White Paper:
Service Account Management in Microsoft Windows NT/2000/XP/Server 2003 Servers

Rev 2 – June 1, 2006

Lieberman Software Corporation
http://www.liebsoft.com

Abstract

This white paper gives a detailed description of the technology behind services used in Microsoft NT, 2000, XP and 2003. Tips and tricks for service management are included as well a list of important features needed in any 3rd party solution.

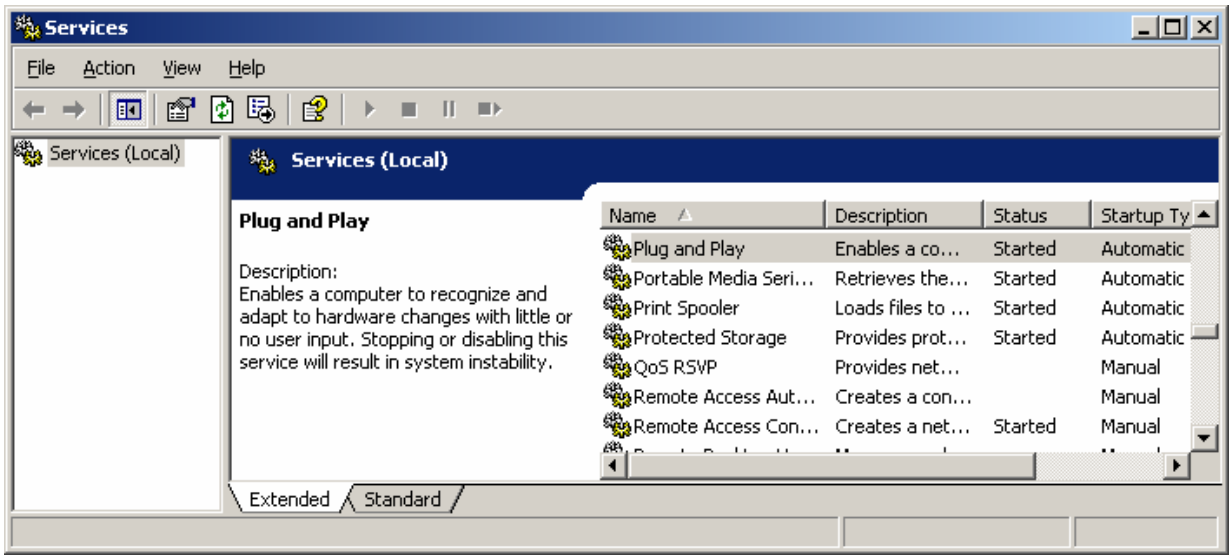
Contents

.

1. Introduction - What is a "Service"?

The simplest definition of a service would be a program that runs at boot-up time and continues running regardless of whether or not any user is logged on. In other operating systems, services are called "background programs or processes" and in UNIX they are called daemons (pronounced 'day-mons').

Services perform all sorts of useful tasks such as providing plug-and-play for your hardware. Plug-and-play means that you can install new devices in your computer such as a new printer, scanner, video card etc., and the operating system will "automatically" detect them and install all the proper drivers without any manual intervention. The "automatic" part of plug-and-play is done by a service that is always running and looking for new things that appear.

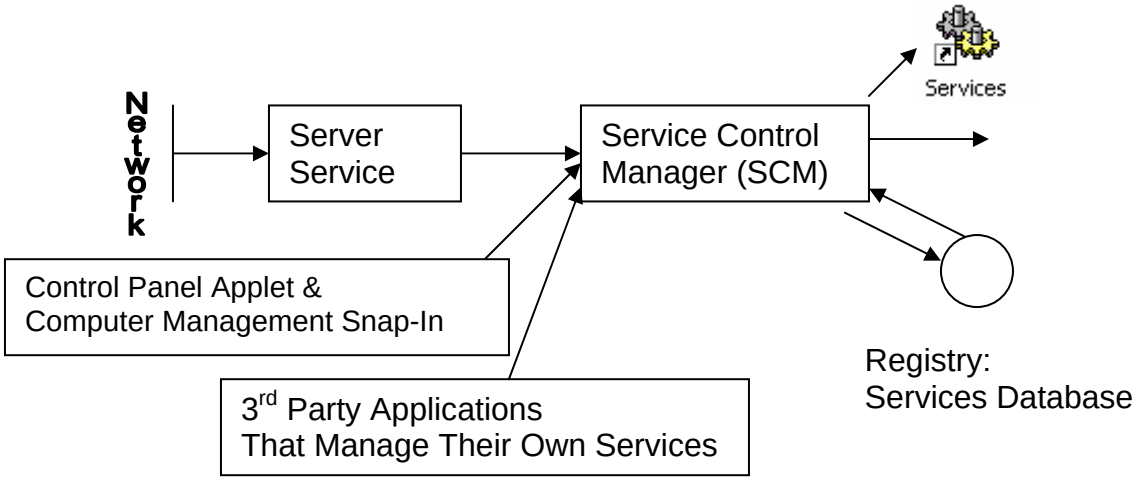


Microsoft added services to Windows NT and derivative operating systems such as Windows 2000, XP, and Server 2003. There are no services in Windows ME, 98 and 95. In case you were wondering, plug-and-play in these more primitive operating systems is accomplished by simulating services, but, these simulated services are not controllable or as sophisticated as real services.

2. How are services managed

A service is an executable program that is either loaded automatically at system startup or on demand. The management of services is done via a subsystem within Windows NT known as the "Service Control Manager" or SCM. The SCM is responsible for installing, removing, starting, stopping, monitoring, and service account management. The SCM is also hooked up to the network via the Server service so all SCM management can be done remotely as well as locally. The Server service depends on ports 137,138, and 139 on TCP/IP to be open and the remote registry access to be enabled on a machine for SCM to work remotely. If a machine is off-line, the Server service is stopped, the ports are blocked, or the remote registry access is blocked, the caller will receive an error codes. Typical error codes include: 53 (machine not available, off-line, not resolvable on network) or 1722 (RPC unavailable).

Clients in a secured environment with firewalls that block the specified ports will find that none of the Microsoft tools will be able to manage their systems across the firewall. The best solution to regaining control over their systems would be to implement a VPN (Virtual Private Network) to tunnel through the firewall, or modify the firewall parameters to allow selected hosts to use the specified ports. If neither of these solutions are feasible, the only alternate would be to run the management program(s) remotely via some sort of remote control software such as Terminal Server.

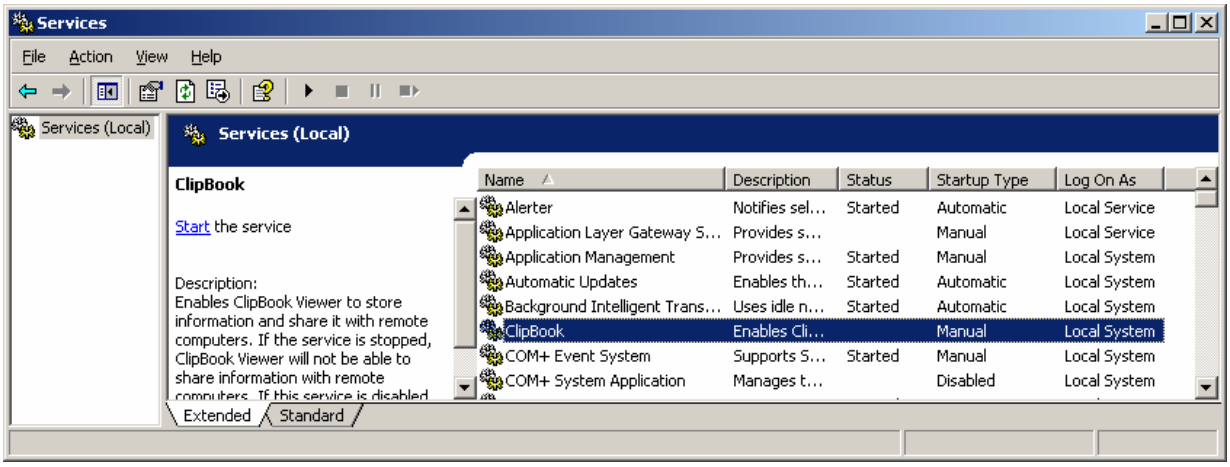


3. Services and the Registry

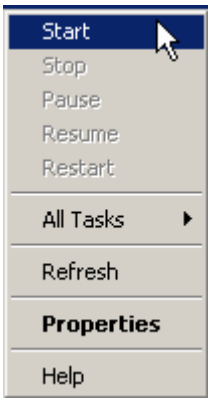
Each machine's SCM maintains the service information for all its services in the local registry of the machine. The SCM is supposed to be the exclusive manager of the services registry areas of a machine. Because many areas of the service database are obscured, encrypted, and hidden, it is a bad idea to directly edit the services area of a machine's registry. Attempting to do so may lead to the SCM reporting deleted services as installed, and report strange errors when other services are started. The most typical error code is #2 indicating that something is missing in the registry for a service under management. The best advice is to allow SCM to do all registry management without any extra help.

4. Service Management Applications

The primary interface to the SCM is the "Services" applet/snap-in that is located in the Control Panel of the operating system. Remote management of services is via the "Server Manager" application in Windows NT or by using the "Computer Management" MMC snap-in available in Windows 2000 and later operating systems.



Pictured is the Services snap-in from the Control Panel of a Windows 2000 system. Notice that we are managing the "(Local)" system services.



The list on the right shows all of the services running on the local machine. The list can be sorted by clicking on the heading tabs (Name, Description, Status, Startup Type, Log On As).

To modify one of the services, you must double-click on the service name or right-click and select "Properties" from the context menu.

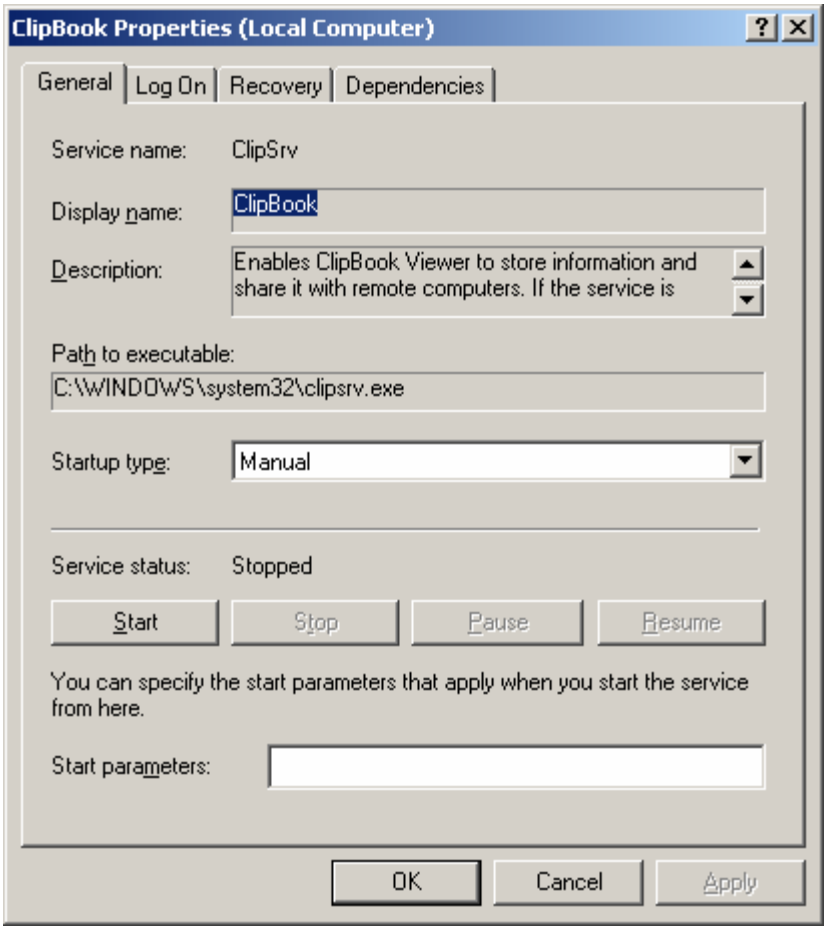
You can also change the state of a single selected task by first selecting a service of interest, then right-clicking and selecting common tasks such as stop/start/restart services.

Tip! With the built-in tools that Microsoft provides, you can only view/change a single service. In many third party applications you are able to modify more than a single service at the same time as well as report on the status of more than a single machine within a single view.

In the following sections we are going to examine all of the properties of a particular service (Clipbook).

General Tab

After double-clicking on the Clipbook entry we see the following tabbed dialog box:



With Windows 2000, Microsoft now shows a lot more information about services compared to what was shown in Windows NT. Here is what each field means:

Service name: This is the internal or real name of the service as far as the operating system is concerned. Registry keys on the local machine will reflect this short version of the service name. This name is constant no matter what language version of Windows is used—the Display name will change depending on the language version of Windows.

Display name: The display name is the version of the service name that you see on all of the Microsoft provided dialogs. This is the language specific friendly version of the service name. All services have both a Service and Display name assigned to them.

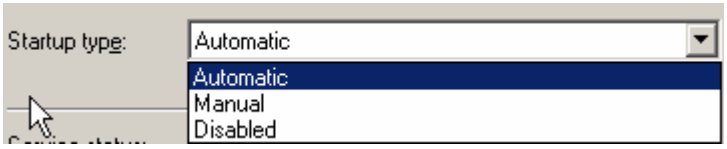
Description: Microsoft added this field to its own system services starting in Windows 2000. It is a description of what the service is designed to do. Third-party services created for compatibility with all versions of Windows will typically leave this field blank. The value in this field has no effect on the function of a service.

Path to executable: This is the location and file name of the service. Microsoft's own provided system services will typically be located in the %systemroot%\system32 subdirectory. Third party applications are urged by Microsoft to locate their services in the installation directory of an application.

Tip! Notice that Microsoft provides no way to edit any of the previous fields. The reasons are as follows: the Service and Display names are hard coded within the executable of the service executable itself. So, only the program developer is capable of changing these strings by compiling a new executable (EXE).

There is no method provided by Microsoft to remove or install services. Similarly there is no way to change the path of a service (field is shown as read-only). The justification of this decision is that the application should provide its own method of installation and removal. Many third party service management applications provide for the mass installation and removal of services as well as ways of modifying the name and path to the service executable.

Startup type: A service is normally started at the same time as the operating system. This is known as an "Automatic" or "Autostart" service. There are also services that require the administrator or other services to manually start them only when they are needed; these are known as "Manual" or "Demand" services. Services that are installed, but should not run are set as "Disabled." Disabled services cannot be started until their type is changed to either "Automatic" or "Manual."



Service status: Shows you the state of the currently selected service. The steady state of a service can be: stopped, started, or paused. Very few services implement the paused state. When transitioning from one state to another (i.e stopped to started), the service is in an intermediate state that should not last forever, however, sometimes it never leaves the transition state. Rebooting sometimes helps this situation, but sometimes the old solution is the removal of the service. Sometimes the SCM can accomplish this with a third party tool to remove a service, and in other cases, the registry area containing the effected service may need manual editing.

Start parameters: Just as you can pass parameters to a program on the command line, you can also pass parameters to a service. Generally, services rarely use any command line parameters as part of their start up. The only exception is for services that need the command line arguments to put the service into special diagnostic modes or to force them into unusual states for debugging purposes..

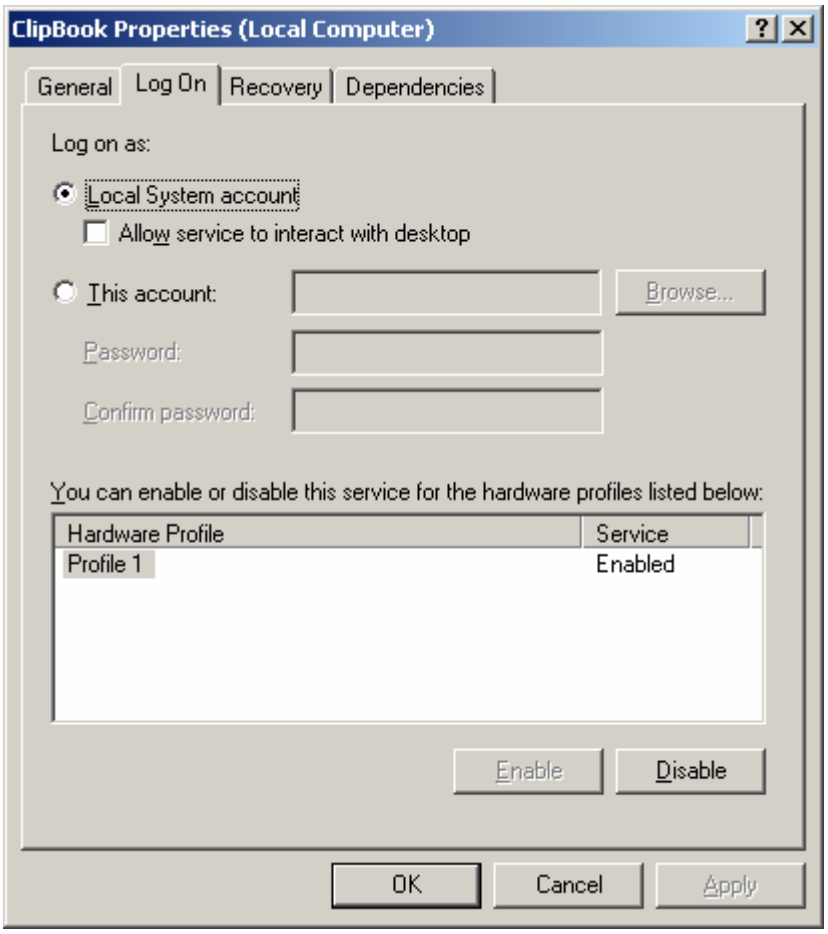
5. Control Buttons: Start, Stop, Pause, Resume

These buttons are grayed out or enabled depending on the current Service status and how the service was written. When a service is written, the developer has the ability to signal to the SCM whether or not the Pause and Resume buttons are effective for the service. If the service does not mark these actions as effective, they will be grayed out on the property page (you will almost always find the Pause and Resume buttons grayed out).

Stopping and starting services is a tricky proposition since some services depend on other services. You cannot stop a service that has other services dependent on it. You must first stop all of the services that reference the service you are stopping. In a similar vein, you cannot start a service if certain base or dependent services are not running. The built-in management of dependent services is very basic and does not correctly handle complex service dependency scenarios. Even most third party utilities fail to handle service dependency complexities. The most sophisticated third party service tools can accomplish the start/stop of even the most complex scenarios with ease. We'll talk more about these issues when we cover the Dependency Tab later in this article.

6. Log On Tab

The Log On tab controls the account that the service runs as. Services must have a log on account to operate. This requirement is necessary since all programs running in NT or later must have an account context to control the scope of their access. Since there is nobody logged on to the machine when it boots initially, the service account allows the service to start well before any user has logged onto a machine. The account requirement also allows a program to persist after someone has logged off of a machine. Services keep running under the context of the log on account for each service until each service is restarted or the machine is rebooted.

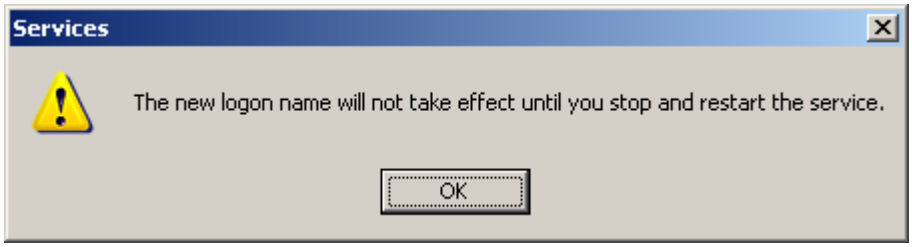


7. How Service Accounts are Changed

Many administrators are confused about the process of changing service accounts. Should the services be stopped first, should the systems be rebooted after the change, what about dependent services? Here is the whole story.

Service account information is only used when the service is started. Once a service is started, you can change the account and password used by the service, but it will still continue using the old account information until the service is first stopped and then started (this is sometimes called a restart). Obviously, if you change the account and then reboot the machine, the new account will be used upon system startup if it is valid and the service is set to start, or is started on demand.

Changing the account in the Log On tab while the service is running will produce the following pop-up:

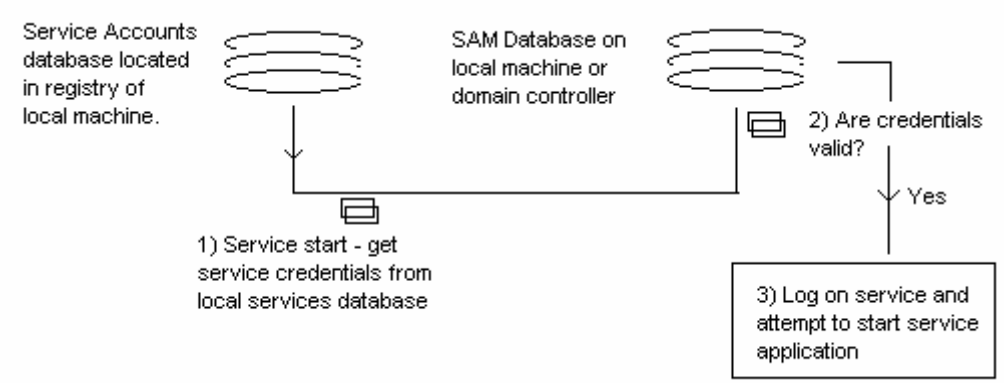


Tip! Be careful changing accounts with the Microsoft snap-in when a service is stopped. The account will be changed, but the service will not automatically start. Even if the service is running, changing the account only produces a warning that you will need to stop and start the service for the new account to take effect. Again, third party tools generally will take care of the stop/start issues for you.

8. Log On Account

Windows NT and above have security models that requires all programs to have account credentials to run. When you run a program from your desktop, every operation performed has your logon credentials attached to it. The operating system checks your credentials against the security restrictions of objects you are attempting to access (i.e. files, directories, printers, other machines) and decides whether the operation is allowed.

Because there is no one logged on to the system when a machine boots up, services must store an account and password to present to the operating system when they start. If there is no account information for a service, or the account/password information is incorrect, the service will not be able to start. The account and password information for all services on a machine are stored in an encrypted area of each machine's registry.



Because a machine maintains a unique database of accounts to be used by the services of that machine, any account other than Local System must be validated by a completely separate SAM database on the local machine or the Active Directory on a Windows 2000 domain. If the service accounts database information for accounts is incorrect, then those services that use the wrong account information will not start.

What this means is simple. When you change a domain administrator account that is used by services, you must change all of the local machine service account databases to use the new domain account information. This is not done automatically; you must do this manually, one machine at a time. You can try to write a script to do this using some of the resource kit utilities, but there are a couple of other issues you must take care of even if you use the resource script applets.

9. Three types of accounts used by services:

Local System – this is the account used by the majority of system services. Windows XP introduces a variant of the Local System account known as Network Service that allows limited network access for the Local System account. The Local System account (internally known as the account LocalSystem) gives a service rights and abilities greater than those of a local administrator (similar to root permissions in Unix/Linux). This account has no password associated with it (it is blank) and can only be used by services. There is no ability for a user to interactively or via the network to logon using this account.

Local Administrator – this is an administrator account located on the local machine. This type of account may be used when only a workgroup (non-domain) network architecture is in place and peer-to-peer authentication is the preferred method of authentication.

Domain Administrator – commonly used by enterprise-wide applications that needs full local and domain wide administrative access. This is most common account used by application services.

How a service should be configured (type of account) is determined by the designer of the service. Setting the service account is generally done at the time an application is installed. After installation, product vendors usually provide little in the way of tools to modify the service accounts that have been distributed throughout an organization. This is an area where a third party tool makes account changes across many similarly configured systems a lot easier.

Note: some applications (i.e. Veritas BackupExec) composed of multiple services may use an assortment of service accounts where some of the services use Local System and others use normal accounts (usually domain administrator). It is extremely important that the administrator of the service accounts does not change an account using Local System to a normal account and vice versa as this can and most likely will cause most applications to fail.

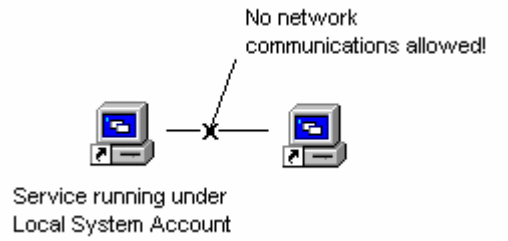
10. Interact with Desktop Option for Local System Accounts

Services rarely pop up on a user's desktop if they need attention. There is a good reason for this: sometimes a service needs attention and there is nobody logged onto the machine. In that case, there is no desktop to interact with. Desktop interaction with a service is also a dangerous option since any operation taken as a result of a user's interaction with a service with this option checked will occur at the operating system level of privilege. The ability to interact with the desktop is totally determined by the developer of a service. Do not turn this option on or off arbitrarily as it may cause major problems with your system. Services that need desktop interaction may freeze up if this checkbox is not checked. Services that don't need this enablement will not be affected by any change in the status of this checkbox.

Tip! Only the Local System account is allowed to potentially interact with the desktop. Other (real) accounts are not allowed to interact with the desktop. Reporting on all services with the setting of "interact with the desktop" is provided in some 3rd party applications.

11. Limited Scope of the Local System Account

Although the Local System account is very powerful, it has a limited scope of authority, namely it can act as it wishes only within the local machine. If a service attempts to contact another machine while running under the credentials of Local System all other machines will refuse to communicate with it. Services that run under the "Local System" account must impersonate a real local or domain account to access remote systems.



12. Changing Service Accounts – Required Rights

Each account used by a service performs a Service Log On that is normally forbidden to all accounts including all administrator accounts (the Local System account is always allowed to do Service Log On). This requires that the "Log on as a service (SeServiceLogon)" right be added to each unique administrator account that is used by any service on a machine. Because many services perform impersonation, a second right known as "act as part of the operating system

(SeTCBPrivilege)" also must be granted to all service accounts. If an account is used by Microsoft Exchange, then two additional rights must be granted: "Backup files and directories (SeBackupPrivilege)" and "Restore files and directories (SeRestorePrivilege)" will also need to be added. Some services may need additional rights.

Tip! When evaluating 3rd party service management tools, verify that they have the general purpose ability to also take care of the rights and group membership management needed for changing service accounts.

13. Security Policies for Service Accounts

Good security policy dictates that domain administrator accounts should be changed regularly (no less than every 90 days) or when the account is potentially compromised (such as when an administrator with knowledge of the account leaves the organization). The challenge is finding all occurrences of administrator accounts used by the services and changing them in the shortest period of time.

14.Changing a Domain Administrator Account

Changing a Domain Administrator account is not as simple as changing it in the Log on tab. If an existing domain administrator account is already in use and it is desired to just change the password of the account (for security purposes), the administrator must first change it at the domain controller, and then change the services that reference the account, except all passwords must be updated.

This seeming simple procedure of changing the domain controller and then changing the services that reference the just modified account is fraught with many problems. First, changing the domain controller information does not mean this change is instantaneously replicated to all other domain controllers. If you change a service log on account, start the service, and it tries to authenticate to a domain controller that has not yet received the updated information, the service will fail to start due to bad credentials.

Some services that use a domain administrator account are constantly logging on and off with different credentials (i.e. Microsoft Exchange). If you have changed the domain controller, but have not gotten around to the service that is logging on and off constantly, the service will begin to fail and shut down.

The third scenario involves so-called good security policies. Many companies implement account lockout. Some go so far as to implement permanent lockout of administrator accounts if they exceed the number of bad password attempts. If you are in the middle of changing administrator password accounts in a domain for services, some of the services will inadvertently log on with bad credentials causing an almost instantaneous lock out of the account and subsequent cascaded failure of services throughout the organization since even the correctly configured services will be unable to authenticate to a locked out account.

15. How to Properly Change Domain Administrator Accounts Used by Services

1) At the domain level, turn off account lock-out. This can be done in NT 4.0 via User Manager for Domains. In Windows 2000, you will want to make the change in the domain controller policy and force this change immediately.

2) Change the domain-wide account used by the service in User Manager for Domains (NT 4.0), or in the Active Directory of Windows 2000/XP.

3) Force replication of the new account to all BDCs (NT 4.0) via Server Manager, or via the MMC plug-in to force domain replication in 2000.

4) Use the fastest mass management tool possible to change all of the affected service log on accounts.

5) Once all of the services are confirmed up and running (you may want to do a few refreshes of the changed services to confirm their new account and operational status), go ahead and turn account lockout back on again.

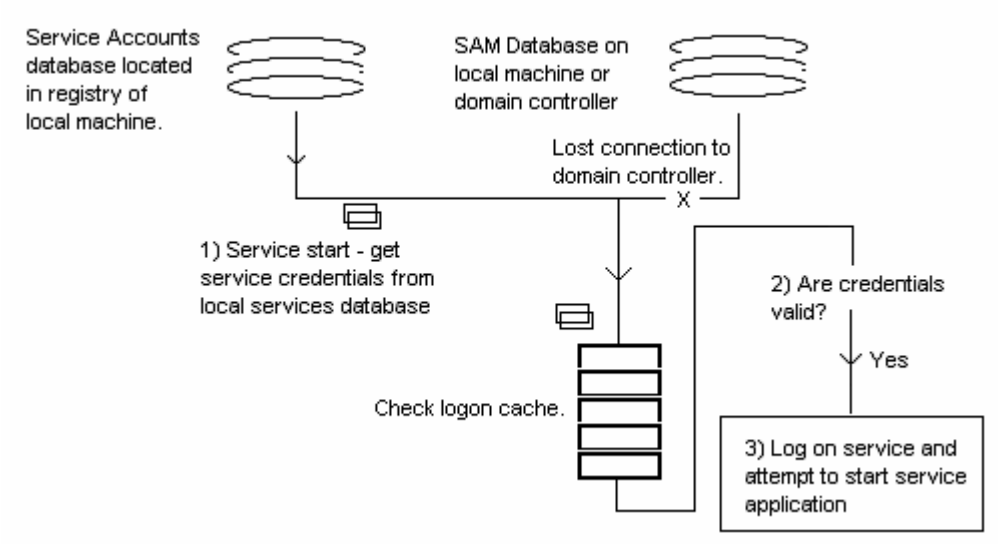
Tip: you may want to leave account lockout off for a little while and examine the Event logs of your domain controllers to confirm that there are no stray services still running with the wrong account settings.

16. Domain controller problems and laptops

When a service starts that uses a domain administrator account, it must contact a domain controller and verify that the account and password specified are correct before the service can start. Imagine what would happen if the domain controller is unavailable or if you are running from a laptop that is disconnected from the network? Right, the service fails to start because there is no way to authenticate the service.

The problem can be even worse for production servers that temporarily lose their connection to a domain controller and cause their working services to fail.

The trick to solving this problem is to logon interactively at each and every machine with each domain administrator account used by services on each machine. By performing an interactive logon, you can put the domain administrator credentials in the "logon cache" of the machine. When the domain controller cannot be found, the service will automatically check the logon cache to verify the credentials, and if they are correct, the service will start correctly and operate as though the domain controller is actually there.



The problem with this strategy is that Microsoft provides no published method of writing a program to update the logon cache of each machine other than walking up to each machine.

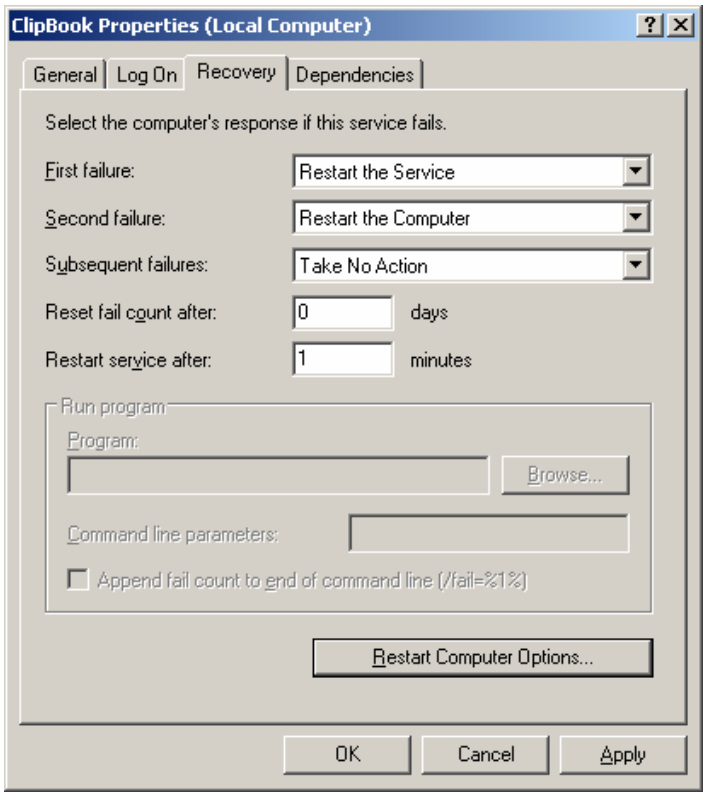
Tip! If you are evaluating 3rd party tools for service management, check if they are capable of managing the logon cache of the systems.

17. Services Accounts that refuse to change – forced reboots

Due to incorrect coding of some application services, after changing the service account, the service will refuse to start correctly. The only solution in these cases is to change the service account without restarting the services (yes, you can change a service account without restarting the service, but the new account is not used until the service starts again). After the service settings have been changed, you can then reboot the systems using the restart option. This is a necessary procedure for some older versions of Veritas BackupExec. For proper management of these cases, your service management tool should allow service settings changes without restarts as well as a method for specifying a reboot at a specified time

18. Recovery Tab

Microsoft added the ability in Windows 2000 to detect and recover from terminated services via the "Recover" tab in the services settings:



As long as a service terminates, and the SCM can detect it, this feature allows the administrator to configure the action to take. The actions include:

- Take No Action
- Restart the Service
- Run a Program
- Restart the Computer

The dialog also tracks the results of the last recovery action and can take different actions for each attempt to recover. In most recovery scenarios, the administrator will want to escalate the severity of action all the way to rebooting the system.

Tip! Although Microsoft has vastly improved the recovery starting in Windows 2000, this implementation does not detect services that are running but no longer responding correctly or in a timely manner. Also, if a service is leaking memory there is no way to cause an action to be performed when the parameters of operation go out of whack. There are many 3rd party tools that measure the responsiveness of a wide range of services and take actions similar to those provided by Windows when a service terminates unexpectedly. Some 3rd party service management tools are capable of restarting known leaking services on a regular basis (this frees all of the memory from the service and restores it to stability for a while) as well as periodically rebooting systems at points of time where system activity is minimal.

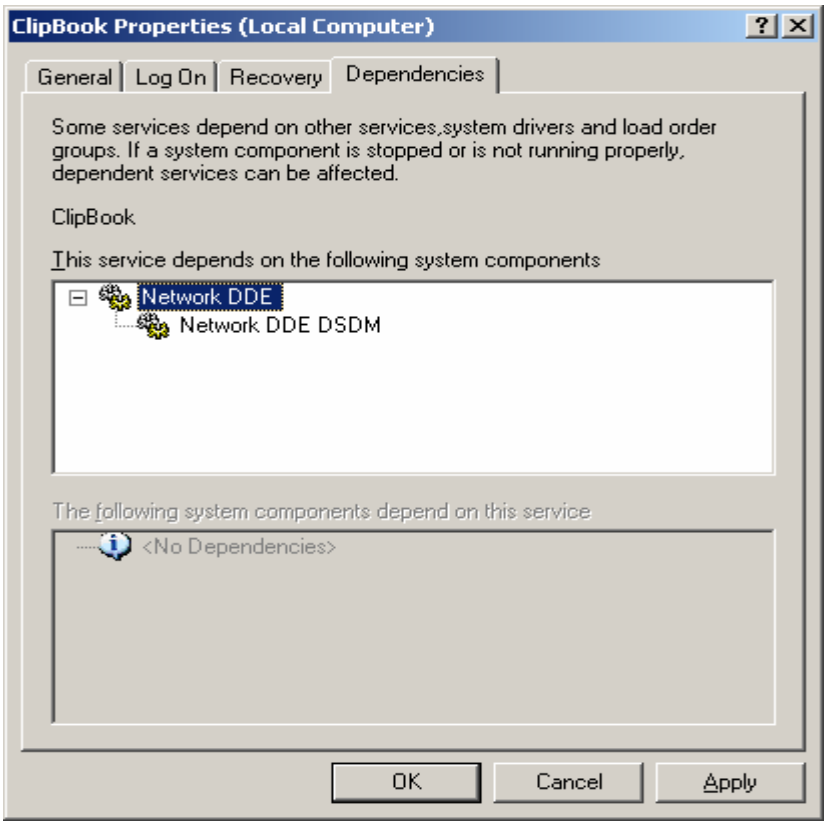
The use of a 3rd party tool for the periodic measurement or at least the periodic restarting of known problematic services and systems is an essential part of maintaining high system availability. We know we all have the problem, these tools make it go away.

19. Dependency Tab

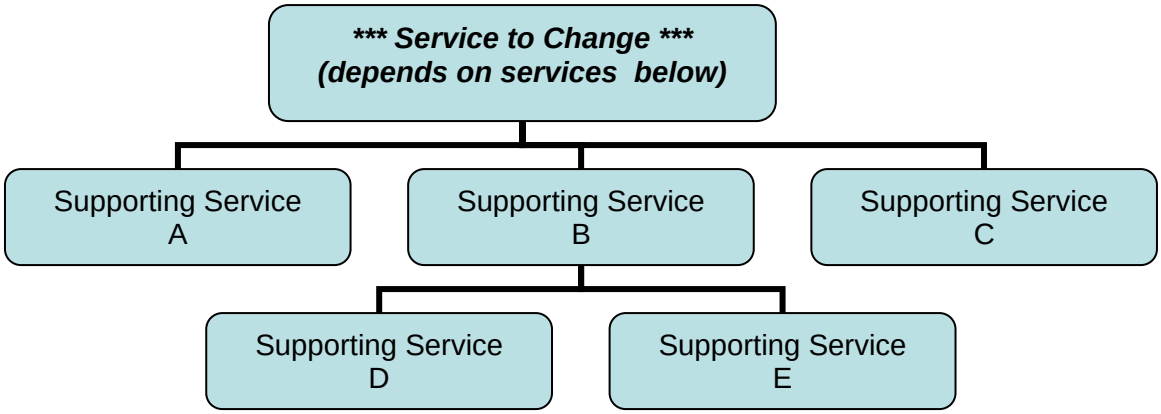
Dependency issues can be the hardest thing about changing services.

There are two types of dependencies: services that a service depends on before it can start, and service(s) that depend on the service. The screen shot below first shows the services that the ClipBook service depends on (Network DDE and Network DDE DSDM). The second window in the screen shows that no service depends on the Clipboard service.

In this case, we stop the Clipbook service whenever we want since nothing depends on the service. We can start the Clipbook service as long as the Network DDE and Network DDE DSDM services are running or can be started on demand.



To understand how dependencies affect service start/stops, imagine a service is a floor you are standing in within a tall building. You depend on the floors beneath you to be there before you can walk on the floor (these would be the services {floors} you "depend on" or your foundation). You can change settings (start service) of your services as desired as long as the supporting services are running or can be started. Before you can remove a floor, all of the floors above it must first be removed. This last case is equivalent to stopping a service while other services depend on the current service.



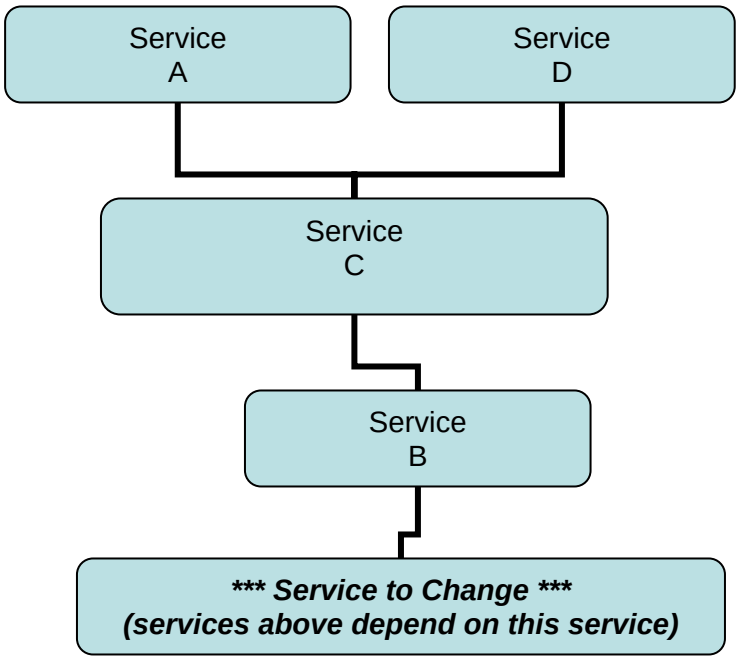
Service to change 'depends on' services below it running

20. Dependencies – Changing a service where other services depend on it

When managing complex service based applications such as Veritas BackupExec and Microsoft Exchange, you may have been frustrated by your inability to stop the services as well as being unable to change the account used by these applications.

The reason for your frustration was due to the complex interlocked relationship between all of the services in these applications. Simple service management tools such as those provided by Microsoft in the operating system are not capable of disentangling the complex service relationships and fail. Only the manufacturer supplied tools generally do the job correctly (and then sometimes even these tools do not work). In the worst case, some applications require the complete reinstallation of an application to change the service account.

With the exception of Microsoft's SNA server, we have not encountered any services that absolutely require the reinstallation of an application to change a service account. In our opinion, this implementation in SNA Server appears to be a fundamental security design error in the application that needs to be corrected. Consider if you have a distributed organization with hundreds of copies of SNA Server, your administrator leaves, and you now need to reinstall SNA Server on all systems with a new account. Imagine!



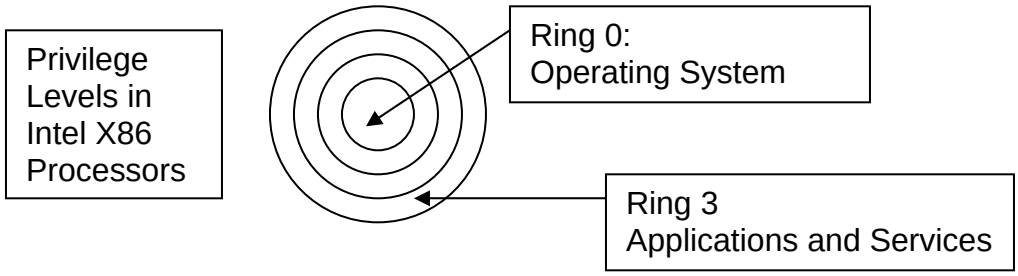
Disentangling the interlocking relationships of services can be done by examining the hierarchical relationships between all services, dynamically building a linear dependency list, stopping all services above the current service, stopping the current service, changing the current service, starting the current service, and restarting all of the services in the just built linear dependency list. Using this technique, even the most complex service can be managed with ease. In some cases additional dependency ordering may be needed, so any tool for service management should provide both dynamic and manual specification of dependent service stop/starts.

Tip! Microsoft's own management tool for services makes no attempt to resolve dependency issues. Make sure that any 3rd party management solution for services is capable of handling these complex dependency relationships. If you have no other choice, you can change your service accounts without stopping and starting services and just restart the machine (and hope for the best).

21. Other Interesting Topics Regarding Services

Privilege Levels

Windows NT and above employ hardware protection within the processor (CPU) that allows code to be classified as to privilege level. The operating system and device drivers operate at ring level 0, also known as kernel-level or system-level privilege. At this level, there are no restrictions on where a program can go or what it can do. The only problem at this level is that if a program or device driver makes a mistake, the operating system normally halts with a trap screen (the famous blue screen of death (BSOD)).



Applications and services operate at ring level 3, also known as user-level or application-level privilege. If an application or service fails at this level, a trap screen will appear (also known as a general protection fault (GPF)), that can be dismissed without the operating system caring.

The decision to have services run at the same privilege level as regular applications is based on the idea that if the service traps (blows up), the operating system should continue to operate without problems.

However, there are few problems with this architecture. First, if a system service such as the NTLM Security Support Provider (LSASS.EXE) fails, no one can logon to the system. There is no way to restart a crashed system service such as the authenticator, so a reboot is necessary to restart it.

A second problem is with application services that fail. Namely, there is no inherent way to gain access to the troubled service to see what is wrong via operating system, since it operates on a console screen that is inaccessible to everyone including the administrator. Yes, services have a keyboard, screen and mouse, but unfortunately, neither you nor I can access these peripherals since they are virtual or phantom devices to trick the services into believing that they are running at a normal console.

Replacing Files

Services consist of an executable as well as other support files. A handy feature for any service management tool is the ability to change the files used by a service when newer versions become available. However, before any file replacements can be done, the service must be stopped so that you do not receive any file locked errors.

An ideal way to perform this file update would be to stop a service, copy the new files in, and then restart the service in one operation. This will allow for the smooth update of files.

Impersonation

Many organizations have to support a wide variety of different domains and workgroups. The problem with Microsoft's built-in administration tools is that they only use the current logon account to access remote systems. If you must administer machines on another domain (where there is no trust relationship with your current domain) or in one or more different workgroup, you are forced to logoff, logon with the proper credentials, and then work on a different group of systems.

The ideal case would allow you to preload alternate credentials for all potential domains and workgroups you will be administering. When connecting to systems, the built-in credentials are tried first, and then all of the alternates are tried until one is found that works, from that point on, the best alternate credentials are used for each system.

Reporting

Because some systems may become removed from the scope of administration by hostile users, some means should be provided to verify the ability of an administrator to access the systems under their control. This list of out-of-control systems should be exportable to Excel or a database system such as Microsoft Access.

Auto Retry and Deferred Scheduling

In every change scenario administrators will encounter a fraction of systems that are temporarily off-line. Because administrative changes need to be carried out as quickly as possible, a secure change program for services must place all missed systems on a deferred processing list for automatic retry at periodic intervals.

The need for auto retry is especially important when the systems being managed are laptops that may appear spontaneously on the network. Without an automated way of continuously checking and changing missed machines, laptop users would most likely miss any administrative changes.

Although many administrators would be happy to make their changes as soon as possible, the best time of day for making changes may be the middle of the night. An ideal tool should allow for the scheduling of service changes while the administrator is home comfortably asleep. These changes can include not only changing service accounts, but also restarting services that are known to leak memory as well as periodically shutting down and rebooting servers to stabilize them.

Logging

The biggest problem with most GUI-based applications is that they generally do not log the changes made by the program. If a system is missed, or an error is returned by a system, there is rarely any trace of anything that may have gone wrong. This means that most tools leave the administrator blind when it comes to the changes that were made to their network.

An ideal tool should log all changes (both success and failure) as well as be configurable to the level of detail returned on transactions. In no case should transaction logs be lost or overwritten as they contain valuable information necessary for information security auditing.

22. Summary

Microsoft has made significant improvements in the management of services with the release of the Services snap-in within Windows 2000 and later versions of this operating system. Even with these improvements, administrators of more than a few systems still need a 3rd party tool to effectively manage services.

There are a number of independent software vendors who have produced tools to fill the gap in the tool set provided by Microsoft. The solutions span the gap from simple utilities that are little more than reformatted versions of Microsoft's own snap-in, to extremely powerful solutions that provide every bell-and-whistle that could possibly be needed for an administrator, no matter how large their enterprise or complex their environment.

In this paper I have hopefully given you not only a tutorial of the technology of services and their management, but a strong understanding of the features you will need in any 3rd party solution for service management.

