Global Information Assurance Certification Paper

GCNT Practical Assignment
Version 2.1

The IIS Metabase: What it is, how to use it and how to secure it.

Jason Trudel

TABLE OF CONTENTS

Introduction

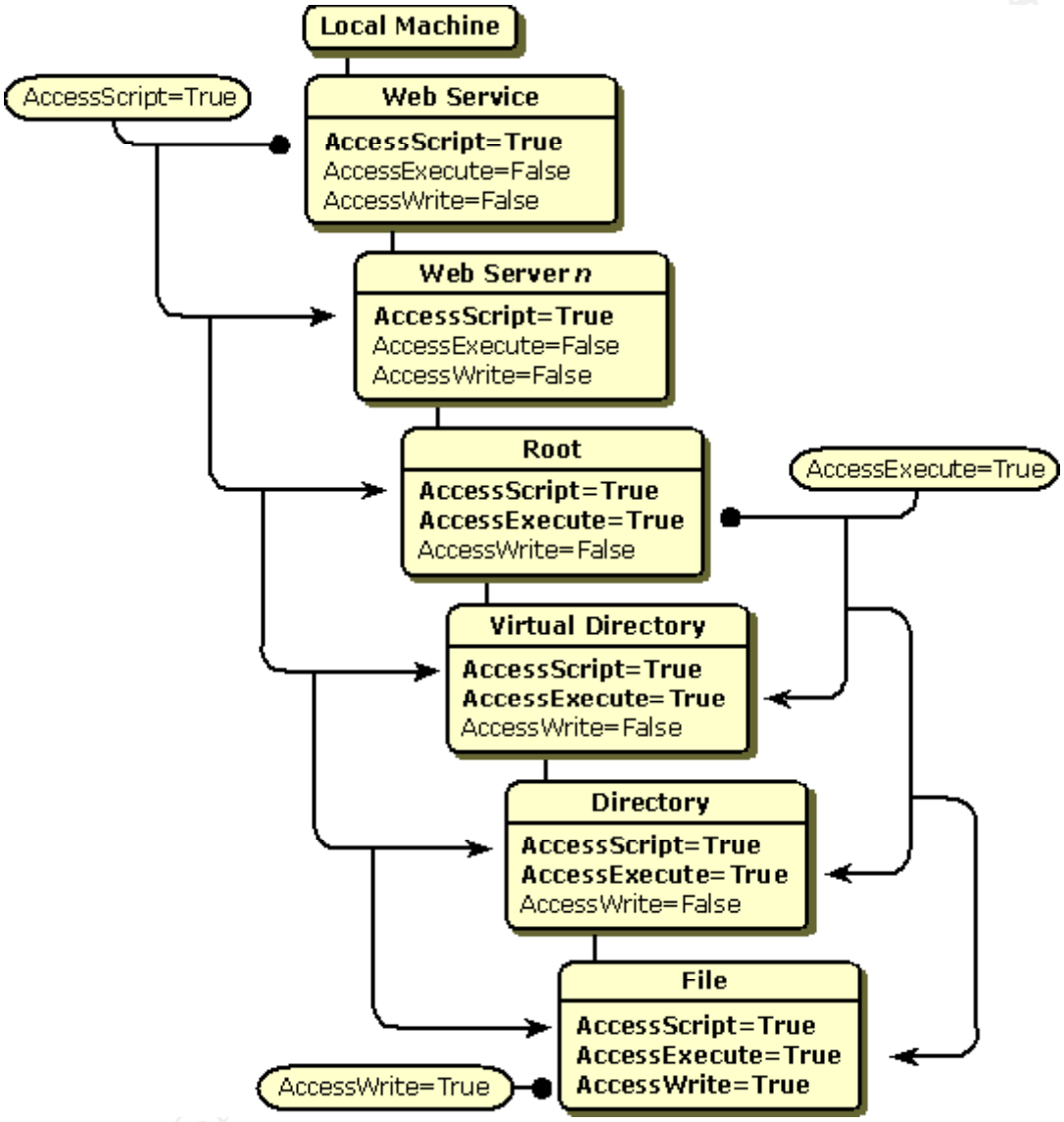
With the ever increasing need for web presence a tremendous amount of new web servers join the Internet everyday. These may be installed by security conscious administrators or just the average person who wants to show off the family photo album. The question is how secure are they? History shows that web servers are one of the most popular targets for attackers, with their high availability on the web, new exploits popping up every couple weeks or just the reality that a lot of them are configured incorrectly. A default installation of Internet Information Server (IIS) is very insecure. This paper will not tell you how to take your default install of IIS and protect it against attackers. That has been done before, and It will be assumed that a hardened install of Windows 2000 Server with IIS 5.0 all patched up and ready to go is done before we go on to the next layer of protection, securing the IIS Metabase.

First we will go over what exactly the Metabase is, how it is used by IIS and how it can be used by the web administrator or attackers if not properly protected. I will discuss tools that are specifically designed to manipulate the Metabase and how applications or scripts can be used to perform tasks that you can't easily perform through other standard interfaces. Finally, we will go over steps to tighten the security of the Metabase and in short, increase the overall security of your web or FTP server.

The MetaBase

IIS has always needed to be a highly configurable application. To be configurable some storage space has to be put aside for configuration settings. Pre IIS 4.0 releases used the windows registry for this purpose along with many other applications and Windows itself. This was not good enough, IIS needed to have quicker access and more configuration than the registry could provide. With the release of IIS 4.0, so came the IIS Metabase, a more sophisticated more flexible "registry" used solely by IIS. This setup is smaller than the windows registry has faster retrieval capabilities. Since IIS needed the ability to configure many options at different levels, a hierarchical structure was used. This structure mirrors the make up of the IIS installation, going down the hierarchy with nodes, keys and subkeys. Like the windows registry it can hold binary data as well as textual data. Unlike the registry, the Metabase uses inheritance. Inheritance will carry a change on a key down the hierarchy onto all of its subkeys, a very useful function. A key is a node in the hierarchy that represents an object, such as a virtual directory. Each key holds properties that

affect the configuration of its associated element. Subkeys can also exist in a key, which could represent a subdirectory in a web or FTP site.



This diagram depicts the flow of inheritance as it flows down the hierarchy. Notice at the Web Service level AccessScript is set to TRUE. This setting will flow into the hierarchy until it is changed to affect everything below it. At the lower root level, the AccesssExecute is set to TRUE and carried on down from there. Finally, the AccessWrite is set to TRUE at the File level.

Metabase ID's

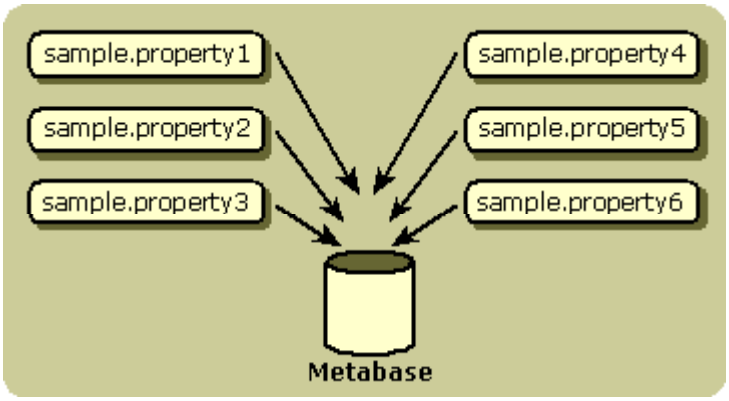
The MetaBase's properties are stored by numeric ID's instead of textual names. Each property will get a unique ID that is sorted into a range represented by a specific component, which makes identification more efficient. The way the range is divided is shown in the following table:

MetaBase ID Range	Owning IIS Component
1000 – 1999	General Server
2000 – 2999	HTTP server
3000 – 3999	Virtual directories
4000 – 4999	Logging
5000 – 5499	FTP server
5500 – 5999	SSL
6000 – 6999	File and Directory Properties
7000 – 7499	ASP
7500 – 7999	WAM
0x8000 – 0x8FFF	Microsoft FrontPage
0x9000 – 0x9FFF	SMTP server
0xA000 – 0xAFFF	POP3 server
0xB000 – 0xBFFF	NNTP server
0xC000 – 0xCFFF	IMAP Mail server
0xD000 – 0xDFFF	General Microsoft Commercial Internet Systems IDs

This is just a guide, but if you are adding your own values to the MetaBase values in these ranges should not be used because each property needs a unique ID. Some property ID's are not made public to ensure security of private information for IIS or other applications

Eight types of properties that the Metabase stores are:

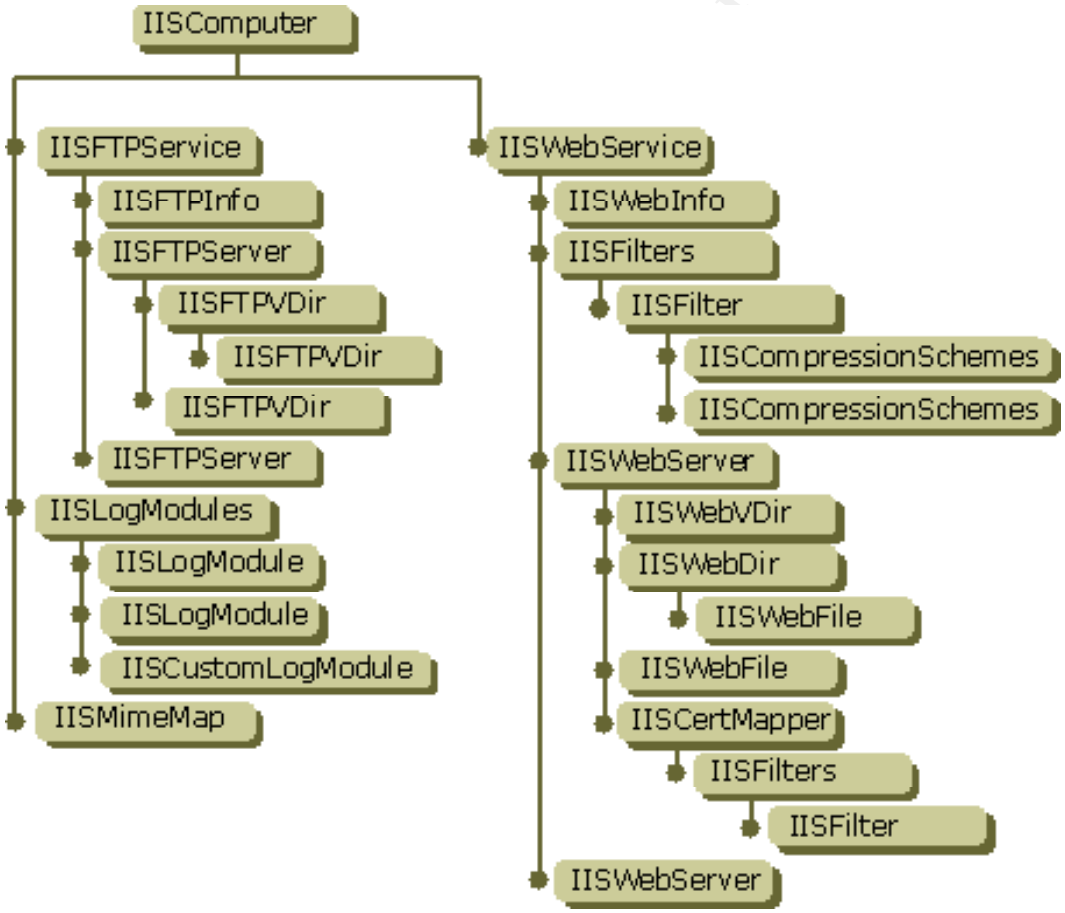
- computer and web site properties
- Logging properties
- FTP Specific properties
- HTTP Specific properties
- Virtual directory and directory properties
- File properties
- Filter properties
- Secure socket layer properties



Physically on your disk, the Metabase resides on your hard drive at %SystemRoot%\system32\inetserv. The binary file is named METABASE.BIN and is around 200 KB with the default install of IIS. "It is loaded from disk when IIS starts, stored to disk when IIS shuts down, and saved periodically while IIS is running" – *Metabase Security and Reliability*, IIS 5.0 Documentation…saved periodically?

MetaBase Namespace

The general structure of the IIS is called the *NAMESPACE*. The namespace specifies the location of the Metabase properties it is organized as follows:



A logical interpretation of this would be as follows:

LM/Service/Website/Root/virtual directory/dir/file

Where:

LM = Local Machine – this would be the machine that you Web/FTP server resides on

Service = Internet Service (W3SVC – Web Server or MSFTPSVC – FTP Server)

Website = Web site – this will show up as a number, I will explain later on how to relate this number to what site it actually is, if your hosting more than one site

Root = virtual directory root – the root folder of that web site

Virtual Directory = virtual directory - each "sub-web" or Virtual Directory

dir = directory – directory residing in each web or Virtual Directory

file = file – your web pages or files in each directory

If we wanted to reference our default web page on the first web site on the server, we could do it the following ways.

The **namespace** of this Metabase path
IIS://LM/W3SVC/1/Root/file

would be associated with the **path**
D:\Inetpub\wwwroot

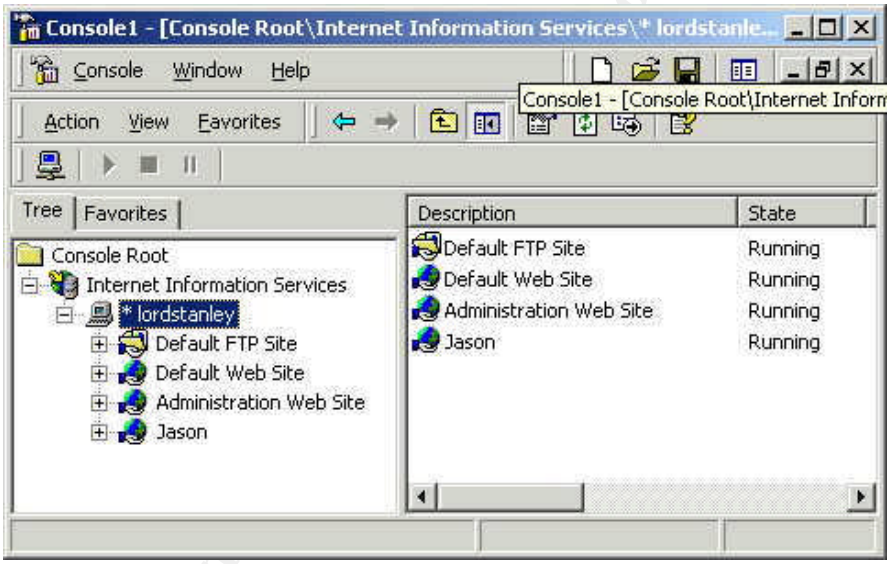
then the **URL** would be:
http:\\www.yourwebserver.com\default.htm

the resulting **physical path** would be:
D:\Inetpub\wwwroot\default.htm

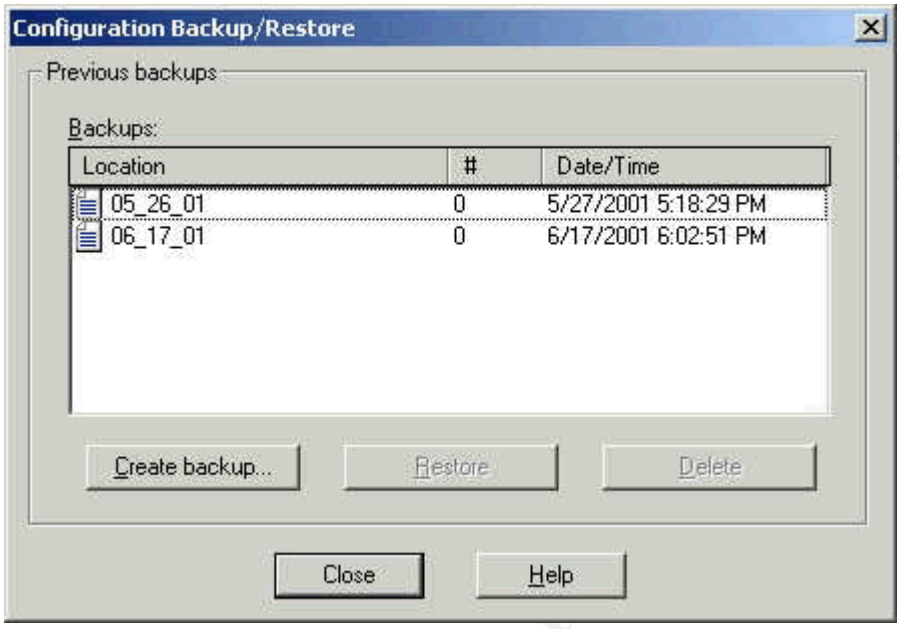
Backing Up/Restoring the MetaBase

Editing the Metabase can be intimidating, all of the warnings that go with any changes or modifications in the Windows registry also apply to the Metabase. Every time you open the IIS Manager (Start/Programs/Administrative Tools/Internet Services Manager) or IIS Snap-in (add Internet Information Services to a Microsoft Management Console MMC) to make changes, you are in fact changing settings for each site that are stored in the Metabase. There are more than these two ways to make changes to a web site, but first a backup of the Metabase should be made. Since the Metabase is separate from the registry, it must be saved separately.

To create a backup from the IIS Manager, right click your **server** and select **Backup/Restore Configuration**.



You now have the options Create backup, Restore or Delete (previous backups). Once you click on the Create backup, you need to name it; the date or version of your site is a good idea here, and your all set.



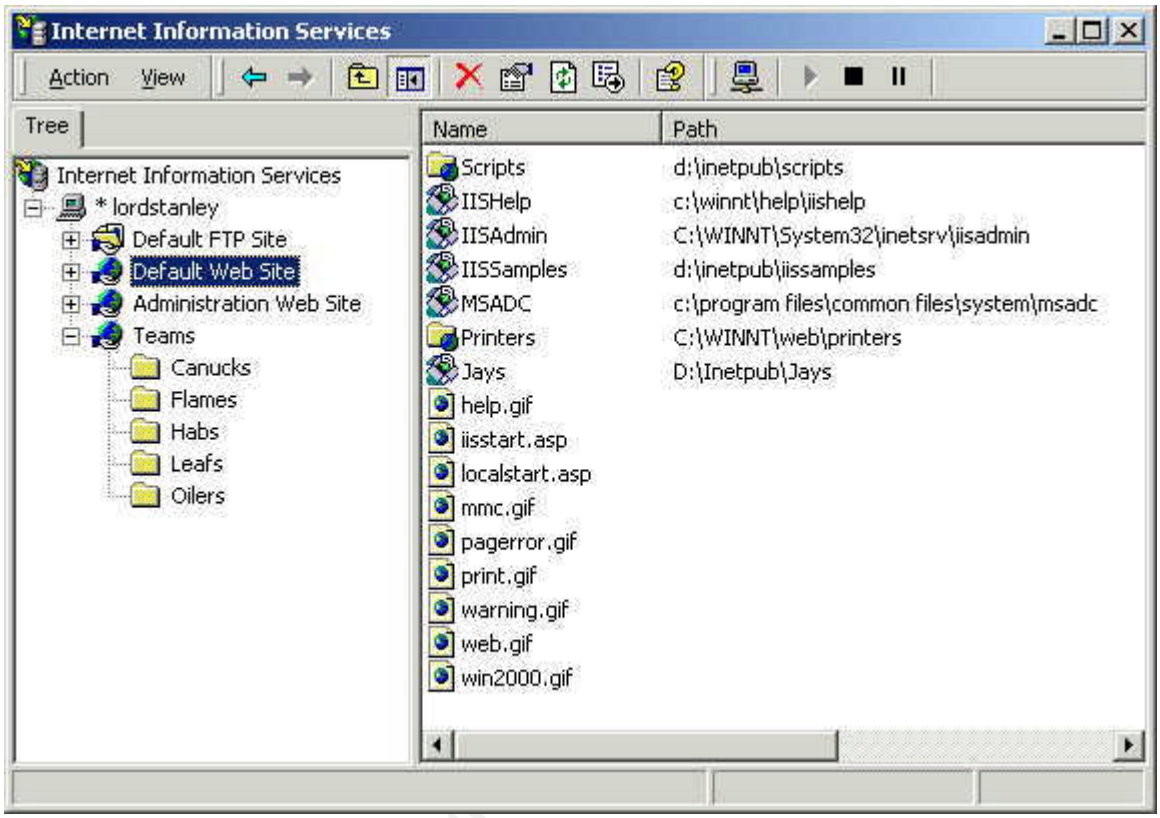
Now you can go back to this to restore your working Metabase anytime. There is more than one way to do this. Using scripts this process could be automated, to provide even more protection against a corrupt Metabase or to create customized backup tools. This would also be useful to store multiple backup versions and restore the version of choice. We will talk about scripting the Metabase in the next section.

Manipulating the Metabase

Manipulating the Metabase can be done many ways. If you've used IIS, then you have used one of the two most common tools: IIS Administration Console or the IIS Administrative Snap-In. The most common changes can be made to the Metabase through these consoles. They are also built to make the proper changes to the appropriate keys to initiate a change. Some properties have dependencies that need to be changed too. There are many tools that built to make changes to the MetaBase and also programmatic interfaces in IIS itself to provide easy access for scripting or writing your own applications. In this section we will talk about some of the most common tools, how to use them and what to use them for.

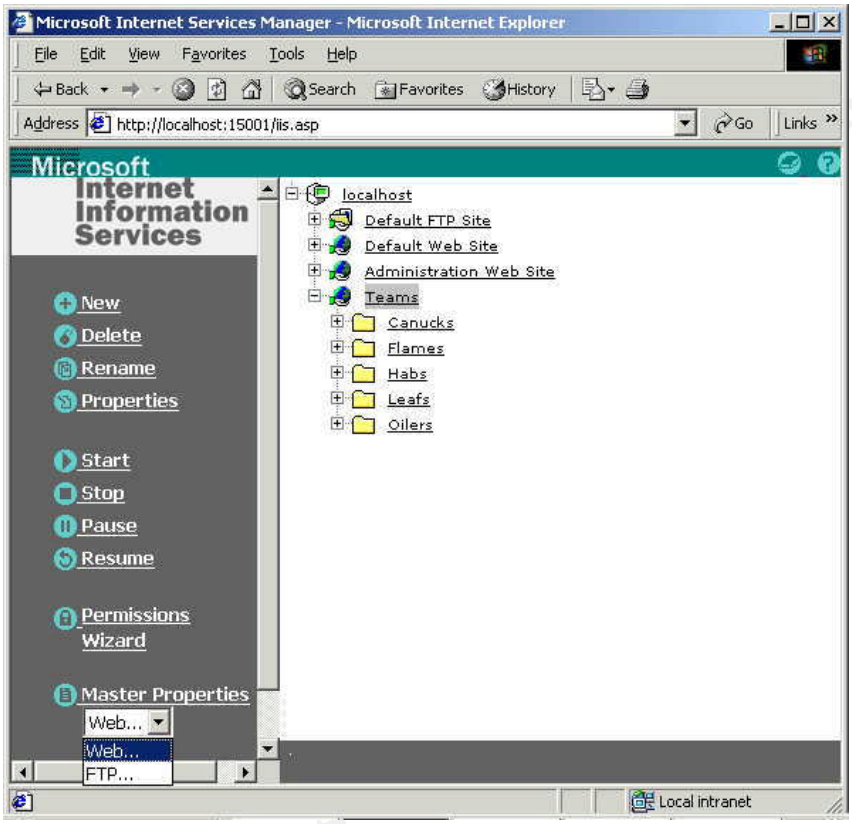
The Internet Services Manager found in **START/Administrative Tools/Internet Services Manager** is the most common tool used when working locally on the

web-server. Any common and some more advanced tasks can be added, deleted or modified through this interface.



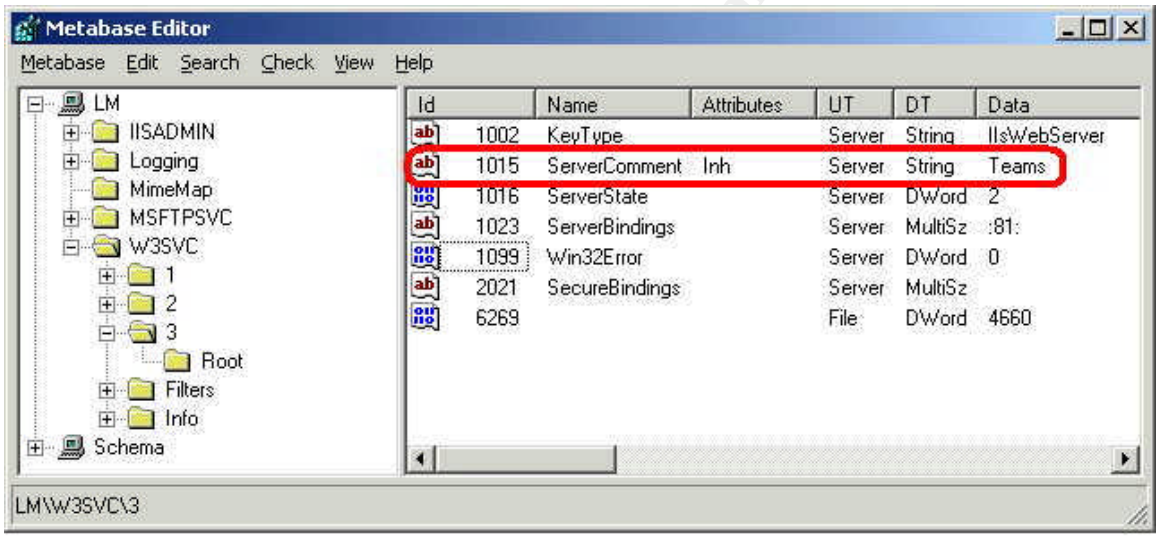
Remote administration is done most commonly through the Internet Information Services Snap-In, which provides the same interface as the IIS Admin Console except you can be on a remote system. This is used from a Microsoft Management Console (MMC)

An HTML Administrative site is also available, and by default, can be accessed at **http://localhost:15001** (the port number will differ according to your system). This site has ALL of the functions of the Administrator Console so be very careful to protect it. The best practice is to delete it all together. If it needs to be used make it a secure site that only local administrators can access. By default, the administrative web site is configured to only allow local access. To secure access from remote machines **IP Address Access Restrictions** can be made to specific IP addresses that you want to grant access.



MetaEdit

The Metabase editor can be used to change any value in the Metabase, that some of the Administrative Consoles do not have options to edit. This program can be found on the IIS Resource Kit and be installed with the MetaEdit setup program. By default, it will install to **\Program Files\MetaEdit**. The RegEdit-like interface is familiar and easy to navigate through. However, as I mentioned earlier some properties have dependencies so do not change anything unless you know exactly what it will affect. Consult Microsoft documentation for details on specific properties. A single change could have disastrous effects on the web server. A common question about the physical layout of a web server is, "under the W3SVC folder there is all of these numbered folders, how do I relate them to a specific site?" If we look under the **\LM\W3SVC\(site #)** the key: ID - 1015 *ServerComment* is where the Description for each site is stored.



The Adsutil is a command-line utility that uses VBScript with Active Directory Service Interfaces (ADSI) to display or change Metabase content. The documentation for Adsutil can be found on your web server here **http://localhost/iishelp/iis/htm/adminsamples/adsutil.htm**.

Usage:

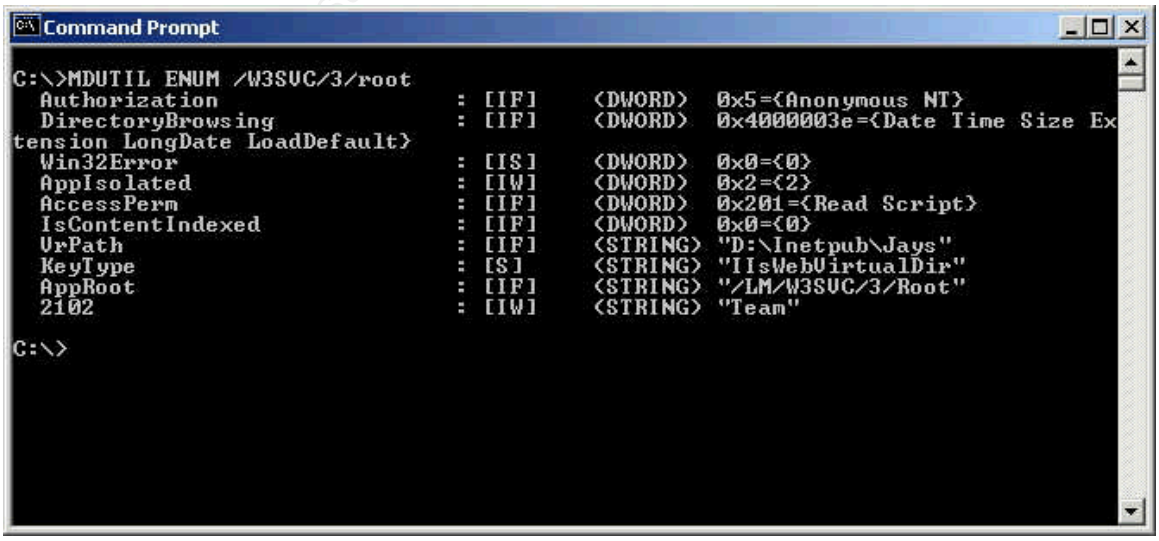
adsutil COMMAND <path> [<param>...]

Commands:

GET *Path* Display chosen parameter.

SET *Path Value*	Assign a new value.
ENUM *path ["/P" | "/A"]*	Enumerate all parameters for the path. /P - Enumerate the paths only (no data). /A - Enumerate all data that can be set on the node.
ENUM_ALL *["/P" | "/A"]*	Enumerate all parameters. /P - Enumerate the paths only (no data). /A - Enumerate all data that can be set on the node.
DELETE *path*	Delete the path or parameter.
CREATE *path [KeyType]*	Create the path and assign it the KeyType.
APPCREATEINPROC *Path*	Create an in-process application.
APPCREATEOUTPROC *Path*	Create an out-of-process application.
APPDELETE *Path*	Delete the application (if present).
APPUNLOAD *Path*	Unload an out-of-process application.
APPGETSTATUS *Path*	Get status of the application.
FIND *Path*	Find the paths where a parameter is set.
START_SERVER *Path*	Starts the server.
STOP_SERVER *Path*	Stop the Web site.
PAUSE_SERVER *Path*	Pause the Web site.
CONTINUE_SERVER *Path*	Unpauses the Web site.
HELP	Prints all available commands.

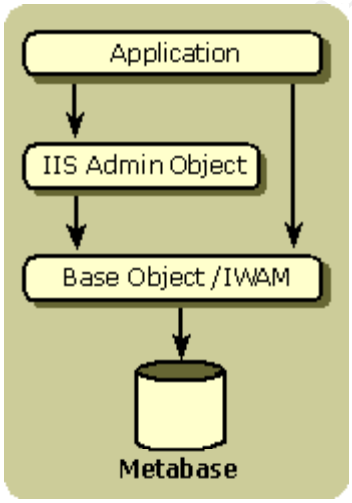
Mdutil.exe is another great tool included with Windows 2000 (although it is not new with Win2K, it is also included with the NT 4.0 Option Pack). It can be found in the i386 directory called mdutil.ex_. With a simple extract, you're ready to go. This utility enables the manipulation with metabase parameters from command line.



The example above shows mdutil enumerating the properties of the /W3Svc/3/root metabase path. The ENUM will enumerate all parameters for a given path that doesn't have inheritance turned on or that have be specifically changed. The left-hand column shows the friendly name for each parameter ID. All parameter IDs don't have a friendly mapping in mdutil so only the ones listed in a mdutil.exe –help, will show up named. Notice that the last mapping, ID 2102 isn't represented by a friendly name, but since the ID is in the 2000 range we know that it is owned by the HTTP server. It is actually the AppFriendlyName property, notice the "Team" data stored. The second column indicates the flags and the user types associated with each property. The *Authorization* property, is inheritable with a File user type, shown by the [IF] listing. The next column shows what type of data is stored in each metabase record and the final column is the actual data. By using the mdutil.exe SET command we can set metabase properties, as I mentioned earlier setting individual properties could be hazardous to your system so use this cautiously.

If you don't like any of these tools that we have talked about and if you have the skill and the time, you can write your own applications to do anything that you would ever wish to do to your IIS Server. The most powerful way to control your IIS web server is through its programmatic interface. Scripts can be written to perform a number of tasks that can't be done easily through the other standard interfaces. These can be executed in an ASP file, manually from the command prompt or scheduled with the at.exe.

There are two ways to code against the Metabase: using **IIS Admin Objects** or **IIS Admin Base Objects**. Although there is subtle difference in the names, they provide two different levels of access to the Metabase from a programmer's point of view.



IIS Admin Objects

IIS Admin Objects are based on Microsoft Active Directory Service Interfaces (ADSI), and can be easily accessed through scripting languages or any language that supports automation. Languages like Visual Basic, VBScript or JScript use the exposed properties and methods from IIsWebServer for web servers and IIsFtpServer for FTP servers, etc. From creating backup scripts; to logging; or a script that will add a "boiler-plate" web site to the server, it can all be done simply to automate any operation that would be needed. There are many scripts included with IIS that use this interface to cover most of the basic (and some advanced) actions. An example that we've already looked at is the Adsutil.vbs; all of the functionality is achieved with the ADSI interface to the IIS Admin Object. Also with the IIS Resource Kit there is all kinds of scripts included, not to be used "as is" in your particular setup, but can be used as a base to write more advanced scripts to meet you needs.

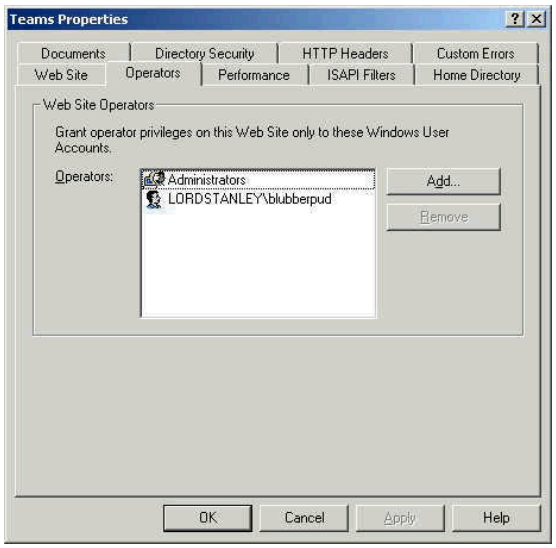
A simple example of a useful script is the following Backup Script:

```
'initialize variables
Dim BUName, BUVersion, BUFlags, CompObj

'default values
BUName = "Backup"
BUVersion = &HFFFFFFFF          'use next available version number
BUFlags = 0                     'no special flags
Set CompObj = GetObject("IIS://Localhost")

CompObj.Backup BUName, BUVersion, BUFlags
```

Short and simple but still effective. First we declare some variables and set values to them. Next, we create an instance of the IIS://Localhost object. By passing the GetObject function an ADSI name, the IIS: portion indicates the name space provider that ADSI should use to get the object. Anything that follows the Localhost will be the path within the name space. The Localhost is the root object of the metabase hierarchy and in the IIS Admin Objects reference section the Localhost object is referred to as the IIsComputer object. After we obtain a reference to the IIsComputer object for the machine, the Backup method of that object is called. This method will make a backup copy of the METABASE.BIN file and store it in C:\WINNT\system32\inetsrv\MetaBack.

A more advanced backup script is included with the default IIS install in the Inetpub\iisamples\sdk\admin folder. It is called metaback.js or metaback.vbs and can be used as a template to automate versioned backups of your Metabase to be restored if the situation arises.

Scripts can only be run by Web Site Operators. Operators are added or removed though the Operators tab in a web site's properties. By default only Administrator are added to this.



Using IIS Admin Objects from ASP pages could result in errors if proper permissions aren't assigned or if the Application is configured to run in its own process space. Generally, applications should not be configured to run in its own process space and anonymous authentication should not be used when accessing any of the script files. If you do run your application Out-of-Process or in its own area of memory, a "Server Application Error" message will be returned. Trying to execute under an account without permissions will result in a "Invalid Syntax" error.

IIS Admin Base Objects

The second way to access the metabase is to write your own applications written in C/C++. This provides a more "low level" entry point for your C/C++ applications for more advanced development choices.

A example of an application to change a virtual directory's path property using the IIS Admin Base Object:
'from Running Microsoft Internet Information Server, Microsoft Press'

```
#define INITGUID

#include <windows.h>
#include <iadmw.h>
#include <iiscnfg.h>

void main()
```


© SANS Institute 2000 - 2005 Author retains full rights.

```
{
        IMSAdminBase *pIABase;

        CoInitialize(NULL);
        HRESULT hRes = CoCreateInstance     (CLISD_MSAdminBase, NULL,
                                             CLSCTX_ALL,
                                             IID_IMSAdminBase,
                                             (void**)&pIABase);
        METADATA_HANDLE metaHandle;
        pIABase->OpenKey (METADATA_MASTER_ROOT_HANDLE,
                          L"/",
                          METADATA_PERMISSION_WRITE,
                          5000,
                          &metaHandle);

        WCHAR path[] = L"d:\\inetpub\\newdir";
        METADATA_RECORD MDRecord =      {MD_VR_PATH,
                                         METADATA_INHERIT,
                                         IIS_MD_UT_FILE,
                                         STRING_METADATA,
                                         sizeof(path),
                                         (UCHAR*)path,
                                         0};
        pIABase->SetData (metaHandle,
                          L"/LM/W3Svc/1/root/NewVRoot",
                          &MDRecord);

        pIABase->CloseKey (metaHandle);

}
```

This application would change your virtual directory's path property using the IIS Admin Base Object. The Advanced Programmatic Administration section in the Platform SDK or the Microsoft Internet Information Service 5.0 Documentation Book is great references for more information.

From what we have talked about in the last section, we can see that there are many different ways to manipulate the Metabase. Looking at the Microsoft Internet Services Manger Console to programming a C/C++ application to produce customizes applications; I have shown the spectrum that an administrator or developer has offered to them to complete their tasks.

Securing the MetaBase

Now you have your web server up and running. You've gone through a tedious and detailed installation of Windows 2000, installed Internet Information Services 5.0, applied all sorts of Service Packs and Hot fixes. Your web site has been built and moved to the server, you've even written some custom scripts and applications that manipulate the Metabase to manage you web server. What should you do next? With security in mind, the last question should always be "is your web server as secure as it can be?" Even after all of the work that's been done on the server we cannot forget to lock down and secure the Metabase. We wouldn't forget to take the appropriate actions to secure the Windows registry, so on a web server the next logical step is to lock down and secure the Metabase.

NOTE: All of the following is a guide. Make sure that any changes made are tested before they are put into a production environment. Take as much care as you would doing anything to the windows registry.

One advantage that an attacker could have is familiarity with the product. Obviously, if an attacker knows where all the critical files are this will provide him/her with a great starting point. What we what to do to take this advantage away is move and possible rename any file or program that could be dangerous if used maliciously against our own web server.

Move HTTP/FTP folders

Double check that the HTTP/FTP root folders have been moved off the %systemroot% volume. If they haven't, move them to a different partition, and adjust paths in the Internet Information Server Management Console.

Move the MetaBase

Moving the Metabase is an important step to securing it. Since IIS stores all of its settings and configuration including passwords (in clear text), if copied an attacker could get any information he/she could wish for.

Moving the Metabase, by default is installed in %SystemRoot%\system32\inetserv folder. The location of the METABASE.BIN file is determined by the following registry value:

Hive: HKEY_LOCAL_MACHINE
Key: \Software\Microsoft\InetMgr\Parameters
Value Name: MetadataFile
Value Type: REG_SZ
Value Data: <drive letter, path, and file name of the metabase

- make a backup of the MetaBase using any of the methods that we discussed earlier
- stop IIS using the net stop IISAdmin /y command
- Using Regedt32, we can edit the MetadataFile value to the new location and name you want.
- While you have Regedit32 open change the security on the InetMgr Key to Administrators - Full control
- Copy the MetaBase.bin to the new location and rename it to what you would like, something that doesn't scream out "I'm important" is good, the extension can also be changed (be careful, be sure to test with any of your applications when done)
- change the NTFS permissions on the new folder to Administrators – Full Control, SYSTEM – Full Control
- audit any failed attempts on this folder
- restart the IIS Services
- delete the original MetaBase.bin file

Secure the MetaBack folder

Secure the MetaBack folder located %systemroot%\system32\Inetsrv\MetaBack. This is the default location that MetaBase backups are stored in a file with the .MDO extension. A backup copy of the MetaBase could be just as valuable to an attacker as the current one.

- perform a backup, after IIS is configured
- set NTFS permissions on the folder to Administrators – Full Control, SYSTEM – Full Control
- audit all failed access on the \MetaBack folder

Secure CScript.exe

Lockdown the Cscript.exe. This is the executable for .vbs files and could be used to execute malicious scripts; Admin scripts or even your own scripts against you.

- set the NTFS permissions on %systemroot%\system32\cscript.exe to Administrators – Full Control

Secure iissync.exe

There is an application called iissync.exe that is used in a cluster environment to replicate MetaBase settings to a clustered node. Someone with access to the box could use this command to send a copy of the MetaBase to any server

running IIS on the network.

- if the server is not in a cluster, delete the iissync.exe
- if the server is in a cluster, it would be a good idea to move the file into a new folder with NTFS permissions set to Administrators – Full Control, and possibly rename it

If you are using or plan to use any of the AdminScripts, this folder contains all of the administrative scripts for IIS.

- move and rename the inetpub\AdminScripts folder and scripts to a new folder
- apply NTFS permissions to Administrators – Full Control
- audit any failed attempts on this folder

If MetaEdit is installed it is installed to \program files\metaedit

- using regsvr32 with the –u switch unregister the MetaUtil.dll – **regsvr32 –u MetaUtil.dll**
- move metaedit.exe and metaUtil.dll to the %systemroot%\system32\Inetsrv folder
- reregister the MetaUtil.dll – regsvr32 – **regsvr32 MetaUtil.dll**

Conclusion

With an better understanding of the MetaBase and what it does, the tools available to manipulate it, I hope, you will appreciate what needs to be protected with the MetaBase in mind. Not only can your web server installations will be more secure, administration can be simplified with the many tools we discussed or automated with the many scripts available. So just remember, if you are ready to let you IIS web server loose to the Internet, make sure it is as secure as it can be.

Sources

1. Fossen, Jason. “Windows 2000: Internet Information Server”, 2001. http://www.sans.org

2. Microsoft Press. “Microsoft Internet Information Server 5.0 Documentation”, 2000. http://www.microsoft.com

3. Braginski, Leonid & Powell, Matthew. “Running Microsoft Internet Information Server” 2001. http://www.microsoft.com/TechNet/iis/

4. O’Brian, Gerry “Controlling IIS Programmatically”, 2001

5. Yegulalp, Serdar “Tweak the IIS MetaBase the Easy Way”, March 8, 2001. http://www.PlanetIT.com/docs/PIT20010308S0018

6. Microsoft “Microsoft Internet Information Services, IIS 5.0 Documentation”, 2000 http://www.microsoft.com/WINDOWS2000/en/advanced/iis/htm/asp/aint94d.htm

7. Courington, David S. “A Step-by-Step Guide to Securing Windows 2000 for Use as an Internet Server”, March 29, 2001 http://www.sans.org

8. Spencer, Ken “IIS Resource Kit Utilities”, October 1999. http://iisadministrator.com/Articles/7173.htm