Server Users

> By now, you are well aware that membership in the Remote Desktop Users group is critical to using a terminal server. For this reason, I recommend managing this group on all of your terminal servers through Group Policy.[5]

In Windows 2003 Terminal Services, each user who intends on using terminal services must be associated with the local servers **Remote Desktop Users** group. The addition of this feature enhances security by controlling access to the server through security groups. To simplify the administration of this feature in a domain environment, it is recommended you create a domain security group called **Terminal Server Users** and then associate that group to the local machine group. This configuration creates a central point of administration in the Active Directory tree to quickly and easily manage which users are able to use Terminal Services. Furthermore, after associating the group once, you will not have to add domain users to each individual terminal server.

In addition, to deploy this feature without having to modify each new Terminal Server in the farm, I recommend utilizing the **Restricted Groups** Security Setting within the Group Policy Object, "**Computer Policy**" discussed in the previous section. Restricted Groups were created to override the built-in local machine groups. They allow an administrator to define two properties for a given group. The two properties are **Members** and **Members of**. **Members** define the members of the group while **Members Of** define which other groups this group is a member of.

When this policy is in place, it overrides any existing group membership information on the local machine. For example, if the local machine group **Remote Desktop Users** has both users and power users as members, but the domain restricted group only has administrators, the effective **Remote Desktop Users** group membership will be administrators.

Group Policy Objects

Group Policy Objects are probably the most powerful tool to configure the Terminal Server operating system. Group Policy Objects are the successor to System Policies introduced with Windows NT. They are system configuration settings deployable to many computers throughout a Microsoft Active Directory Tree. They provide a hierarchical deployment based on a logical organization of objects. Terminal Servers benefit substantially through a well designed GPO Deployment. They allow an administrator to fine tune virtually every configurable setting and apply them to users and servers based on any conceivable characteristic. Group Policy Objects can be applied to particular groups, users, or computers depending on a variety of factors such geographical location or desired department.

The most flexible design for Terminal Servers is to isolate them within their own Active Directory Container object. It logically groups them together in order to apply customized configuration settings. This is important because most Terminal Servers are hardened substantially beyond what a typical workstation Group Policy Object would contain, yet much differently than how a typical domain controller or other member server may be configured. A terminal server is a breed of its own and should be administered as such.

Once the Terminal Server Active Directory Organization Unit has been created, you can easily add new Terminal Servers that will quickly inherit the hardened settings. This drastically reduces the time to roll out new servers and more importantly, creates consistency among them. Within the properties of the Organization Unit is stored the Group Policy Object Links. The Links refer to the policy templates which may be added to the organization unit. Often there none, or only one link in the list. Our design incorporates four policy template links.

The first policy is titled “Computer Policy”. The computer policy adjusts settings specific to the computer for which it is applied. The successive policies are the “All Users Policy,” “Restricted Users Policy,” and the “Software Restrictions Policy.” The “All Users Policy” is used to configure settings among all users, including administrators. The “Restricted Users Policy” is a link that hardens the Terminal Servers for standard end users. I configure the “Restricted Users Policy” by altering the security settings on the link itself, and denying access to administrators without affecting authenticated users. The “Software Restrictions Policy” is where I set software restrictions. I’ll discuss Software Restriction policies in depth later.

For each of the four Group Policy Template Links, Microsoft[6] suggests applying the option that only the computer or user portions of that template be processed. Each template stores both computer settings and user settings. For our four templates, we’d be processing eight distinct policy objects. By tuning appropriately, I can reduce policy processing in half and increase the overall perceived performance. The computer policy should be configured to only process computer settings, while the two user oriented templates and the software restrictions template should only process the users’ settings and ignore the computer settings.

Regarding the settings for the four templates, there isn’t one single formula of settings that works well for everyone, nor is this document intended to delve into each setting and analyze the respective impact. Instead, I will review the settings which have the greatest impact and discuss others in brief. However, there are many hundreds of settings in GPO’s and a careful review of each should be preformed to identify items that may serve a specific purpose for an organization that may not apply to everyone else. I will, however, go over a number of the significant settings and concepts to discuss their importance. In general though, begin with a restrictive policy and then enable features as the business demands require them, as Mitch Greyson recommends:

> I recommend starting with a very restrictive policy, then testing what you can do as you reenable certain features. Keep in mind that you should use Group Policy to eliminate confusion for users and to help protect the system from undesired and inadvertent activity.[7]

The loopback policy object

There is one policy setting required in order to make all of the aforementioned GPO’s work correctly, and hence deserves special attention. It is the **User Group Policy Loopback Processing Mode** property.

To understand why it is required, it is important to understand that as policies are applied, normally the users’ policies are applied at the Organization Unit of which they are members, not the Organization Unit where the Terminal Servers reside. However it is often desirable to make user policy settings that differ or perhaps even conflict with the settings from the normal users’ default policy while they are

using the Terminal Server. The loopback policy is ideal for this type of situation. It allows us to fine tune the user environment on a terminal server without affecting the end-user workstations.

One option to consider when enabling this setting is whether or not you **Merge** or **Replace** unconfigured settings with those configured in the user's Organizational Unit. I recommend **Replace**, as it gives more consistent results as stated here:

> Replace is generally preferred to Merge because the results can be hard to predict. In general, when in merge mode, settings defined in the Computer GPOs take preference over those defined in the User GPOs.[8]

However, you should review your results either way as there is no one correct answer; it really depends on your GPO structure.

Notable Computer Policy Object Settings

The Computer Policy contains an entire section under Computer Configuration dedicated to Terminal Services. This section can be found in

Computer Configuration\Administrative Templates\Windows Components\Terminal Services.

There are more than 40 different options in this section of the GPO. Many of the settings found here can also be found in the **Terminal Server Connection Configuration** tool. However, the GPO centralizes administration and generally creates more consistent results in a multi-server environment.

A few properties to note that are otherwise difficult to configure or administer are as follows:

Remove Windows Security Item From Start Menu – Self-explanatory. However, while it removes the **Windows Security** item from the start menu, it doesn't prevent a user from using the keystroke combination CTRL-ALT-END to display the security dialog box.
Sets Rules For Remote Control Of Terminal Services User Sessions – Useful to set the global Remote Control policies, instead of individually on each server.
Client/Server Data Redirection – All of the redirection options from the local machine can be configured in the policies.
Enryption and Security – The client connection encryption level, in addition to RPC Secuirty Policies.
Always Prompt Client For Password Upon Connection – This is useful to discourage users from embedding their password into the client, or for those who do, the password prompt is still presented regardless, preventing unauthorized access through a hijacked computer.
Sessions – Set time-out and connection limit values for sessions.

Notable Users Policy Object Settings

The Users Policy, both Restricted Users and All Users, contain many settings that can be tuned to harden a Terminal Server. The bulk of the Users Policy Settings that we are concerned with exist under the **User Configuration\Administrative Templates** section of the policy. Beneath it are sub-sections for the Start Menu & Taskbar, the Desktop, the Control Panel, and the System.

User Configuration\Administrative Templates\Windows Components\Windows Explorer\Turn On Classic Shell – This option disables Active Desktop, and more importantly, Web View in explorer. Web View allows explorer to become a pseudo browser which could allow users to access information in a manner that isn't intended.
User Configuration\Administrative Templates\Start Menu And Taskbar – this folder contains the majority of settings used to lock down a Terminal Server desktop. In an extreme case, almost all options would be enabled here.
User Configuration\Administrative Templates\Desktop – Contains several options that would add additional hardening to a locked-down desktop.
User Configuration\Administrative Templates\Control Panel\Prohibit Access To The Control Panel – There are few reasons why a user would need access to the control panel, and therefore it is recommended access be disabled.
User Configuration\Administrative Templates\System\Ctrl+Alt+Del Options\Remove Task Manager – Prevents users from launching task manager from the system Ctrl+Alt+Del menu. Task Manager is a system tool that should be reserved for administration use only.

For more information on Group Policy Objects specific to locking down a desktop, Microsoft has published a guide entitled Locking Down Windows Server 2003 Terminal Server Sessions.[9] The guide goes into great depth, discussing each of the many policy object properties and functions.

Group Policy Software Restrictions

In Windows XP and subsequently Windows 2003, Microsoft introduced Software Restriction Policies. Software Restrictions restrict access to software applications. The concept is not new, but the features have been greatly enhanced. In this section, I'll discuss Software Restrictions, their history, their features and why they are important. Finally, I'll go through how they can be implemented.

Restricting access to applications has been around for as long as NT shell policies. Originally, the function was quite simplistic. If the administrator wanted to prevent users from launching an application, perhaps a setup procedure, then he added setup.exe to the list of prohibited applications. While this worked ok, it didn't take users long to figure out that if they renamed setup.exe to install.exe, they were then able to load said application without a problem. So the administrator adds install.exe to the list of restricted applications, and before you know it, the user has renamed it again, this time to somecrazyapplicationname.exe, and so forth. Soon it was realized that the **prohibit shell applications** policy served little value and most administrators either simply didn't use it, or sought third party applications to suppress access to applications.

Software Restrictions were substantially enhanced with the release of Windows XP and Windows 2003 server. For the first time, the operating system gave administrators a choice to block all or allow all applications and create an associated exception list. Furthermore, Microsoft enhanced the exception list to verify more than just the file name and to look at such things as a file hash or

publisher based certificate authority. These features have restored a broken system to something useful.

The reason I use software restrictions in a Terminal Services environment is to control explicitly what is run on that server. There are two reasons this is important: security and change control.

First, with regard to security, I want to prevent any sort of malicious application from loading. Malicious applications such as keyboard logging tools could easily run in the background unbeknownst to the user and capture secure data being entered by the user. In addition to direct malicious behavior, I also want to prevent any of the privacy invading "spyware" applications from running and gathering information about what our users are doing.

The second reason to run with software restriction policies is to manage change control. As mentioned previously, change management is exceptionally important in a Terminal Services environment. If users are allowed to run an application that behaves poorly on their workstation, it probably will only effect themselves. However, on a terminal server, one user, or one bad application can affect tens or even hundreds of users. In a Terminal Services environment, every application should undergo extensive testing before being brought into production. Allowing even one user to launch an untested application can cause unpredictable results.

> The Problem with Unknown Code - Hostile code is not the only threat—many non-malicious software applications also cause problems. Any software not known and supported by an organization can conflict with other applications or change crucial configuration information. Software restriction policies were designed to help organizations control not just hostile code, but any unknown code—malicious or otherwise.[10]

An administrator's job is to provide a stable, consistent, and reliable environment for the end users. In order to achieve that goal, the administrator will need to understand how every application will affect the reliability, consumption of resources, and perceived impact on the end users. Allow a user to browse out on the internet, download programs, and install them on the Terminal Server can and will result in catastrophic results.

Software restrictions are implemented in Group Policies. When implementing a new Software Restriction Policy, an administrator needs to decide if it's an "allow all applications", except those listed in the policy rules, or "Deny all applications," except those listed in the policy rules. In other words, allow all, or deny all. For Terminal Servers, and Terminal Services implementations, I recommend choosing the more restrictive, Deny All policy, and then create an exception list. This establishes a very strict policy that enacts exacting control over what is allowed to run.

The choice of which applications to allow is unique to each environment. It depends entirely on the suite of applications you intend to roll-out. Analyzing a current running environment to gather a list of applications will be required.

The ability to troubleshoot software restrictions is also very important. To begin with, I've recommended the policy be stored in its own isolated policy link. This is done primarily for the sake of troubleshooting. By isolating the software restrictions in their own policy link, the administrator can easily disable the policy for troubleshooting without sacrificing the other policy settings.

Additionally, Microsoft offers other mechanisms to troubleshoot Software Restriction Policies. This is useful for both troubleshooting and auditing. The first place to look, whether you are troubleshooting, or auditing, is the System Event log. This event log records all failed attempts to load applications that were denied as a result of the software restrictions policies. The event log helps identify the failed application and then the administrator can adjust the rules accordingly. But equally as useful is searching the event log periodically to seek out users who are perhaps attempting to circumvent the acceptable use policy, too. A series of recurring failures from one user, where the application name is suspicious and perhaps changing each time may indicate a user who is trying to challenge or circumvent the security.

More information on troubleshooting, including how to create a custom log file and other useful utilities can be found in Microsoft's document, "Using Software Restriction Policies to Protect Against Unauthorized Software."[11] This complimentary resource is invaluable for creating and administering Software Restriction Policies.

Conclusion

Terminal Servers are powerful tools in today's connected world. They can provide access to applications and data while overcoming a variety of traditional workstation deployment challenges. However, without properly planning and securing your Terminal Server, you can leave yourself over exposed and easily compromised. This document has covered the key elements in designing a secure Terminal Services deployment. From proper planning, design and installation, through administration and configuration, I've covered the components that are necessary to review in order to establish a secure foundation for Terminal Servers.

References

[1] Microsoft Corporation. "Windows 2000 Services" July 2001
URL: http://www.microsoft.com/windows2000/docs/win2kservices.doc (December 2003)

[2] Microsoft Corporation. "The Ten Immutable Laws of Security"
URL: http://www.microsoft.com/technet/security/10imlaws.asp (December 2003)

[3] Microsoft Corporation. "Platform SDK: Terminal Services - Remote Desktop Protocol"
URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/termserv/termserv/remote_desktop_protocol.asp (December 2003)

[4] Microsoft Corporation. "HOW TO: Use IPSec Policy to Secure Terminal Services Communications in Windows 2000." Microsoft Knowledgebase Article 315055. Last Reviewed: November 18, 2003
URL: http://support.microsoft.com/?kbid=315055 (January 2004)

[5] Mitchum, Greyson. The Definitive Guide to Windows Server 2003 Terminal Services Realtimepublishers.com. Chapter 4, Page 86. URL:
http://www.tricerat.com/ebook/member/final.pdf

[6] Microsoft Corporation. "HOW TO: Optimize Group Policy for Logon Performance in Windows 2000." Microsoft Knowledgebase Article 315418. Last Reviewed: September 22, 2003
URL: http://support.microsoft.com/?kbid=315418 (Decemter 2003)

[7] Mitchum, Greyson. The Definitive Guide to Windows Server 2003 Terminal Services Realtimepublishers.com. Chapter 4, Page 85. URL:
http://www.tricerat.com/ebook/member/final.pdf

[8] "Locking Down a Terminal Services Session"
URL: http://www.sanx.info/tipShow.asp?articleRef=17 (January 2004)

[9] Microsoft Corporation. "Locking Down Windows Server 2003 Terminal Server Sessions" July 2003. URL: http://www.microsoft.com/windowsserver2003/techinfo/overview/lockdown.mspx (December 2003)

[10] Microsoft Corporation . "Using Software Restriction Policies to Protect Against Unauthorized Software" August 2002. URL:
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/winxppro/maintain/rstrplcy.asp (January 2004)

[11] Microsoft Corporation. "Using Software Restriction Policies to Protect Against Unauthorized Software" August 2002. URL:
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/winxppro/maintain/rstrplcy.asp (January 2004)