The dangers of shared hosting services

How ASP (Active Server Pages) can easily allow total control of the server by any anyone who has access to the shared server.

Picture this:

company "XYZ Ltd" has just purchased a co-hosting package from a leading UK ISP with the following features:

- Windows 2000 Operating System
- FrontPage support
- ASP (Active Server Pages)
- an SQL Server database
- 100Mb of web space

As per current practices, "XYZ Ltd" signed a contract with the ISP that makes it liable for anything that could happen with or though the rented web space.
In return, the ISP will endeavour to maintain its network and the server as reliable and secure as possible.

Given such strong claims and the ISP's public concern and awareness of security, company "XYZ Ltd" was confident that their website content (public pages, private pages, database content, database access codes, client personal information, etc…) was secure and only visible to authorized users.

But what would company "XYZ Ltd" (and their clients) say if they knew that the Windows 2000 web server used to host its website allowed anybody with an account in that server (i.e. all ISP clients that host their website there) to do the following:

- find sensitive information about that server such as
  - server setup
    Computer Name, Operating system, Type of Processor, Number of processors, All Users Profile path, Common Program Files, System Drive, System Root, User Profile Path, and others
  - user information such as:
    User name, Full Name, Comment, Country code, Account active, Account expires ,Password last set, Password expires, Password changeable, Password required, User may change password, Workstations allowed, Logon script, User profile, Home directory, Last logon, logon hours allowed, Local Group Memberships, Global Group memberships
  - password policies
  - open shares
  - other computers on the network
- Browse the entire server hard drive (including company "XYZ Ltd" directory and files structure)
- See (i.e. type) the contents of files (including company "XYZ Ltd" files that contain sensitive information such as: ASP source code, database access codes and "members only" web pages)
- Copy company "XYZ Ltd" files to other parts of the hard-drive and (if desired) upload or email those files to somewhere on the internet
- upload Trojans to the server that could allow remote access or control over the server

BOX 1 – GLOSSARY

ASP (Active server pages) – An Active Server Page (ASP) is an HTML page that includes one or more scripts (small embedded programs) that are processed on a Microsoft Web server before the page is sent to the user. An ASP is somewhat similar to a server-side include or a common gateway interface (CGI) application in that all involve programs that run on the server, usually tailoring a page for the user. Typically, the script in the Web page at the server uses input received as the result of the user's request for the page to access data from a database and then builds or customizes the page on the fly before sending it to the requestor.

Shared hosting - is Web hosting in which the service provider serves pages for multiple Web sites, each having its own Internet domain name, from a single Web server. Most Web hosting companies provide shared hosting. Although shared hosting is a less expensive way for businesses to create a Web presence, it is usually not sufficient for Web sites with high traffic. These sites need a dedicated Web server, either provided by a Web hosting service or maintained in-house.

It sounds surprising that such sensitive information is so easily available, but the problem is widespread. And whilst this is not a major problem in a server hosting only the website of *one* company, this becomes a serious problem in the scenario we are presenting: a Shared Server hosting dozen of websites from different companies.

We will now run through a practical example of what we discovered whilst performing a security audit for a client whilst working on a website development project.

The client (XYZ Ltd) needed a new website so we (in partnership with other companies) worked with them on the planning, information architecture, design, implementation and finally hosting.
When we were given the user details of the hosting server (a shared server with the same specifications as the ones mentioned earlier) we decided to test the server's security.

As providers of security solutions, we performed those tests to ensure the ISP's claims about their security was true, and to reassure our client about the safety of their data.

Step 1: Directory Browsing and see File Contents

The first test performed was to see if we could see more files on the server other than the ones we were allowed to.

With shared hosting each client should only be allowed to read and write into the directories allocated to them (for example in c:\inetpub\wwwroot\client_XYZ)

So we wrote a little ASP (Active Server Page) script that could read the directory structure of the server and see the contents of files (see box 2).

And sure enough, due to lax security configurations on the server we were able to see the directory structure and the files contents. Why? Because the user account dealing with our requests had more access rights than it should have, This is a basic security misconfiguration.

But what's the fuss, you might say?

The problem is that once we were able to 'navigate' at will on the server and see most files we wanted, we could do the following:

1. see all web pages hosted on that server, regardless if they where in public or private (password controlled) areas
2. see the contents of all scripts pages (ASP, INC, etc) that showed us where the databases where located and what where the access codes (the 'global.asa' is also used to store this information)
3. using the access codes found in the previous point we where able to login successfully into several SQL Server databases. Note that we had FULL CONTROL over these databases since we where using authorized accounts
4. we where able to download to our home computer ENTIRE ACCESS DATABASES containing user information (such as login details, addresses, etc...) because their only protection was being located in a hidden URL (for example: www.company_xyz.com/admin/database/Company_XYZ_Database.mdb). This is a very basic security mistake, because once the source code of the script files that use that database is known, the database is downloadable from the internet

BOX 2 – ASP SCRIPTS

How to read any directory contents from the server with .ASP code?

a. First create an object with the functionality to access the server
   Set my_Server_Object = CreateObject("Scripting.FileSystemObject")

b. Then open another object with the contents of the folder you want to see (in this case *("C:\inetpub\wwwroot")*
   Set my_Folder_Object = my_Server_Object.GetFolder("C:\inetpub\wwwroot")

c. then load the list of directories into a variable
   Set my_Sub_Folders_Object = my_Folder_Object.SubFolders

d. and the list of files into another variable
   Set my_Files_Object = my_Folder_Object.Files

How to read any file from the server (i.e get its contents)?

a. using the same object as before
   Set my_Server_Object = CreateObject("Scripting.FileSystemObject")

b. create an Object with the desired file *("C:\config.ini")*
   Set my_File_Object = my_Server_Object.GetFile("C:\config.ini")

c. Open the file for reading
   Set my_File_Contents = my_File_Object.OpenAsTextStream(1)

d. Output the contents of the file into a variable
   FILE_CONTENTS = my_File_Contents.ReadAll

Step 2: Upload File contents

The next test we did was to see if we could upload files to the server. As before the only areas that we should be allowed to upload files

was to our client's area.

We uploaded an .ASP script called 'upload.asp' which as the file name indicates is a script that allows the upload of files to the server.

We changed in the 'upload.asp' script the target directory of the server (i.e. where the file is saved once it is uploaded) to "C:\" (the root of the C Drive).

And we tried to upload a file. And it worked!! We where able to upload files to an area that should be restricted to administrators only.

This is another example of lax security practices on that server. The C: drive by default has write access to EVERYONE, which means that the user account being used to process our script (upload.asp) also has write access to this area

Step 3: Copy Files internally

Now that we knew we could write files to the C:\ directory (and several others) we tried to copy files (using another function from the 'Scripting.FileSystemObject' object) from outside our working directory (for example a file from another website also hosted in that web server) into the C:\ directory.

And, we were also able to do it. Again, because the user account dealing with our requests had too much access rights than it should have had, This is a basic security misconfiguration.

Step 4: Email Files from the server

At this stage it wasn't very surprising to find that we could use the email Object (kindly supplied by the ISP to enable our Client to have pages in his website with forms that automatically send emails...) to send an email to us (or to anyone) with files as attachments located anywhere in the server.

In practical terms. we could email to you the database we discovered (containing usernames, passwords and addresses) that belongs to another website also hosted in that server!

Step 5: Execute commands in the server

Stepping up a gear, we decided to check if the Wscript.shell was there and if so, if we could use it to execute commands (see Box 3)

And, as before the test worked. Now we could execute commands in the server

We could do this by hand coding, but using a tool was much easier.

CmdAsp.asp is an interactive ASP page command prompt which executes cmd.exe. The cmd process, as the ASP page, executes in the context of the web server and makes it easier to assess your exposure to these user accounts.

Example of commands that can be executed this way:

- *ipconfig –all* : shows network configuration
- *set* : shows information about this server
- *net users* : displays the list of users (i.e. usernames)
- *net localgroup* : shows existent security groups
- *net view :* list of computers in the Local Area Network (LAN)
- *net accounts* : details about how accounts are managed (for ex: password policies)
- *net share* : list of open shares in this computer
- *net start :* list of services (i.e. programs) that are currently running
- *ping 10.1.0.1* : send a ping to a local machine (small message to test connections)

BOX 3 – How can you run commands on the server?

When you use the run method of the Wscript.shell object in an ASP page that is running in-process (in Inetinfo.exe), the command that is called is executed in the system context

The following ASP scripts creates a directory called 'temp' in the root of the C:\ drive

```
<%
    Dim oScript
    Set oScript = Server.CreateObject("WSCRIPT.SHELL")
    Call oScript.Run ("cmd.exe /c mkdir C:\temp", 0, True)
%>
```

In the procedure of securing a web server, it is important to understand your level of risk from operations performed by IUSR_COMPUTER, IWAM_COMPUTER and SYSTEM user accounts - the accounts that IIS will use to execute scripts such as ASP or Perl - and locking them down appropriately.

Step 6: Execute files in the server

Since now we could upload files and execute them , the next test was to see if we could execute a program uploaded by us.

To do this test we used the famous NetCat (tool that is able to write and read data across TCP or UDP connections) which would allow us to run a shell (command prompt) in the server and connect to it from our computer

- using 'upload.asp' we uploaded nc.exe to the c:\temp directory of the server
- using 'cmpasp.asp' we executed nc.exe on the server ("c:\temp\nc.exe -L -d -e cmd.exe -p 2003 " this command bind the cmd,exe program to the port 2003)
- finally, in a command prompt in our computer we executed the command "nc {server IP} 2003"
- and we were in: we now have a remote command prompt on the server

At this moment we have a shell in the server's running with the privileges of IWAM_COMPUTER

We also decided to stop our security research, but from here on in, it is not very difficult to have total control of this server (i.e. gain administrator rights)

Step 7: We contacted the ISP and told them about the problem

At this stage we contacted the ISP and explained to them the problems we discovered and how to fix them.
We didn't tell our client about this problem since their immediate reaction would be to either cancel the current contract or sue the ISP.

This could have been a very serious problem because if someone with malicious intensions (i.e. a hacker) had the same access we had (a valid user account in that server) they could:

- deface all websites existent in that server (change home page, delete files, etc…) either with
  - political or satirical propaganda
  - content that could get the web site owner sued
  - illegal material (child pornography or racist content)
- install virus or Trojans in the websites (which would be picked up by the website's users)
- collect personal user information
- change the prices of goods and capture credit card details (on the sites with e-Commerce)
- use that web space to host illegal material

Scary isn't it?

But wait! you say, this only happens if:

a) one of the ISPs clients with an account in that server has malicious intentions
b) the attacker discovers a valid username and password with access to that server

The problem is that the scenario presented is not that far fetched because:

- the ISPs made no screening whatsoever to avoid a)
- few people have good password management (from simple passwords, to writing it on little papers)
- most account details are sent un-encrypted over the internet (when you FTP into your site or read your email, the password is visible to anyone that is listening)
- most servers allow brute force password attacks (i.e. try thousands of password combinations until one works)

So what is the solution: Implementation of **IT SECURITY BEST PRACTICES** and the use of IT security companies that will test the servers using the a hackers approach.

By thinking like a hacker, you can make their life a lot more difficult !

Dinis Cruz
IT Security Consultant and Trainer
dinis@ddplus.net

ddplus: consultancy, security, training
www.ddplus.net

Is your computer protected from hackers?
http://www.ddplus.net/DDPlus_Website/IT_Security/Check_Your_system.htm