Global Information Assurance Certification Paper

SANS GSEC Practical Assignment version 1.3

FTP and other goodies: What are your users pushing today?
Michael Bell
February 23, 2002

Background

For many in the field of computer network security, emphasis has been placed almost entirely upon filtering inbound traffic to prevent unwanted intrusion(s). How many of us have actually taken a serious look into the type(s) of traffic outbound to the Internet through our firewalls? Of those who have, how many of us perform this on a regular basis? Occasionally we may look into items like bandwidth utilization, appropriate use, etc., but are we really helping our users without understanding the business functions they perform using the Internet? Has a recently changed policy or procedures had unanticipated downstream affects on how our users conduct business? (I'll provide one example of this later in the discussion.) We mainly worry from day to day about hackers, DoS attacks, port scans and other inbound items but do we really know what our users are doing ***outbound***? While much has been said about responsible use of the Internet, it is probably a stretch to rely on the average user to worry about things like confidentiality, availability, and integrity while performing their daily duties. One item worth looking into is the use of file transfer protocol (FTP). More importantly, what are your users doing with FTP and what is being done with their data once it passes through the firewall outbound?

Scenario

While FTP has been around for many years [1], it wasn't for use if you were worried about the confidentiality and integrity of the data. Most users of FTP were always interested in passing the data in the most efficient manner to the intended recipient. Our measure of data integrity was usually quantified by successful delivery of the entire file without having to resend. (Mainframe users may remember the IND$ method of transferring files and wondering if the whole file arrived or just a fraction of the original.) Why worry about how it got there, as long as it got there and was useable, right? That worked great over leased lines and with transfers amongst trusted relationships. Besides, FTP never claimed to be a secure method of data transport, just reliable. As our Internet acceptance grew we then found a less expensive way to move the data without the dedicated leased lines. Is your data confidential in nature? That's the first question asked of those who wanted an FTP server installed. As long as confidentiality was not an issue, the only stipulation was just make sure the very large file transfers are performed at night (or whatever off peak Internet utilization hours were appropriate) and thereby help save the corporate Internet bandwidth during the business day.

As concern grew regarding Internet threats and vulnerabilities, firewalls were modified so we could continue to receive data via FTP without fear of being exposed to other threats coming inside. For those of us who were responsible for the firewall configuration, we were taught early how many ports we had to open if we wanted to allow FTP into our network. FTP may start the connection using a high source port connection to TCP port 21, but the remaining packets in the data transfer used randomly generated high ports numbers. There have been attempts to write a 'firewall friendly FTP' [2] but this suggestion apparently never caught on or wasn't feasible for other reasons. Then there was the dreaded FTP 'bounce attack' which instilled more fear in an already difficult situation [3] for those of us with FTP servers to look after. Good thing we created that second DMZ area so that we could allow FTP inbound without subjecting our other web devices to additional risk. Although we opened up a huge hole (with regard to TCP port numbers) in the firewall to allow transfers into our network, at least we were still protected on the 'inside' (intranet). We then focused on creating secure 'incoming' and 'outgoing' directories to allow data to be put onto our FTP server without fear of any new virus or worm getting into the 'inside'. With a quick disable of anonymous login things were working properly, everyone was authenticated and could only see 'their' directory. Addition of a great virus scanner to check the files before allowing the data to be brought 'inside' and we had no more FTP worries. Quite a bit of work to get to this point but our users benefited from the new server and its use quickly became routine. Later, when it was decided to receive confidential data a second FTP server was created in the intranet and users of this device were required to authenticate through a VPN switch. Thus we had encryption for the contents of the FTP transfers.

All of this to point out the pains that were endured to secure and maintain ***inbound*** FTP traffic.

With the FTP server secure it was very easy to forget about FTP altogether. Outbound FTP was not a daily focus, if we didn't manage the server then how could we be held accountable for the method or manner in which data transfers were conducted? Besides, most of our mainstream users weren't comfortable using FTP and wouldn't want to learn it anyway. (That's probably why web browsers can utilize FTP to transfer files without the end user even realizing it). They already knew that FTP was not secure, it didn't encrypt, and that the usernames/passwords were sent as clear text [4]. We all learned those facts years ago [5], first day of FTP class, right? Besides, most of our users didn't remember how to get to a DOS prompt, didn't know what FTP stood for, and wouldn't be able to connect to the server without some serious coaching. Somewhere in the background to our work the user community was innocently downloading all of these great new FTP utilities (via HTTP!) while we ignored their daily activities. Data files concurrently grew larger and larger in size as technology (hard drives, memory, software) expanded the role of the business desktop PC. These were graphical (GUI) programs that looked just like the Windows "My Documents" folder in their desktop and helped the users understand how to FTP with a drag-and-drop methodology. When they had a large file to move, they simply used their new

FTP software and off it went. Soon everyone was sending files via FTP as a matter of their business day.

"Quick, FTP that file over to Accounting so that they can see for themselves how much the budget has changed this month!" I overheard while down in the Engineering department recently.

As I kept walking I thought to myself "I didn't know Accounting had set up an FTP server, wonder who did the work?" Hope they secured it as well as we did ours. Wait a minute: Did he mean to FTP the corporate budget to Accounting? Did I just hear that correctly? A quick walk back and a glance over the administrative assistant's shoulder verified my concern - ***confidential corporate data just went out over the Internet via FTP to our Accounting office in Anytown, USA!*** Please tell me that this particular administrative assistant is part of the VPN tunnel connected to the Accounting office, right? You guessed it, no VPN, no encryption. How long has this practice been going on? She must be new here, but wasn't that her supervisor that instructed her to send the data?

After getting back to my desktop I confirmed the old policy on FTP still existed and was posted for all to see (it had been quite a number of years since this topic was discussed). A quick call to management for permission and I fired up one of the packet sniffers to see FTP packets merrily going out to destinations unknown. Quickly we decided to implement a new filter to stop outbound FTP completely. We planned later to discreetly make the supervisor in question aware that unless they were using one of the FTP clients that supported encryption or a VPN tunnel, this was not a recommended way to move confidential data. With the outbound traffic filter done, just like my Internet radio filter (the one that was instantly rendered useless by the application when it immediately switched from a specified TCP port to TCP port 80 for it's connection), we were satisfied that another crisis had been averted. We rationalized that the Engineering department probably had just started this practice to get around the recent change to the Email policy with regards to the maximum attachment file size. After all, those Engineering drawings and other related documents could be very large files to send.

Then the phone calls started coming in to the help desk. First was the Payroll department on the phone, reporting problems with the network. Taking the call quickly (have to keep this department happy!) we found the complaint to be that the weekly transfer of CONFIDENTIAL payroll data to our payroll service contractor had suddenly died in the middle and could not be restarted for some unknown reason. Yes, outbound FTP without encryption again.

Discussion

Humorous anecdote or sad commentary, either way it brings to light several important issues. How could this have been overlooked? First of all, some old lessons needed to be relearned or at least revisited. Secondly, new access and new tools made what was

once a bastion of only the technically capable an easy to use everyday tool. Our Internet firewall wasn't blocking outbound ports or traffic types, we relied on the end user to comply with the published Internet usage guidelines. A review of past network audits revealed some outbound FTP, but no statistics were given and we did not have any data regarding the amount of confidential data that was outbound. Sure FTP had been part of the IP stack software on the desktops of our users for quite some time but it was not easy to use and certainly hadn't been point and click capable. The end user did not feel 'comfortable' with FTP. But someone forgot to tell the user community they weren't capable and keep them in the loop with regards to data confidentiality and integrity. The act of transferring the data was the only user function with no apparent knowledge about the potential for compromise. If a user could quickly send out payroll information that contained social security numbers, addresses, pay rates, etc. on the Internet what would be the liability if this data were intercepted, manipulated, or (gasp!) published? How could anyone recover from that level of disclosure?

In defense of the end user, if you are instructed to transmit data to an FTP server, how could you know that it wasn't via the corporate VPN or that the product you were using was capable of encrypting the data but wasn't configured to implement this enhancement? The end user has little knowledge of such information available to them, just their desire to comply with given policies and to protect valuable corporate assets while getting their respective job duties completed. If no one takes the time to explain, if they are never instructed on how to encrypt or why, simply told that the policy is to encrypt, then all sorts of scenarios become possible.

While the thought of intercepting and manipulating payroll records is the stuff Hollywood movies and good daydreams are made of, we could hardly run the risk of our sensitive corporate data being put out onto the Internet for all to see or use without authorization. All sorts of competitive advantages could be gained. Our current Internet use policy did not restrict outbound FTP in any way with the exception of content. It basically stated that confidential or sensitive information should not be transmitted via the Internet unless encrypted or otherwise protected against compromise. We forgot to watch our outbound FTP, too focused on newly rumored Instant Messenger application vulnerability or the latest signature file for our intrusion detection system. We didn't keep our policy in the forefront and most importantly, we didn't think about content and it's sometimes special 'handling' needs. What if this had been a confidential medical record or the secret recipe to the company's fried chicken? Whether dealing with HIPAA (the Healthcare Insurance Portability and Availability Act, a.k.a. Public Law 104-191) compliance [6] or simple departmental policy, our use of FTP had become a wide-open faucet of confidential data being sent without any protection in some cases. We had done a great job (we thought) on handling the integrity of our inbound FTP transfers, we had spent agonizing months dealing with the confidentiality of email attachments, but nothing concerning outbound FTP. Our periodic audits of network traffic had revealed the presence of outbound FTP traffic but little attention was paid to the destination or content. We missed the boat entirely on this one. What other boats had we missed as well?

Our next course of action was to get our arms around the size of the issue at hand. The practice of using desktop FTP applications obviously had momentum already, the phone was ringing off of the hook with the outbound FTP blocking filter in place. Our naive reliance on the owners of other FTP servers to properly secure our confidential data had left us in a compromising situation. We needed to brief management quickly so that we could get some clear, concise steps on the table and their quick approval. Some hasty research revealed that some FTP clients that were in use currently had the capability to encrypt data and to protect username/password accounts. Although this capability was in place, it was not being utilized.

Encryption was deemed the appropriate solution, as it was the most common means used to protect data against theft or eavesdropping. A policy regarding encryption standards was already in place and current so we didn't have to go back and rewrite anything, just address the immediate need and the correct application of the appropriate policy(s). It was agreed we had some catching up to do. With management approval, we implemented the following immediate action steps in our effort to mitigate the situation:

- Immediate blocking of all outbound TCP port 21 packets.
- Notification of new outbound FTP restrictions to individual department heads.
- Summation of FTP transfer needs based upon content and user location.
- Contact the FTP server owners' to determine their software packaging capabilities with regards to encryption of data and username/password combinations.
- Evaluation of several desktop/server FTP products in our laboratory environment to verify that indeed encryption was meeting departmental needs.
- Identification of specific users who would need FTP access and restriction of all outbound FTP to unauthorized users.

Encryption was what we needed, and we wanted the best we could get. Not only in capability but also in terms of ease of use, speed, and effectiveness as well as longevity. We found that not all FTP software packages had encryption algorithms that met our needs. Some seemed on the surface to meet the requirements but further reading revealed potential flaws in the encryption algorithms selected. [7] One of our options was the use of 128-bit Secure Socket Layer (SSL) encryption if the companion server software was utilized [8]. What is SSL? According to Netscape, maker of the Navigator web browser product:

> "SSL (Secure Sockets Layer) is a communication system that ensures privacy when communicating with other SSL-enabled products. Technically speaking, SSL is a protocol that runs above TCP/IP and below HTTP or other top-level protocols. It is symmetric encryption nested within public-key encryption, authenticated through the use of certificates. An SSL connection can only occur

between an SSL-enabled client and an SSL-enabled server. In fact, when a server is running in SSL mode, it can only communicate through SSL" [9]

SSL wasn't new technology, was readily accepted by most users and servers, and could afford a higher level of security than DES if used in the 128-bit mode. Secure shell (SSH) seemed easiest to implement due to its use for other web services and availability of Window's client [10]. We also had several unique situations including an FTP server running Linux that had no administrator we could contact. We were able to convince management to allow us to help the owners secure this server using SSH and techniques we had discovered in our investigations [11].

The most promising commercial FTP client piece boasted secure connections ***to any*** FTP server using our choice of Rijndael or Twofish encryption algorithms. The Rijndael (pronounced Rain Doll) algorithm had recently (November 26, 2001) been selected as the newly proposed Advanced Encryption Standard (AES) by the National Institute of Standards and Technology (NIST). [12] For those new to encryption, AES is scheduled to replace DES as the encryption algorithm standard. See quote from the NIST AES selection announcement below:

> "The AES has been developed to replace DES, but NIST anticipates that Triple DES will remain an approved algorithm (for U.S. Government use) for the foreseeable future. Single DES is being phased out of use, and is currently permitted in legacy systems, only.
>
> Triple DES and DES are specified in FIPS 46-3, while the AES is specified in FIPS 197. The status of the algorithms in each FIPS is handled separately by NIST." [13]

It was determined that with the recent announcement of the new AES standard it would be more productive to implement the new AES capable client FTP software now rather than an older technology which may have to be upgraded in the near future. DES encryption had been broken years ago and was losing favor quickly although widely deployed. [14] Triple DES, or 3DES, was more effective and could use a theoretical key length of 168 bits (3*56 bits) but it has been reportedly slow for some applications and it was clear the standard was moving, NIST had already announced their new selection for AES. So what would this new AES do for the data or, more appropriately, to the data? Well, take a look at the available key sizes quoted from the NIST Q & A page:

> "The AES specifies three key sizes: 128, 192 and 256 bits. In decimal terms, this means that there are approximately:
>
> 3.4×10^{38} possible 128-bit keys;
>
> 6.2×10^{57} possible 192-bit keys; and

1.1×10^{77} possible 256-bit keys.

In comparison, DES keys are 56 bits long, which means there are approximately 7.2×10^{16} possible DES keys. Thus, there are on the order of 10^{21} times more AES 128-bit keys than DES 56-bit keys." [14]

Given the potential for key size in AES over DES, it appeared that it would offer a more secure encryption capability far into the future. So now we had found the encryption standard we wanted to implement (AES) in a package that we could deploy quickly. The new software piece not only met all of our technical requirements but previous versions were actually in use by some of our FTP users so the upgrade path and training was more readily completed. Through additional lab testing and some field-testing with various users, we were able to standardize everyone who wanted to perform outbound FTP on one client software product to meet all of our needs. This kept the desktop support personnel from having to ramp up to support various platforms. Although not freeware, the new FTP software cost was reasonable (approximately $400.00 US for the client piece) when it's capabilities and the reduction in the number of desktops that would be utilizing FTP client software was factored in to the equation. The resultant list of action items helped us shore up our long forgotten FTP policies:

- Standardization on one client FTP program.
- Use of AES Rijndael algorithm for encryption where appropriate.
- VPN tunneling where possible.
- Policy modification to clarify FTP practices with regard to sensitive data.
- Closer auditing of outbound traffic for type and content.
- Issuance of static IP addresses on those desktops approved to use outbound FTP with encryption.
- Training of FTP users to use new software and its encryption functionality properly.
- Training to increase confidentiality awareness.
- Implicit deny of any unauthorized outbound FTP on firewall with the exception of those approved desktops and their respective IP addresses.

All of this resulted in a more secure environment for transferring data as well as opening up some much needed lines of communication between user departments and those of us in network security.

While we didn't completely have control over outbound FTP with these new measures, (some users could FTP using a web browser and HTTP port 80 but they could also utilize SSL for encryption in that situation) the increased awareness among the user communities fostered a cooperative approach and achieved more understanding of the technical and business issues. By approaching an old problem with new tools, we were able to resolve the issues quickly without disrupting work flow significantly. Management involvement was paramount as existing procedures had already been established within the user community and had momentum. By integrating the solution

with the specific needs of each department, we were able to integrate the new software and procedures with little resistance.

A subsequent audit of outbound traffic revealed that although FTP utilization had increased, the traffic was only originating from the specified desktops that had the new client FTP software installed. (This was deemed to be due to the existence of the outbound FTP filter on the Internet firewall that blocked such outbound traffic to any desktops except those authorized.) Now our current focus is to evaluate the use of instant messenger software (IM) to transfer data. Our audit revealed several desktops utilizing this method of communications and most have built-in file transfer mechanisms that could be sending unencrypted confidential data outbound.

Summary

When analyzing your network security policies it's important to keep outbound traffic content in mind. Make sure that confidentiality of data in both directions is addressed. Review/Audit outbound traffic on a regular basis and carefully analyze the patterns and contents. Simply performing periodic audits is of little use if the results aren't properly analyzed. Periodically revisit those old policies and review them for the sake of new employees or for knocking the dust of off old ideas that are still valid. As users become more capable through either training or the use of new software, maintain awareness regarding old discounted practices that may become more widespread. Remain cognizant of new restrictions that may cause users to seek alternative methods or technologies in meeting their business needs. (In our case, the reduction in maximum file size attachment on email had inadvertently increased the amount of outbound FTP utilization.)

Just as new technology had made it easier for our user community to increase their use of outbound FTP file transfers without encryption, it also enabled us to find a secure method for resolving the issue. Keeping abreast in the latest standards and software products is a worthwhile effort to assist in solving new situations as well as some old ones that may not have been as readily solved in the past. Probably most our most important lesson learned was the practice of keeping lines of communications open with our user communities with regards not only their business practices, but also their business needs as well.

Resources

[1] RFC959 “File Transfer Protocol", STD 1, by J. Postel and J. Reynolds, October 1985
http://www.ietf.org/rfc/rfc959.txt

[2] RFC1579 “Firewall-Friendly FTP”, by S. Bellovin, February 1994
http://www.faqs.org/rfcs/rfc1579.html

[3] “FTP Security Considerations” by M. Allman, May 1999
http://www.ietf.org/rfc/rfc2577.txt

[4] “FTP Server Security” By Mark Gibbs, Network World, 08/20/01
http://www.nwfusion.com/columnists/2001/0820gearhead.html

[5] “FTP and clear text account/password” circa 1989
http://www-mice.cs.ucl.ac.uk/multimedia/misc/tcp_ip/8905.mm.www/0148.html

[6] “HIPAA Compliance: Cost-Effective Solutions for the Technical Security Regulations”
by Tautra Romig, November 21, 2001
http://rr.sans.org/legal/compliance.php

[7] “FTP Server Overview” by Nelson King
http://serverwatch.internet.com/ftpservers.html

[8] WS_FTP Pro software specifications page
http://www.ipswitch.com/Products/WS_FTP/

[9] Understanding Encryption and SSL
http://developer.netscape.com/docs/manuals/proxy/adminux/encrypt.htm

[10] SECURITY AND ENCRYPTION - SSH
http://www.terena.nl/libr/gnrt/security/s6.html

[11] Securing FTP uploads using SSH (A practical guide to securing FTP under Linux) 08/02/02
http://www.securiteam.com/unixfocus/5LP022K0KC.html

[12] Beyond FTP Encryption
http://www.beyondftp.com/encryption.htm

[13] ADVANCED ENCRYPTION STANDARD (AES)
Questions and Answers

http://csrc.nist.gov/encryption/aes/aesfact.html

[14] DES Encryption Overview
http://www.tropsoft.com/strongenc/des.htm

[15] ADVANCED ENCRYPTION STANDARD (AES)
Questions and Answers
http://csrc.nist.gov/encryption/aes/aesfact.html