Global Information Assurance Certification Paper

Mark Riederer
7th December 2003
GCES Practical Assignment, version 1.4b
Option 2 – Case Study in Information Security

Creating a Secure Inter Company File Transfer System

Index

1. Abstract

Acme is a large multi national company with the requirement to exchange large amounts of data with many different clients and vendors via the internet. This paper will outline how Acme replaced its insecure Internet facing FTP server, with a secure internet facing file transfer system.

2. Introduction

This document will cover what is currently in place within the organisation, namely an insecure FTP server; a brief outline of the FTP protocol, laying out the advantages and disadvantages of using the FTP protocol and the FTP installation that currently exists within Acme. It will then go on to look at alternatives that exist to plain old FTP covering both commercial and "free" solutions, such SFTP, HTTPS based systems and hardened FTP systems, It will also include there pros and cons. Finally, it will outline the solution that Acme chose to implement which is based on a hardened Windows 2000 server running IIS5.0, with HTTPS and WebDAV installed.

3. Current File Transfer System

For the past decade Acme has had one Internet facing FTP server located at their head quarters in the USA which is used for transferring data to and from clients, vendors and customers. It has been only occasionally used by the EMEA (European, African and Middle East) based projects and IT centre, mainly due to administrative issues and problems associated with time zone differences such as:

- No SLA's for setting up new accounts and directories – best endeavour
- No defined support procedures
- No points of escalation
- No ability to assist with large transfer requirements due to lack of storage.

The FTP server is based on a Microsoft NT 4.0 platform with SP6a installed, running IIS 4.0, with everything, but FTP disabled. It resides within the USA corporate DMZ behind a clustered firewall installation and its IP address is NAT'd for internet access. The IP address which is translated via the use of NAT, is published as an Internet DNS name to assist users with accessing it.

The FTP home directory is subdivided into two separate top level directories called Public and Publish.

- Public – Allows any user to download files that. An account is required to publish data in this directory.

- Publish – This directory contains project, customer and vendor specific directories. To be able to download data to any of these directories, the user requires an account with the correct permissions to access the directory.

The fact that anonymous access is allowed is not ideal, but the USA IT department seemed prepared to turn a blind eye to this until the increasing popularity of downloadable music (MP3) and video files impacted on the USA based corporate FTP server resulting in litigation from a music industry anti-piracy body. At this point, with money now being an issue, the security of the FTP server, from both a technical point of view and an administrative point of view, was deemed insufficient.

This combined with the increasing number of requests from EMEA based projects for FTP type services, some in the tens of GB's range, has resulted in the EMEA IT Centre solutioning, designing and placing into production, its own Internet facing file transfer system.

4. The FTP Protocol

In the last couple of years the FTP protocol itself has also taken on the label of being insecure, but it is still widely used today. What follows is a brief history and description of FTP, and a more detailed analysis of its faults.

According to RFC959[1], the first incarnation of the FTP protocol came to light in 1971 as RFC114[2] which was designed to allow hosts at M.I.T. to transfer data between themselves. The RFC that covered FTP as we know it today (RFC765[3]) was published in 1980.

FTP is a TCP based protocol designed to transfer files over any IP capable network. The use of TCP ensures the reliable delivery of data.

[1] Postel, J. and Reynolds, J. http://www.faqs.org/rfcs/rfc959.html

[2] Bushan, A. http://www.faqs.org/rfcs/rfc114.html

[3] Postel, J. http://www.faqs.org/rfcs/rfc765.html

The objectives of the FTP protocol are detailed below;

1. To promote the sharing of files
2. To encourage the use of remote computers
3. To account for variations in file storage systems found on various hosts
4. To transfer data reliably and efficiently

In all of the points FTP does an admirable job.

So how does FTP work? Two different TCP ports are used during a standard FTP transfer, usually ports 20 and 21. Port 21 is the control port and the TCP session setup using this server port is used to pass all the control commands (cd, lcd, username and password, etc). Port 20 is the data port and any sessions created using this are used for transferring data. There are actually two different forms of FTP transfer, Active and Passive and the differences between the two are described below.

In an Active FTP transfer the following methodology is used.

- Upon initiating the FTP command the client allocates two high port numbers (greater than 1024).
- The client then initiates a connection from one of the high ports numbers to port 21 on the server, which as described above is the normal FTP control port.
- Once the client has been authenticated by username and password (anonymous users can be allowed) the connection is established.
- The client then sends an FTP PORT command to the server on port 21 which notifies the FTP server of the second high port number allocated by the client.
- The server then initiates a connection from port 20 to the client on the port provided by the client using the FTP PORT command. This is the data session.
- Once this connection is established the data transfer will then take place.

In a passive FTP transfer, the initiation is similar to that of an Active FTP transfer, but instead of issuing a PORT command the client uses the PASV command. This instructs the server to use a high port of its own for the data connection rather than then more common port 20 used in an Active FTP transfer. The server then replies to the PASV command on the control channel notifying the client of the new port number to be used for the data session. The client then uses this high port number to establish the data session. PASV FTP is what you will find most web browsers support.

So in an Active FTP transfer the client initiates the control session to port 21 and the server initiates the data session from port 20. In a Passive FTP transfer the

client initiates the client session to port 20 and then initiates the data session on the port provided by the server with the PASV command.

This whole process is described in detail in RFC959 and numerous other sources/

5. Common FTP Vulnerabilities

So why is FTP considered so insecure?

There are a number of non FTP specific vulnerabilities that can be used against FTP servers such as DNS Cache Poisoning and ICMP redirects. Likewise incorrectly configured platforms (UNIX, Microsoft Windows) and unpatched systems can also provide further insecurities on an FTP server. These can all be negated by following industry standard practises during server builds and ensuring that OS and application patching is kept up to date.

More FTP specific vulnerabilities include;

- Username and password is the only means of authentication. This leaves the FTP server and any other systems protected by the same password open to brute force and dictionary attacks.
- Everything (date, username and passwords) is transmitted in clear text. This means that the session can be easily captured using a packet capture tool, the data stolen or altered and the username and password used for purposes other than it was intended.
- No server identification. How can you be sure that you are uploading or downloading your data to the legitimate server?
- The PORT command has a vulnerability associated with it; this includes the FTP Bounce attack and bypassing dynamic packet filtering devices.
- PASV command vulnerability
- Numerous other FTP vulnerabilities which are OS dependant both client and server.

The Cert[4] website details many of these and other vulnerabilities in more detail.

Whilst many of these can be negated by making sure that both the client and server are correctly patched, the first 3 vulnerabilities are flaws which cannot be overcome when using the FTP protocol. In particular the second vulnerability, the transmission of everything in clear text, is the most serious in a large enterprise.

[4] Cert. http://www.cert.org/nav/index_red.html

Using some simple packet capture techniques, it is possible to not only capture the data itself but to gain the username and passwords used to authenticate the FTP connection.

Whilst the security of the data itself very much depends on the sensitivity of the data, the username and password are of utmost importance.

At the very least, if the user database is stored locally on the FTP server, the inappropriately obtained username and password would then allow the unsolicited upload and download of files which could include MP3, video files and warez. This could result in legal action concerning copyright breaches, confiscation of equipment in the case of obscene material, or at the very least an FTP server with no spare storage.

If the username and password were a network username and password such as domain credentials from a Microsoft Active Directory managed domain, the consequences could be even more extreme as the 'hacked' credentials could be used to gain access to other network devices.

6. Improving File Transfer Security

So what can be done to improve the security of file transfers between third parties?

It goes with out saying that the File Transfer server should be deployed in an area segregated from the rest of your internal network. A traditional DMZ being the ideal location and the assumption is being made here that the DMZ and DMZ hosts are protected as most DMZ's are with a variety of different security products and procedures, including firewalls, IDS, etc.

As has been mentioned previously, the server should be locked down (including having all un-required services and applications removed). It is of vital importance to keep the server OS patching up to date, as this should take care of any known OS vulnerabilities. An Anti-Virus package should also be deployed on the server to not only ensure that the server is not infected, but that none of the files being transferred to and from the server are infected. A host based IDS should also be considered, but be warned that this could add considerable complexity to the server installation and would require a lot of effort to use correctly.

6.1 Hardening Existing FTP Installations

There are a number of things that can be done to improve the security of an FTP server. Organisations such as Cert and the National Security Agency all have documents on this and hardening server operating systems in general.

For simple FTP server hardening the best advice I have found is that given by Ray Zadjmool in his article 10 Steps to a Secure FTP Server[5]. Whilst it is focusing on Microsoft Windows systems, the 10 steps can also be applied to UNIX, Linux and any other FTP Server system. The 10 steps are;

1. Disable Anonymous Access – All users must have an account
2. Enable Logging (for audit purposes)
3. Harden the ACLS (permissions) in the FTP directories – No Everyone account
4. Setup your FTP site as a Blind Put – Only if the server is used solely to transfer files to the FTP site
5. Enable Disk Quotas
6. Use Logon Time Restrictions
7. Restrict Access by IP
8. Audit Logon Events
9. Enable Strong Password Requirement
10. Enable Account Lockout and Account Lockout Threshold

Acme has implemented all the above steps on the current FTP server except steps number 4 and 7, as the server is used to pickup and drop off files by many different users from behind NATed IP addresses.

However at the end of the day the major vulnerability of the transmittal of un-encrypted data, usernames and passwords still exists.

[5] Zadjmool, R. 10 Steps to a Secure FTP Server.
http://www.windowsecurity.com/articles/Secure_FTP_Server.html

6.2 FTP over IPSEC VPN

The concept behind this method of FTP delivery is very simple. It requires all external (and internal users if required) to use an encrypted (IPSEC) VPN to access the FTP server. This ensures that all the FTP session data (usernames, passwords, and data and control commands) are encrypted and therefore cannot be “sniffed”.

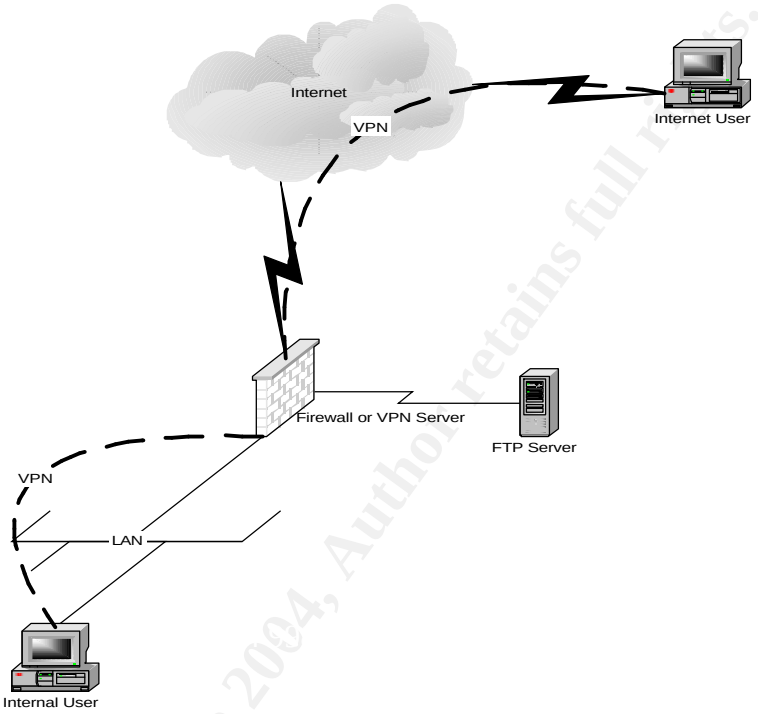

Figure 1 FTP over Encrypted VPN

In this setup the FTP server is deployed behind an IPSEC VPN capable device (Firewall), and any user requiring access to the FTP server would first off all be required to initiate an encrypted VPN. This alone would involve authenticating the end user in addition to setting up an encrypted tunnel to communicate with the FTP server. PPTP could also be used in place of IPSEC.

Normal FTP security should still be followed as defined in Section 5.1, but this means of securing the FTP server is extremely secure.

However, the following are a number of disadvantages associated with this method.

1. It requires the implementation of a VPN infrastructure if one does not already exist that could be made to fit. This could be either in the guise of a software based system or a hardware based system. It might even be possible to combine the FTP server and VPN Server on one box, but that is beyond the scope of this document.
2. The deployment and management of this VPN system would add considerable overhead to the financial and resource costs of the supporting the FTP server.
3. In addition to the above disadvantages, the other major disadvantage is that each client trying to access the FTP server via an encrypted VPN would have to have VPN client software installed on there PC. Distributing this to a widely deployed user base consisting of users from a multitude of companies is simply not practical, and considering the large number of external users requiring access to the file transfer system, this option has also been ruled out as a means of providing Acme with secure file transfer system.

6.3 FTP over SSH (SFTP)

A vast majority of SFTP products are commercial products and similar to the FTP over VPN method previously discussed, transport FTP data over SSH with authentication either being provided by standard username and password or by PKI authentication.

Normally Public Key Infrastructure is associated with the use of private and public keys to authenticate and to provide proof of trust before initiation of encrypted session. A great article on the basic fundamentals of PKI can be found on the Articsoft website.

http://www.articsoft.com/wp_pki_intro.htm

HTTPS is most probably the best know means of deploying an encrypted PKI infrastructure, but the HTTP protocol is fairly limiting in that it has very limited file management capabilities so unless you just want users to have the ability to pickup files the use of HTTP to create a secure file transfer system should normally be discounted. However there are some ways of adapting HTTP to provide a better and more secure means of file transfer. These are described later in the document.

Secure Shell (SSH) is a TCP/IP protocol that provides authentication, encryption and data integrity to secure network communications. It first appeared in the IT world in 1995 and was designed as a replacement for the insecure r-commands found in UNIX. Further updates have been introduced leading up to the

introduction of SSH2[6] in 1997. Since then SSH has been migrated onto many different platforms including Microsoft Windows. It is now used as a secure alternative too many different legacy commands, but mainly as a secure alternative to Telnet and the UNIX\Linux r-command.

A great article on SSH in depth can be found on the VanDyke website.

http://www.vandyke.com/solutions/ssh_overview/ssh_overview_introduction.html

SFTP is a very simple concept. You install the SSH server software on your FTP server and the SFTP client on your client PC. The client then connects to the FTP server via an SSH session. The FTP protocol is then layered over the SSH protocol.

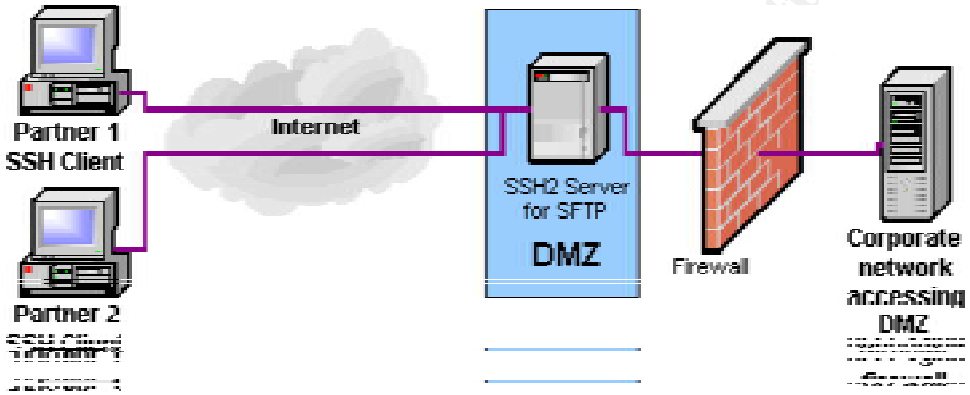

Figure 2 SFTP – Taken from the VanDyke web site [7]

The FTP server can be located on the same server as the SSH server, or it can reside on a separate server. The main difference being that in a single server environment all communication is encrypted, where as in a segregated server environment the data communication between the SSH Server and the FTP Server is un-encrypted.

By using SFTP you are providing your FTP client and server with the following security functionality;

1. User Authentication – All user credentials (usernames and passwords) are transmitted in encrypted format. PKI based authentication and others are also available.
2. Host Authentication – The use of a server host key allows the server to prove its identity to a client. This negates the possibility of vulnerabilities such as the “Man in the middle attack”.

[6] Ylonen, T. and Moffat, D. SSH Transport Layer Protocol. http://www.ietf.org/internet-drafts/draft-ietf-secsh-transport-17.txt

[7] VanDyke. http://www.vandyke.com/solutions/ssh_overview/ssh_overview_filetransfer.html

3. Data Encryption – All data transmitted is encrypted and cannot be unencrypted with out the use of the pre defined shared key. The most common encryption algorithms used are Blowfish, DES, 3DES and more recently AES.
4. Data Integrity – Provides assurance that the data was not interfered with during transmittal.

It is possible to implement your own SFTP solution with out commercial entities being involved by the use of OpenSSH and some other 'free' offerings, but they are very much focused on the UNIX and Linux environments. As with many other open source implementations, the installation and support of these versions of SSH can be temperamental and the use of them in an enterprise environment where commercial interests may be at stake is not recommended.

There are however, a large number of commercial SFTP packages that can be purchased, these packages provide easy to configure installation packages and full support is available. Common commercial packages include;

1. SSH by F-Secure - http://f-secure.com
2. WS-FTP Pro by IPSwitch - http://www.ipswitch.com
3. FTP Voyager by Rhino Software - http://www.ftpvoyager.com/
4. SecureFX by VanDyke Software - http://www.vandyke.com

All have subtle differences and most have evaluation versions available.

However as with the FTP over VPN solution, there are still some drawbacks most of which are common with FTP over VPN, with the main one being that a specific client is required to access the SFTP server.

Whilst many of the clients associated with the above products also allow normal FTP, because of the extra time and effort required for installation and support, this version of FTP was also discounted as means of providing a secure file transfer mechanism for ACME.

6.4 HTTPS and WebDAV

As previously mentioned, HTTP (or in our case HTTPS) on its own, is not a capable enough protocol to allow file management to be conducted. HTTP provides a means for users to pick up data files (read content) from HTTP enabled servers, but it provides no means for users to upload new documents and carry out other file management functions such as uploading files, deleting file directory or creating new ones.

WebDAV (Web-based Distributed Authoring and Versioning) or DAV for short provides additional file management functionality to the HTTP1.1 protocol and is common to the two main web servers, Microsoft IIS and Apache.

6.4.1 HTTPS

HTTP (Hypertext Transfer Protocol) is a set of rules used to transfer data files over the internet and has been around since1990. As a protocol it runs on top of the TCP/IP suite of protocols. A web browser such as Netscape, Opera or Microsoft Internet Explorer would be used to access an HTTP based server. It was further developed as HTTP1.1 which introduced compression and persistent connections. This was introduced in RFC2068[8] in January 1997 and was further developed under RFC2616[9] in June 1999.

HTTPS (Hypertext Transfer Protocol over Secure Sockets Layer or HTTP over SSL) uses the Secure Socket Layer (SSL) to provide a secure means of transporting data over the internet via HTTP. It should not be confused with S-HTTP which has been proposed as a standard.

The SSL protocol was developed by Netscape and uses Digital Certificates to provide data encryption, server authentication, and data integrity and client authentication. SSL comes in two versions, 40bit and 128bit which refer to the length of the session key which is generated during each encrypted transaction. The longer session key, the more difficult it is too 'crack'. Today most modern browsers support 128bit SSL. For a more in depth description of how SSL works go to the page on Netscape.

http://developer.netscape.com/tech/security/ssl/howitworks.html

[8] RFC 2068, http://www.faqs.org/rfcs/rfc2068.html

[9] RFC2616, http://www.faqs.org/rfcs/rfc2616.html

6.4.2 WebDAV

As has previously been discussed, HTTP lets users read data that has already been published on HTTP enabled servers. But, it has no means of allowing users to post new data or edit existing data. WebDAV standardises all this missing functionality making HTTP enabled servers writable as well as readable.

According to Russell Kay in his article WebDAV[10], WebDAV is defined as "a standardised platform-independent set of extensions to HTTP1.1 that allows users to collaborate over the internet by editing and managing files remotely".

WebDAV or DAV for short, first appeared in 1998 and was formalised in February 1999 under RFC 2518[11]. Today it is used by many collaboration tools such as Documentum, Microsoft Sharepoint and Outlook Web Access, these all use WebDAV to provide there collaboration functionality. Some of the new functions added to HTTP by WebDAV are Delete, MkCol (similar to mkdir) and the copy and move functions. It is installed by default in many operating systems such as Microsoft Windows (since Windows 95), Linux, MacOS and Netware. This means that most users have access to its functionality.

By combing HTTP with SSL and WebDAV, it is possible to provide a secure means of File Transfer.

7. The Deployed Solution

To provide a secure, supportable and cost effective replacement for the existing FTP server, ACME took the decision to deploy a Microsoft Windows 2000 server running IIS5.0 with HTTPS and WebDAV enabled.

Whilst a Microsoft based system would not at first seem to be as secure a solution as one based on Linux or UNIX, most of the companies support personnel and procedures revolve around Microsoft products, it was therefore felt that supporting a non Microsoft based system was not ideal.

The company has a standard server hardware production range based on the Dell Poweredge Server and also has a standard Microsoft Windows Server 2000 build document. The build document specifies in detail the steps needed to be taken to successfully build a Windows 2000 server. Further procedures also exist for taking the standard Windows 2000 server and turning this into a web server, which after all is basically what the file transfer server has become. This build

[10] Kay, R. WebDAV. http://www.computerworld.com/softwaretopics/software/groupware/story/0,10801,86688,00.html?nas=QS-86688

[11] RFC 2518. http://www.faqs.org/rfcs/rfc2518.html

document also includes the installation of Norton Anti-Virus for virus protection and St. Bernard Software's UpdateExpert for Microsoft patch deployment.

The server was to be known as ea-fileshare.acme.com, and with the installation of a 128bit SSL certificate would only be accessible via https://fileshare.acme-ea.com.

As the Fileshare server is using a standard Acme Windows 2000 server build, it will be supported and monitored using Acme's standard set of support procedures.

7.1 Server Platform

A Dell Poweredge 2650 server was purchased with the following peripherals.

- Dual Xeon 2Ghz Processor
- Dual PSU
- 1GB SDRAM
- 3x 36GB Disks (2x Raid 1 & 1 hot spare) – Operating System
- 1x PowerVault disk expansion cabinet
- 5x 146GB disks (4x Raid 5 & 1 hot spare) – 438GB storage area

As all data stored would be transient data, it was felt that there would be no requirement for data back up and a ghost image of the OS partition would suffice.

This spec also gave plenty of redundancy, and as it was a server spec commonly deployed within the company, there is plenty of spare hardware available in the event of failure. The possible addition of extra Powervault disk expansion cabinets also allows for any future increase in the size of the date store.

7.2 Operating System

Whilst some consideration was given to using Windows 2003 and IIS6.0, as this has not been standardised throughout the company and no build document or operating procedures exist, it was felt that it was best if Windows 2000 with IIS5.0 be used as the OS.

ACME operates two Active Directory domains; Acme and Acmexnet. Acme is used solely for internet company resources, and Acmexnet is used solely for external, non trusted resources. This means that all servers accessed by external third party companies must lie within the Acmexnet domain, like wise all the user accounts. A one-way trust exists between the two domains that states that Acmexnet trusts Acme. This means that users in Acme can access Acmexnet resources (given the correct file and directory permissions are in place), but Acmexnet resources cannot access anything in the Acme domain. In the unlikely

event of the Acmexnet becoming compromised, there is another logical layer of security between the two domains.

Whilst it was not practical to include the Windows 2000 build document, the IIS and WebDAV build document created during this project is shown below.

Configuring WebDAV Servers for xnet.acme.com (DRAFT)

I. **Install Windows 2000 Server and add to xnet.acme.com.**
- Install standard Acme SP4 image.
- Install SAV.
- Do not install IIS.
- Apply all security patches via UpdateExpert.
- Remove unnecessary ODBC connections.
- Secure TCP\IP stack. In the registry editor, set **HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters *InterfaceGUID*\PerformRouterDiscovery** to **0**. If the reg value (DWORD) doesn't exist, create it.
- Disable the tftp service by editing the file **%SystemRoot%\system32\drivers\etc\services** and place a # (pound sign) in front of the 'tftp' service.
- Restrict the access to Local Security Authority. In the registry editor, set **HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\LSA\ RestrictAnonymous** to 1. If the reg value (DWORD) doesn't exist, create it.

II. **Install IIS.** Do NOT install the following IIS components:
- FTP
- MS FrontPage 2000 Server Extensions
- Internet Service Manager
- NNTP Service
- SMTP Service
- Visual InterDev RAD Remote Deployment Support

III. **Run IIS Lockdown.**
- Logon to server.
- Download iislockd.exe locally and run.
- For server templates, select **SharePoint Portal Server** and check **View Template Settings**.
- Install all defaults except for:
  - Select **Remove unselected services**
  - Click **Yes** in response to the warning message box.

IV. **Install further security on the server's Default Web Site.**
- Disable the IUSR account.
- Uncheck **Anonymous Access** from **Properties• Directory Security• Authentication Methods**.
- Check **Basic** and nothing else.
- Under **Basic**, click **Edit** and enter **xnet.acme.com**.

V. **Reconfigure URLSCAN**
- Notepad **%SystemRoot%\system32\inetsrv\urlscan\urlscan.ini**

- Edit **urlscan.ini** to deny or allow the types of files that can be placed in the Fileshare. For example, by default, URLSCAN will disallow the following file extensions:

```
[DenyExtensions]
; Deny executables that could run on the server
.exe
.bat
.cmd
.com
```

To allow https transfer of these files, remove these extensions from the [DenyExtensions] header from **urlscan.ini**. To disallow https transfer of media files place the following lines under the [DenyExtensions] header:

```
[DenyExtensions]
.aif
.aifc
.aiff
.asf
.asx
.au
.avi
.cda
.IVF
.m1v
.m3u
.mid
.midi
.mp2
.mp2v
.mp3
.mpa
.mpe
.mpeg
.mpg
.mpv2
.rmi
.snd
.wav
.wax
.wm
.wma
.wmd
.wmp
.wms
.wmv
.wmx
.wmz
.wvx
```

- Stop and start IIS to effect the **urlscan.ini** changes.

VI. Place server in the appropriate server OU in xnet.acme.com. Move the server from **xnet.acme.com/Computers** to **xnet.acme.com/Servers/WebDAV/*CompanyName*** to harden the servers via GPO policies. For more information on what these policies do, see the **Security Operations Guide for Windows 2000 Server**.

VII. Create the internal data share.

- Create the data directory as *DataDrive*:*CompanyName*. For example, **D:\AcmeDATA**.
- ACL the directory as appropriate.
  - ACL the directory with *ServerName***Administrators** full control.
  - For internal users, ACL the directory with:
    - The **int.acme.com** group that needs access to this directory
    - The appropriate access rights for that group.
  - For external users:

        - If a group hasn't been created, create a global group using the convention *CompanyName Resource Access*. For example, **Acme FileShare Full**.
        - If a user hasn't been created, create a user using the convention *CompanyNameNum*. For example, **Acme00001**.
        - Affiliate the user to the group.
        - ACL the directory giving the *CompanyName Resource Access* with the appropriate rights.
    - Share the directory using NTFS Sharing.
        - Name the share. For example, **\\Webserver\AcmeData$**
        - Permission the share with **Full Control** for **Everyone**.

VIII. Create the external web site virtual directory.

- Name the virtual directory same as the NTFS share without the $. For example, **AcmeData**.
- Enter the path. In this example, it is **D:\AcmeDATA**.
- Check only **Read, Write** and **Browse** for the access permissions.
- From **Properties• Directory Security• Authentication Methods**, make sure that configuration is the same as the default website (see Section IV of this document).

IX. Install Certificate. After installing a certificate, make sure that:

- Under **Directory Security• Secure Communications**, check **Require Secure Channel (SSL)**
- Check **Ignore Client Certificates**.

X. Make the server accessible over the Internet.

XI. Test.

- For internal users, map to *ServerName**ShareName* and perform file transfers.
- For external users:
    - From the browser, click **File• Open**
    - Enter **https:**//*ServerInternetURL***/***VirtualDirectoryName*. For example, **https://fileshare.acme.com/ACMEdata**.
    - Check **Open as Web Folder**.
    - Click **OK**.
    - Perform file transfers.

.

XII. Submit to Enterprise Operations for final security audit. The audit should rename the administrator account and update the administrator password.

XIII. References.

- Microsoft Knowledge Base Article – 323470: How to Create a Secure WebDAV Publishing Directory
- Security Operations Guide for Windows® 2000 Server
- Microsoft's Improving Web Application Security
- Security Guide for Hardening Windows 2000 Application Servers (including IIS 5.0)

Various other documents exist for the management of the Acmexnet domain, but they fall outside the scope of this document.

In addition to the above OS installation, it was also decided to install Veritas's StorageCentralSRM product.

http://www.veritas.com/products/category/ProductDetail.jhtml?productId=StorageCentralSRM

This gave us two basic functionalities, storage quota and content management. This allows the setting of storage limits per project (directory) and allows easy control of the types of files that can be stored on the fileshare server.

7.3 Policies

The policy document below was developed for use with the new Fileshare server

Acme Fileshare Policies
The Acme Fileshare sites are internet resources that our customers and suppliers can use to share files with our internal users. Only authorized external users can access these resources and access is limited only to the files allowed for access by Acme.

Authorized External Users
All authorized external users will be granted access via a logonid and password. Anonymous access will not be allowed. Requests for access to the Fileshare resources should be forwarded to your Acme project representative and must be authorized by Acme. Requests for access should include:
User's first name.
User's last name.
User's employer.
User's telephone number.
User's e-mail address.
User's street address.
ACME project or department sponsoring the user.
Customer or supplier status.
Resource that user needs access to.
Logonids not used within 90 days will be expired.

Fileshare File and Directory Security
Authorized external users will have file and directory security rights as deemed appropriate by Acme:
Read-only rights will allow copying of files from the Fileshare.
Read-Write rights will allow users to copy files into the Fileshare.

Virus Scanning
All files transferred to the Fileshare will scanned for viruses. If the virus scan finds a virus in a file, the file will be removed from the site.

Transferable File Types
The Fileshare will allow transfer of all file types except for the following:

Illegal, Offensive, or Inappropriate Content
Files that contain illegal, offensive, or inappropriate content will result not only in the removal of the files but also revocation of access to Fileshare resources.

It has also been agreed that all data more than 30 days old will be removed automatically.

Permissions are assigned to the Fileshare, as they would be to any other normal file and print directory.

Clients wishing to connect to the Fileshare server must do so using a DAV enabled web browser. Any modern web browser should be capable of this. The first time it is accessed by this method, the user would have to open the web page as a web folder (in effect enables WebDAV) and bookmark the page. Once this has been done, using the bookmark will mean that the user will no longer have to open the page as a web folder as this will be done automatically.

Internal users would be able to access the Fileshare folder either via the above method or more likely via a standard mapped network drive.

8. Conclusion

The Fileshare server has been in full production use for 2 months and is functioning much as expected. It is now providing Acme with a secure means of transferring data to and from third party external companies based on tried and tested technology and standards.

By using HTTP in conjunction with SSL and WebDAV Acme was able to deploy a secure file transfer system that authenticated the users using an encrypted username and password, ensured the identity of the file transfer server, encrypted the data during transmission and ensured the integrity of the data.

Its major limitation at the moment is an inability to carry out automated batch file transfers; although it is thought that if the requirement for this functionality becomes a reality, that some simple PERL scripting would provide the automated functionality.

9. References

Postel, J. and Reynolds, J. “File Transfer Protocol”, RFC 959. October 1995. http://www.faqs.org/rfcs/rfc959.html

Bushan, A. “File Transfer Protocol”, RFC 114. 16 April 1971. http://www.faqs.org/rfcs/rfc114.html

Postel, J. “File Transfer Protocol Specification”, RFC 765, June 1980. http://www.faqs.org/rfcs/rfc765.html

Cert Coordination Centre. “Vulnerabilities, Incidents and Fixes”, December 2003. http://www.cert.org/nav/index_red.html

Zadmjool, R. “10 Steps to a Secure FTP Server”, 08 July 2003. http://www.windowsecurity.com/articles/Secure_FTP_Server.html

Ylonen, T. and Moffet, D. “SSH Transport Layer Protocol”, October 2003. http://www.ietf.org/internet-drafts/draft-ietf-secsh-transport-17.txt

VanDyke Software. “Functionality of Secure Shell – Secure File Transfer”. http://www.ietf.org/internet-drafts/draft-ietf-secsh-transport-17.txt

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., and Berners-Lee, T. “Hypertext Transfer Protocol – HTTP/1.1”, RFC 2068, January 1997. http://www.ietf.org/internet-drafts/draft-ietf-secsh-transport-17.txt

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P. and Berners-Lee, T. “Hypertext Transfer Protocol – HTTP/1.1”, RFC 2616, June 1999. http://www.faqs.org/rfcs/rfc2616.html

Kay, R. “WebDAV”, 03 November 2003. http://www.computerworld.com/softwaretopics/software/groupware/story/0,10801,86688,00.html?nas=QS-86688

Goland, Y., Whitehead, E., Faizi, A., Carter, S., Jensen, D. “HTTP Extensions for Distributed Authoring – WEBDAV”, RFC 2518, February 1999. http://www.faqs.org/rfcs/rfc2518.html

Useful Websites

Cert Coordination Centre - http://www.cert.org/nav/index_red.html

ArticSoft - http://www.articsoft.com/wp_pki_intro.htm

VanDyke Software - http://www.vandyke.com/solutions/ssh_overview/ssh_overview_introduction.html

National Security Agency - http://www.nsa.gov/snac/index.html