Realtime publishers

The Shortcut Guide™ To

Eliminating Insecure and Unreliable File Transfer Methods

2012 Edition

sponsored by

Attachmate

Dan Sullivan

Introduction to Realtime Publishers

by Don Jones, Series Editor

For several years now, Realtime has produced dozens and dozens of high-quality books that just happen to be delivered in electronic format—at no cost to you, the reader. We've made this unique publishing model work through the generous support and cooperation of our sponsors, who agree to bear each book's production expenses for the benefit of our readers.

Although we've always offered our publications to you for free, don't think for a moment that quality is anything less than our top priority. My job is to make sure that our books are as good as—and in most cases better than—any printed book that would cost you $40 or more. Our electronic publishing model offers several advantages over printed books: You receive chapters literally as fast as our authors produce them (hence the "realtime" aspect of our model), and we can update chapters to reflect the latest changes in technology.

I want to point out that our books are by no means paid advertisements or white papers. We're an independent publishing company, and an important aspect of my job is to make sure that our authors are free to voice their expertise and opinions without reservation or restriction. We maintain complete editorial control of our publications, and I'm proud that we've produced so many quality books over the past years.

I want to extend an invitation to visit us at http://nexus.realtimepublishers.com, especially if you've received this publication from a friend or colleague. We have a wide variety of additional books on a range of topics, and you're sure to find something that's of interest to you—and it won't cost you a thing. We hope you'll continue to come to Realtime for your educational needs far into the future.

Until then, enjoy.

Don Jones

Copyright Statement

Chapter 1: Understanding the Limits of Traditional File Transfer

If you had to identify the most salient characteristics of enterprise IT operations, you would certainly include the enormous volumes of data in motion. Data moves from point of sales systems to back-office systems, from financial systems into data warehouses, and from data centers to failover and disaster recovery sites. This data moves through IT infrastructure in a number of ways. Tightly coupled systems may use application programming interfaces (APIs) to pass individual, transaction-level data. This is a sound solution when requirements demand rapid movement of data as soon as it is available; for example, when passing credit card transaction data to a risk analysis system that predicts fraudulent activity. In many cases, however, data is more efficiently moved in batches from one application to another. In these cases, file transfer is the tried-and-true method that is widely used.

The Shortcut Guide to Eliminating Insecure and Unreliable File Transfer Methods examines how common practices in file transfers undermine the efficiency and security of business operations. This guide consists of four chapters, each of which addresses a particular aspect of the file transfer problem and offers a way of mitigating those problems.

This chapter starts with an examination of the question: What is it about common file transfer methods that are so problematic? After all, if so many of us have been using these techniques for so long, how bad can they be? The unpleasant answer becomes clear pretty quickly once you start delving into the details. In brief, the shortcomings you can tolerate in one quick-and-dirty file transfer solution can easily find their way into multiple mission-critical workflows undermining the integrity of these broader systems.

Chapter 2 considers business objectives that are jeopardized by the problems highlighted in Chapter 1. These include compliance, flexibility, and scalability of IT infrastructure, management issues, cost control, and workflow efficiency.

By the end of Chapter 2, you should have a clear understanding of the problems with insecure and unreliable file transfer methods as well as the business issues that are driving the need for a better solution. Chapter 3 defines functional requirements that can help guide the selection of a secure, reliable, and efficient file transfer system. This chapter considers the need for supporting information flows, security, file transfer optimization, transaction controls, process reporting, and event processing.

Chapter 4 moves on to issues related to deployment and management. Once a secure and reliable file transfer system is in place, there will be questions about policies, procedures, and service level agreements (SLAs). Chapter 4 walks through a deployment methodology that begins with assessing current file transfer methods and identifying critical business requirements to establishing policies and procedures and rolling out a file transfer solution.

Together, these four chapters will help you understand the limits of traditional file transfer solutions and the risks they pose to businesses. This guide will also provide guidance on selecting, deploying, and maintaining a secure and reliable alternative file transfer solution.

Characteristics of Typical Homegrown File Transfer Solutions

Many IT professionals have had to deal with the problem of transferring files. Software developers routinely work with files, generating output from their applications that will be needed as input to other applications. Systems administrators are constantly working with operating system (OS) and network log files, application output files, backup files, and a seemingly endless list of other types of data files that are needed in today's enterprises. As these examples may indicate, file management is often part of a larger workflow and that, itself, can be a problem.

Perils of Quick-and-Dirty File Transfer Solutions

Let's consider a hypothetical scenario involving file transfer tasks. A sales manager in a company has determined that online sales are underperforming expectations. She thinks the problem may be related to usability issues in the Web site design, so she has asked the business intelligence (BI) analyst in the group to analyze click stream data to better understand the paths customers take through the Web site.

The BI analyst decides to pull log data from the Web server and run it in his favorite statistical analysis package. He'll need to keep his copies of click stream data up to date, so he decides to write a Perl script to copy the file from the server to his desktop. After a few days analyzing the data, the BI analyst develops a program to identify patterns in the click stream data and use them to classify several types of customer interactions or sessions with the Web site. The sales manager finds the analysis is just what she needed and decides to incorporate the results into the sales data mart.

The data mart has evolved over the past few years in the sales department, usually by incremental additions of data. Some source data comes from a file system, some is extracted from other databases, and a couple of other sources are on an application server and another Web server. Adding the click stream data is just another data source, so past patterns are followed. An extraction, transformation, and load (ETL) workflow is created to copy the statistical analysis results from the analyst's workstation to the data mart. (The sales manager knows that process should really run on a production server, but there is no time to do the migration now; all involved agree to get to that as soon as possible.) The result is a workflow depicted in Figure 1.1.

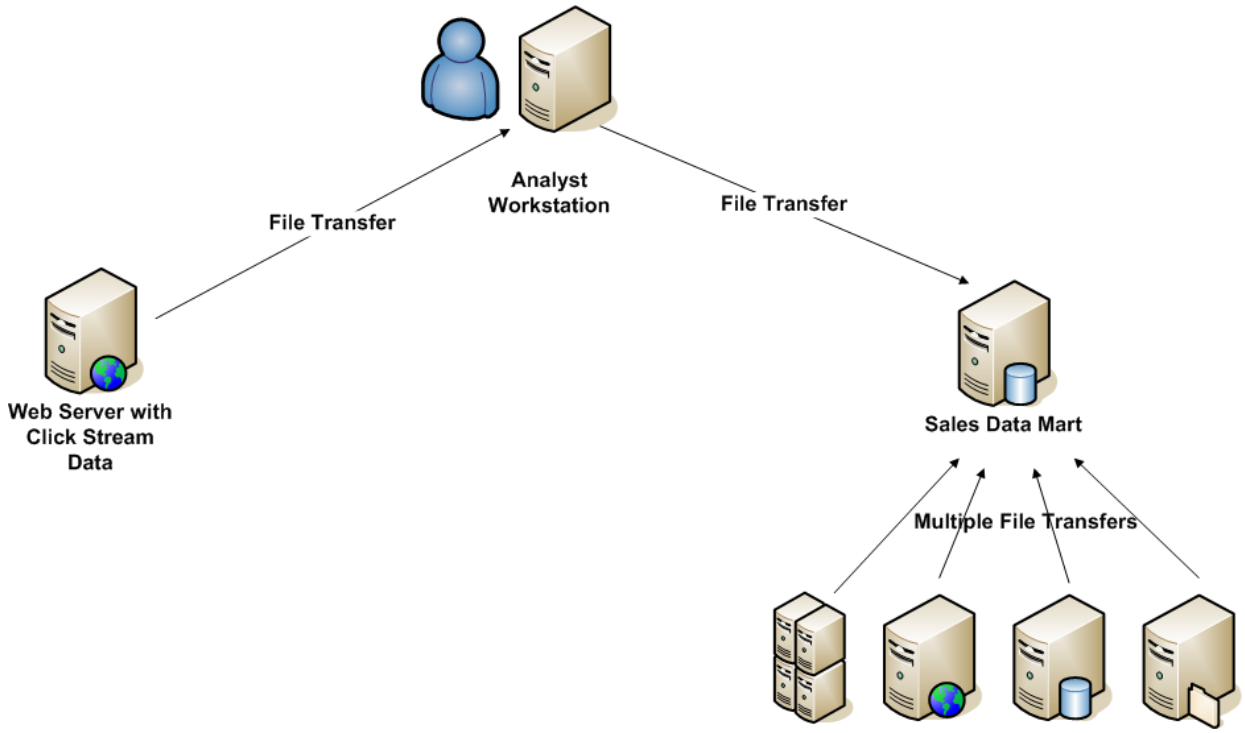

Figure 1.1: Simple, isolated file transfers can quickly become critical parts of larger workflows.

What Figure 1.1 does not show is how the different file transfer processes are implemented. In this scenario, there are several methods:

- The BI analyst is an adept Perl programmer, so the file transfers from the Web server to his workstation and from his workstation to the data mart are performed with Perl scripts he wrote specifically for those operations.
- The developer responsible for transferring files from the file server, which is running a Windows OS, used some Visual Basic code from another application and adapted it to this file transfer problem.
- The files from the database server, which runs Unix, are copied with Unix commands in a Korn-shell script.
- The files from the Web server and application server, which both run Linux, are preprocessed with text processing utilities, awk and sed, before they are copied by Perl scripts to the data mart.

Each time a need arose to transfer files to the data mart, the developer responsible for the task took the most efficient path: writing a custom script to do exactly what was needed using a programming tool that best fit the source platform of the files. As a result, a series of local, optimal decisions leads to a substantially sub-optimal global solution.

File transfers are such a common and fundamental task in IT operations that the task warrants a global solution. To understand what that global solution should include, let's consider in more detail some of the characteristics of homegrown solutions.

5 Detrimental Characteristics of Homegrown File Transfer Solutions

Five characteristics stand out when considering homegrown file transfer solutions; each of these contributes to undermining their reuse and maintenance. These characteristics are:

- Custom scripts
- Focus on a single problem
- Duplicate functionality
- Limited error handling
- Difficult to maintain

These characteristics are closely related and collectively undermine the long-term utility and generalized use of these programs.

Custom Scripts

File transfer is one area where a common need frequently leads to individual solutions. Many of us in development and systems administration roles have written custom scripts for managing file transfers. It is so commonplace, many of us would not think twice about starting up our favorite editor and cranking out the Perl code to get the job done. In the best of cases, we use design patterns and common programming constructs along with copious documentation so that the next person who comes along to maintain the script will not feel like she is deciphering Mayan hieroglyphics.

The biggest problem with custom scripts is the need to keep recreating a solution to a common problem. This is unfortunate. Other areas of software engineering and systems management use common solutions, including OSs, databases, application servers, report generators, and other problem-specific applications. Custom scripts are by definition, not standardized; they are:

- Written in different languages. Perl is probably the most popular programming language used but Python, Ruby, and Unix shells are useful for file transfer operations.
- Written by different developers. Perl's motto is "There is more than one way to do it." Perl is a versatile language and that leads to multiple solutions to common problems, such as file transfer. Design patterns are available for data manipulation in Perl (see, for example, David Cross' *Data Munging with Perl*, Manning Publications, 2001) but when under the gun to deliver a solution, developers understandably turn to methods and techniques they have used and know work.
- Written to solve a single problem rather than abstract the problem and solve the more general problem

The reason for such customization is that these scripts tend to focus on a single problem.

Focus on a Single Problem

A second common characteristic of homegrown solutions is that they tend to focus on a single problem. This focus stems from the localized view of the problem at hand. Developers on a project are charged with completing their development on time and in such a way that reliably meets the project requirements. Similarly, systems administrators are responsible for keeping applications and systems up and running as efficiently as possible. These perspectives do not lend themselves to solving the more global problem of creating a robust, reliable solution for file transfer tasks across the enterprise.

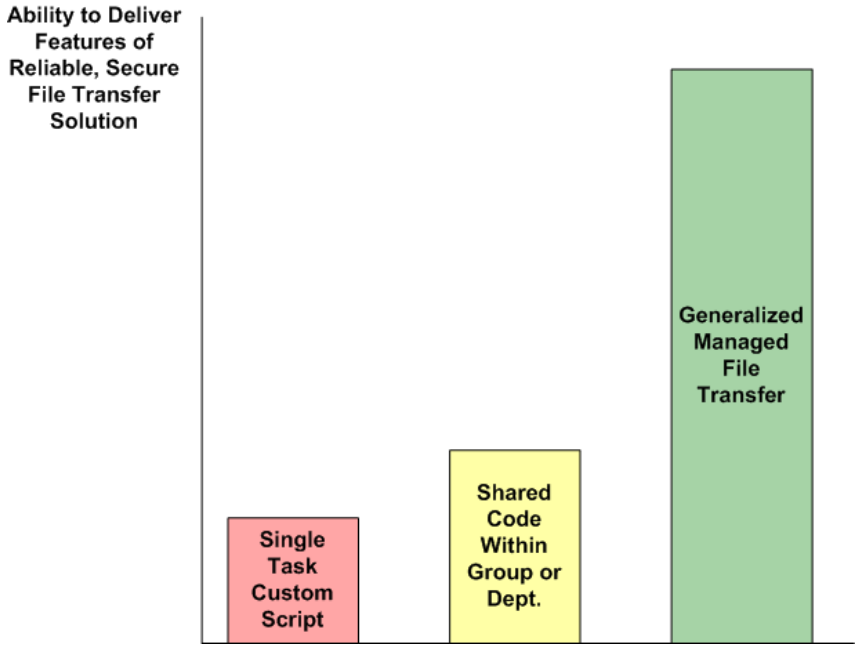

Figure 1.2: Focusing on a single problem limits the range of functionality; taking a more generalized approach for department-level needs is a step toward a general, managed solution.

In some cases, a group or department faces enough file transfer issues that they start to standardize. For example, they may decide to always use Perl for their scripts and they may routinely copy code from old scripts into new ones. These practices lead to common features across scripts, but there are still shortcomings. For example, each script is still a solution to a specific problem and applying that script to a new problem can require significant changes. If someone finds and corrects a bug in one script or adds a new feature to another, those improvements are not available to the other scripts. Sharing a common evolution of code is helpful in some ways, but it is still an insufficient solution for enterprise-scale file transfer needs.

Duplicate Functionality

At the other end of the code sharing spectrum is the opposite problem: duplicating functionality. Given the willingness of developers to share code, post solutions in developer forums, and pass on good resources for new programming ideas, it's safe to assume this is not a group fond of reinventing the wheel. Once again, the pressures to get a solution done and to focus on the bigger picture (usually a project for which file transfer is only a small piece) influences developers to build "just enough" to get the job done.

Why not just go find another piece of code that already solves the problem at hand? The cost of searching out other developers' code, understanding how it works, and adapting it to one's particular needs is often greater than devising your own solution. This is especially the case if the file transfer code includes minimal features and skirts issues such as error handling, verification, and reporting.

There are parallels here to what economists call negative externalities. When a factory emits pollutants into the air, the true costs are not borne by the factory owners; we all share in those costs. When developers write just enough code to satisfy their immediate needs, they risk creating negative externalities for the business at large. Is the code secure? Are passwords stored in plain text in a script? Does the file transfer method meet compliance requirements? Even if you answer 'no' to any of these questions, if the code copies a file from point A to point B, then the code does what it is supposed to do. At least by one set of measures. As you shall see in the next section, unmet business requirements are becoming more prominent criteria in business decision making.

Limited Error Handling

When it comes to error handling, it does not help to be an optimist. In IT, things go wrong on a fairly regular basis. Disks run out of space, programs do not finish running as quickly as we think, and they sometimes return results we don't expect. File transfers are prone to their own set of all-too-common problems:

- Insufficient space to store files
- Insufficient space to decompress files once they are transferred
- Challenges with managing encryption keys for decrypting files
- Protocol-related errors, like dropped packets and lost connections
- Determination of the precise nature of an error instead of simply assuming a copy operation failed

These problems are especially challenging and more likely to occur with large file transfers. They are also challenging to handle well. In fact, developers could easily spend more time writing error handling code than code that actually performs the file transfer functions! It is no wonder, when you build homegrown solutions to limited tasks, error handling often comes up short.

Difficult to Maintain

Custom code can be difficult to maintain. The first version may be clear and logical to the original developers, but as new requirements emerge and other developers add modules and correct bugs, the logic may become less clear and less cohesive. Ironically, the very act of maintaining custom code can decrease the maintainability of the program.

Difficulties arise from two sources. First, there are business requirement difficulties. The original developer may be aware of some non-obvious requirement that she rightly codes for but does not document as well as she should have. Subsequent maintainers may struggle to understand why a particular piece of code is written the way it is or why it is there in the first place. Without knowing the business requirements, another developer may alter the functionality of the program and inadvertently remove an important feature.

The second source is programming difficulty. Sometimes the more efficient the code, the more difficult it is to understand. Code with minimal data structures and tightly written logic may execute quickly but be hard to understand. Scripting languages can pack a good bit of functionality into operators, and programmers can take advantage of side effects of operators to improve speed and reduce the number of lines of code they need to write. This is often desirable but, like so many other aspects of managing file transfers, there are tradeoffs.

There has always been the potential for homegrown solutions to crop up in projects across the enterprise. This problem will likely grow worse with the adoption of public cloud computing. An individual analyst with an idea, some data, and a credit card can take her project to a cloud provider like Amazon EC2, Windows Azure, or any of a variety of other providers. Cloud computing has the potential to help companies develop better insights into their data while cutting their capital expenditures on hardware. Unfortunately, it is also an opportunity to compound existing problems with ad hoc file transfer solutions.

These characteristics of homegrown file transfer solutions are closely tied to a number of shortcomings that are undesirable from a software development perspective as well as from the point of view of business users.

Shortcomings of Homegrown Solutions

The major shortcomings of homegrown solutions can be broadly grouped into five categories:

- Lack of robustness
- Poor adaptability to emerging requirements
- Poor scalability
- Potentially poor security
- Cost of maintenance

These shortcomings are directly related to the adverse business consequence you see with many homegrown solutions.

Lack of Robustness

Robustness is the quality of being able to adapt to a wide range of inputs and operating conditions. Applications that lack robustness, sometimes referred to as *brittle applications*, break easily when explicit and implicit assumptions of the application developer are not met. Imagine a simple but brittle file transfer script that copies a single file from a local directory to a shared network drive:

```
copy c:\temp\data.txt z:\stage\data.txt
```

For such a simple command, it can break in many ways:

- The source file may not exist
- Users running the script may not have read access to the source directory or write access to the target directory
- There may be insufficient space on the target disk
- A file with the same name may already exist in the target directory
- The network mapping specifying the location of the Z: drive may not be defined

These issues do not even touch on the lack of flexibility and adaptability of the script.

A robust script is one that can function with multiple sources and targets, gives the user opportunities to recover from errors, and adapts to non-fatal conditions, such as a file with the same name already existing in the target directory. Certainly, many developers who create custom file transfer scripts have the ability to write robust code; the problem is the time needed to do it. As the list of potential problems shows, there are many ways to break a file transfer application.

With the increasing use of public cloud providers, developers also have to account for file transfers between on-premise servers and cloud storage. As businesses continue to generate, store, and analyze large data sets, there is an increasing need for computing resources for data analysis. Public cloud providers can meet these computing needs, but successful implementation of this type of big data analytics depends upon the ability to transfer data to the cloud reliably and in a timely manner.

Poor Adaptability to Emerging Requirements

Requirements can change in several ways, including the range of inputs that must be handled, the window of time allotted to perform a transfer, the security requirements related to the transfer, and the locations to where files are transferred. Seemingly simple changes can highlight implicit assumptions that do not always hold. For example, the previous simple example assumed we were moving a specific file from one particular directory to another. Granted, this is a trivial example and most of us would have written the script to accept a source file path name and a target file path name. That is just the beginning of the ways you could design the program to be more adaptable. For example, you might want to consider the following questions during the design process:

- Should you assume the copy operation would be between devices on the same network?
- Should you avoid mapped network drives and use server names instead?
- Should you use the syntax specific to Microsoft OS domains (for example, \\servername\shared_directory\) or should you use Internet domain names (for example, datastage.mycompany.com) instead?
- Can you assume there will be a single source and single target files? Should you accommodate the possibility of merging multiple input files into a single target file?
- How will files be transferred from on-premise file systems to object-based cloud storage?

The ability to scale to emerging requirements is another type of potentially new requirement. Scalability is a multifaceted and complex area of file transfer management.

Poor Scalability

Scalability is the ability to continue to function under increased workloads while meeting performance objectives. In the case of file transfer applications, you can increase workloads along at least four different dimensions:

- Volume of transfers
- Frequency of transfers
- Support for new transfer partners
- Integration with new business processes

File transfer applications should be able to scale up, as with the first two dimensions, and to scale out, as with the last pair of dimensions.

Volume of Transfers

The most obvious need for scalability is when the size of files or the number of files increases. In order to meet SLAs, file transfers have to complete in a specified period. Whether it is order information that has to be posted to customer accounts or transaction data that needs to be added to a data warehouse, file transfers have to complete in time for the entire workflow to finish on time. There are several approaches developers can take to improve throughput: compressing files before transfer, using multiple network connections to transfer files in parallel, and using deduplication techniques to transmit duplicated blocks of data only once. These techniques of course add complexity. The effectiveness of each technique will vary depending on the characteristics of the data transferred.

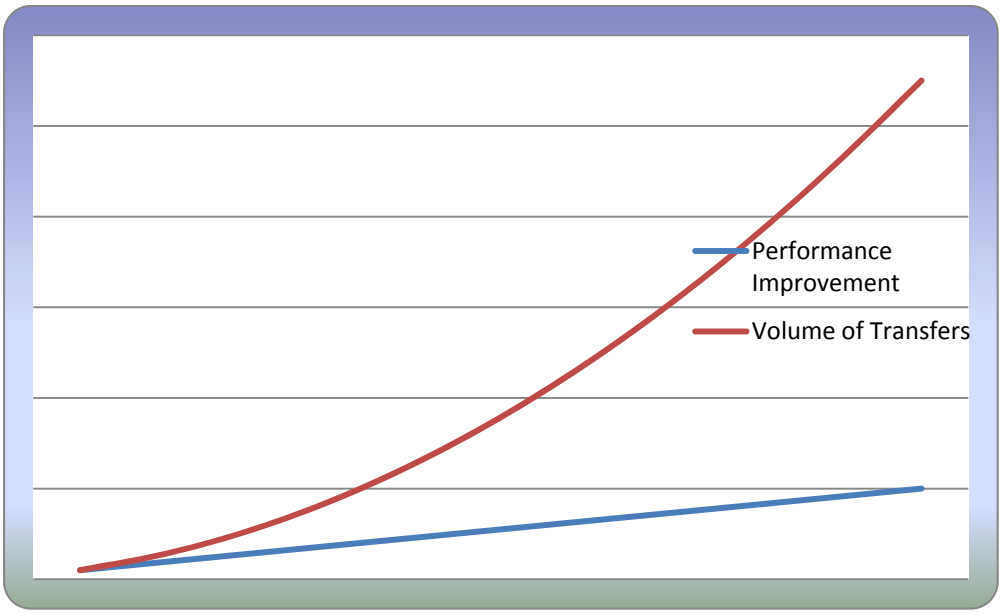

Figure 1.3: The level of performance improvement readily implemented in custom file transfer scripts can be easily outpaced by the growth in volume of transfers.

Frequency of Transfers

Increasing the frequency of transfers can have similar impacts as increasing the volume of transfers. The net effect in both cases is that the amount of data transferred during a given period of time is increasing. In addition to the potential issues outlined with regard to increased volume of transfers, there may be need for additional code to handle problems introduced by more frequent transfers. For example, later stages of processing may not complete fast enough to process previously transferred files. In this case, multiple files will be staged on the target. The transfer program will have to avoid potential filename conflicts when writing files to the staging directory.

Support for New Transfer Partners

Moving on to the scaling out dimensions, let's consider the need to support new transfer partners. Unless a homegrown solution is written in a sufficiently abstract and generalized way, it can be difficult to adapt to new transfer partners, especially with regards to security. Transfer partners may have different authentication mechanisms; how will the homegrown application accommodate those? Will the authentication module have to be rewritten for each new trading partner? How will security information, such as usernames, passwords, or digital certificates, be managed? How will a cloud provider's authentication mechanism, such as authentication keys, be supported? Is there a generic model in place to handle all of these or will there be partner-specific modules? If the answer is the latter, this will significantly curtail the speed with which the transfer program can be used with new trading partners.

Integration with New Business Processes

The last dimension of scalability is the ability to integrate with new applications and processes, which is especially important with the increasing use of cloud computing. In an earlier example, we saw how a BI analyst had to write a custom program to transfer click stream data from a Web server to a statistical analysis package. A file transfer program was already in place for moving data from the Web server to the data warehouse. Ideally, that program should have been used for the click stream data as well. After all, it already worked with the Web server, so issues such as scheduling a process and running it with proper authorizations had already been implemented. A scalable solution is one in which common requirements, like scheduling and authentication, do not inhibit the use of existing transfer applications to new applications. Another common shortcoming in custom, homegrown solutions is they may contain security vulnerabilities that expose the business process to unnecessary risks.

Potentially Poor Security

Information security often entails a balance between security and functionality. The most secure system is also unusable. In the case of file transfers, there are several potential risks that can be introduced with custom solutions:

- Putting usernames and passwords in script in clear text. For example, if ftp is used to transfer files and a username and password is required to authenticate, one could simply code a line such as ftp://username:password@ftp.mycompany.com
- Authenticating with user accounts with more privileges than are required to perform file transfers. For example, using an existing user with elevated privileges rather than creating a dedicated account for file transfers is a potential security risk.

- Using unpatched ftp servers or other applications with known vulnerabilities creates risks. OSs and utility programs used in file transfer should be under patch management controls. This is more difficult if there are multiple copies of vulnerable applications in use.
- If file management practices are not secure, one could compromise the security of the entire process, for example, by failing to minimize the storage of sensitive information in the DMZ of the network and to minimize the number of open ports on the internal network.

As the issue with patching points out, security is dependent on sound maintenance, which brings us to our final shortcoming of homegrown solutions.

Cost of Maintenance

Software maintenance is an ongoing need of any production application. You cannot eliminate the cost of maintenance but you can keep it to a minimum. Many software engineering best practices are designed to make code more understandable, reliable, and robust and that in turn makes the code less costly to maintain. Applying these best practices takes time and a willingness to commit resources to the effort. When you short-change development of file transfer programs because you assume file transfer is an easy task that is only a small part of a larger workflow, you set up your business for increased long-term maintenance costs.

The increased costs are obvious in a couple of maintenance areas. First, there is the cost of troubleshooting. The more dissimilar transfer applications are, the more difficult they are to maintain as a group. There is little or no learning that can be carried over from debugging one application to the next. Transfer programs may be written in different programming languages, use different network protocols, and apply different strategies for improving performance or optimizing for space. The lack of common programming models also makes it difficult to find bottlenecks. The root of a performance problem in one program may be unrelated to the same performance issue in another application. Just as you "reinvent the wheel" each time you create yet another custom file transfer program, you are also creating additional costly maintenance work for yourself or other developers.

The shortcomings commonly seen in homegrown file transfer solutions are not just a problem from a software engineering or IT management perspective; they directly impact business operations outside of IT. The lack of robustness, poor adaptability, poor scalability, potentially poor security, and the cost of maintenance directly contribute to unmet business requirements.

Unmet Business Requirements

Up to this point, we have considered common characteristics of homegrown file transfer solutions and their shortcomings. Many of the shortcomings stem from the fact that many of these programs are designed according to one set of requirements and then you expect them to function against another set of requirements. It is easy to imagine conversations in meetings about new requirements:

- We have a Perl script for transferring files between the Web server and the data warehouse, why don't we just use that to transfer transactions from the customer order system to the data warehouse?
- Let's use the script for downloading data from the West coast parts supplier with that new supplier in Texas.
- Sales volume is up, let's run that file transfer program every hour instead of every night so that we can get more up-to-date figures to the sales managers.

These are all reasonable suggestions on the surface. It is only when you dig deeper that the problems become clear, such as scripts written to run on one OS that won't run on another, transfer partners that use different security systems requiring different application code, and scripts that ran nightly and were not designed to finish in the short timeframes needed for operations during the business day. At this point, you should be starting to see the business impact of homegrown file transfer solutions.

Don't Shoot the Messenger (or in this case, the Developer)

We should note that the problems and shortcomings outlined here do not arise because programmers do a poor job of writing code. Rather, we tend to limit the requirements we give to programmers while at the same time imposing strict timetables for delivering on the application. This results in limited room for the developer to implement more adaptable, scalable, and robust programs. This will only change when we see the need for managed file transfers as instances of a specific type of information management problem and not just an isolated need within a single project.

The unmet business requirements fall into several categories:

- Compliance
- Ability to keep up with increasing need for file transfer services
- Support for multiple uses
- Ease of management
- Cost control
- Support for workflows

IT operations are expected to support compliance with regulations addressing the privacy of personal information and the integrity of business information as well as industry-specific governance issues. Often, to be in compliance, you must not only be doing the right thing, you have to be able to demonstrate that you are doing the right thing. In other words, it is not enough to have a file transfer program that uses encryption to transfer files, you need logs showing those programs were used, the encryption was strong enough (for example, transfer was done with a strong encryption algorithm and sufficiently long keys), and the transfer occurred between two servers with verifiable identities. This level of detailed logging and reporting may not be found in all custom transfer programs.

Business is anything but static. Unfortunately, it is very straightforward to design and implement a file transfer program for today's requirements and maybe a bit more, but it will likely be insufficient for what might be needed in the not too distant future. Project managers who are responsible for implementing a new service or workflow may plan for some increase in scale, but it is unreasonable to expect them to expend time and resources developing a file transfer system that would be useful to another, separately funded project. The logic of project budgeting reduces and even eliminates incentives to support multiple uses and curtails even the willingness to dedicate resources to designing a scalable solution. This is especially the case when scalability issues will only become clear once the system is deployed and maintained out of a separate, IT operations budget.

Ease of management, cost control, and workflow efficiency all fall into a similar category. Project managers, department heads, and developers can all make rational decisions about file transfer and end up with systems that fail with regard to these issues. The real cost of these shortcomings is borne by the business at large. Chapter 2 will delve into more detail about the business consequences of homegrown file transfer solutions. Before turning to that, though, let's briefly discuss a more structured approach to business file transfer.

Defining a More Structured Approach to Business File Transfer

A more structured approach to managing file transfers is required to avoid the shortcomings and adverse consequences of developing and maintaining multiple custom solutions. The first step is to recognize that file transfers are complex tasks that require appropriately designed software. From there, it becomes clear that you cannot develop this level of complex software with the same resources or for the same cost as a custom solution for one isolated use case. Ideally, the structured solution will be a general-use tool that scales up and scales out to meet the demands for file transfer across the enterprise. This requires four distinct functions:

- Centralized management for defining transfer jobs, monitoring the status of executing and scheduled jobs, prioritizing tasks, and other administrative functions
- Security services to manage authentication with transfer partner systems, encrypt confidential data, log transaction details, and enforce security policies governing file transfers; security services should work with reporting and logging services to keep administrators informed of significant security events, such as failed login attempts during transfers or instances of insufficient privileges to perform a transfer

- Task specification mechanism to define transfers—this should include the ability to define the schedule of transfers, choose error handling options, define security requirements, securely capture and store usernames and passwords, and specify triggers or event processing steps, such as executing a script upon successful transfer. The task specification mechanism should support transfers to business partners, including cloud providers.
- Reporting and logging services are needed for both tracking individual transfer jobs and for monitoring global trends, such as increases in the number of files transferred, the volume of data transferred, error rates, and security-related issues

Each of these services compensates for some of the shortcomings commonly seen in custom solutions. Although it is not listed, scalability is a critical feature of a managed file transfer framework. Unlike the reporting services or task specification mechanism, scalability is a global property of the way the system is designed. Scalable systems will use a combination of features, such as compression protocols, multiple connections, and multi-threaded processes, to avoid bottlenecks and maintain sufficient throughput.

Scalability is a key consideration when supporting large-scale analysis operations, commonly referred to as "big data." Big data typically includes analyzing large data and sometimes diverse data sets, such as various types of logs. These data sets are generated and stored on many servers but have to be consolidated in a cluster or cloud computing environment for analysis. In order to keep pace with these volumes of data and the pace at which they are created, it is imperative that you have a scalable file transfer solution in place.

The chapters that follow will further examine how a structured framework for file transfer works to address the unmet business needs that remain when custom solutions are in use.

Summary

The need to transfer files is ubiquitous in IT operations. We are constantly moving data between desktops and servers and between servers within a business, but we are moving data between transfer partners as well, including cloud computing providers. It is easy to see each instance of the need for a file transfer as just another task in a more complex workflow. When you do that, you tend to create custom homegrown solutions to solve the immediate problem at hand. Over time, as you keep repeating this pattern in other projects and in other parts of the organization, you will find that you have created a sprawl of file transfer programs with some unfortunate shared characteristics. The most important are lack of scalability, costly maintenance requirements, poor security, and an inability to adapt to changing business requirements. These, in turn, lead to a host of unmet business requirements. Fortunately, by considering these file transfer needs as instances of a more general IT operation, you can justify dedicating resources to solve the enterprise-scale problem facing the business. The forthcoming chapters will examine how managed file transfer solutions address enterprise requirements for secure, reliable, and robust file transfer services.

Chapter 2: Analyzing 5 Key Drivers for Improving File Transfer

IT professionals are constantly faced with a wide array of new technologies promising greater efficiencies, higher performance, and faster development times. Why should they concern themselves with file transfer, which is, after all, a solved problem? The truth is file transfer in many organizations is a partially solved problem. Ad hoc solutions, which are programs designed to meet a single file transfer requirement or a small number of requirements, typically do not fully address the range of business requirements. Certainly, ad hoc solutions can copy files from Point A to Point B but that is not enough.

File transfers have to meet certain levels of reliability, performance, security, cost effectiveness, and accessibility. When ad hoc solutions come up short on these requirements, businesses will likely find they have increased risk of security lapses, incurred hidden costs of debugging and maintenance, and been forced to work around performance problems. These unmet requirements underlie several key drivers to adopting a more managed file transfer solution, including the need for

- Compliance
- Flexibility and scalability
- Management
- Cost controls
- Workflow efficiency

This chapter will discuss how file transfer solutions affect each of these drivers and highlight ways managed file transfer solutions can help meet these requirements.

Compliance

Businesses are expected to conduct operations in ways that mitigate risk to the enterprise. Government and industry regulations are codified expectations of what kinds of information is to be protected and minimum standards for protecting that information. After large-scale corporate accounting scandals and well-publicized data breaches, it is not surprising that regulations have been established to protect the integrity and confidentiality of corporate and personal information.

Some regulations are broadly applicable and others are industry specific. Regulations often have common objectives with regard to data integrity and confidentiality and are therefore applicable to file transfer practices. We will consider five diverse regulations in order to form a sufficiently comprehensive picture of the types of requirements these regulations establish with regards to file transfers. The regulations we will consider are:

- Sarbanes-Oxley (SOX) Act
- Payment Card Industry Data Security Standard (PCI DSS)
- Health Insurance Portability and Accountability Act (HIPAA)
- Gramm-Leach-Bliley Act (GLBA)
- 21 CFR Part 11

These span the broadly applicable, such as SOX to the transaction specific (PCI DSS) to the industry specific, such as HIPAA, GLBA, and 21 CFR Part 11. In all these cases, file transfers can be part of data flows subject to some degree of regulation.

SOX

SOX is a set of regulations governing the financial reporting of publicly traded companies. The regulations were established in the wake of several accounting scandals that were deemed so potentially damaging to trust in the market that a federal law was passed to establish standards for protecting the integrity of financial reporting. At first glance, it might appear that SOX would only apply to file transfers involving financial reporting information, such as transferring a general ledger. This is not the case, as we shall see in a hypothetical scenario; but first, let's review some of the key requirements of SOX.

IT-Related SOX Requirements

Perhaps the best known, and most challenging, part of SOX that relates to IT responsibilities is Section 404. That part of the legislation describes requirements for management and IT to ensure proper internal controls with respect to financial reporting. The specific requirements include a number relevant to file transfers that may be involved in financial reporting:

- Assessing the adequacy of internal controls to mitigate the risk of misstatements in financial reports
- Identifying points in workflows where tampering could occur
- Understanding controls to prevent tampering and detect it if it occurs
- Assess controls on the financial report generation process

There are more requirements of information technology professionals but these are enough to demonstrate how poorly secured and insufficiently monitored file transfer procedures can undermine SOX compliance.

SOX and File Transfers

Given the requirements on information technology processes, it is clear that transferring files containing financial reports are subject to SOX. These are not the only times in which SOX is relevant, though. The financial reports themselves are based on data from potentially many sources. Any tampering with data in these source systems or when data is transferred from one system to another can result in compromised results. Just as a polluted tributary can contaminate other rivers, a compromised data source can undermine the integrity of other data management systems that use it as a source system.

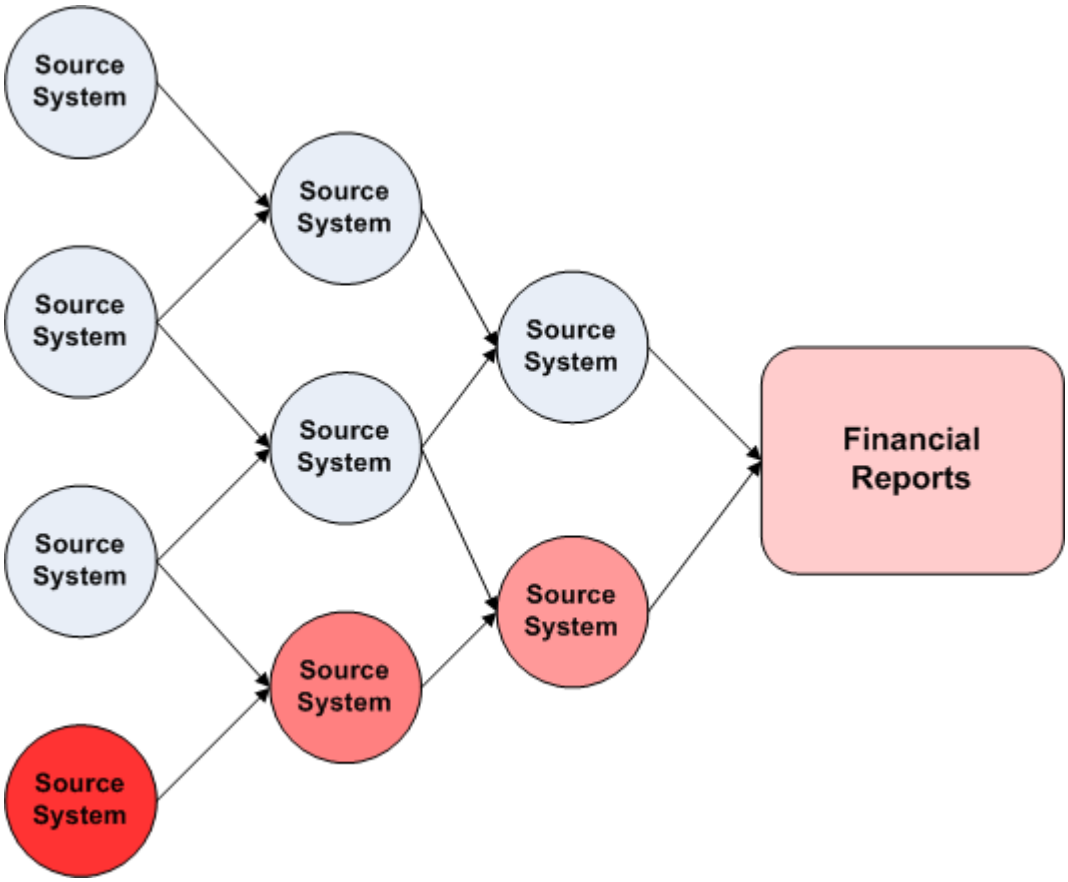

Figure 2.1: Financial reports are generated from source systems that are dependent on other systems through a chain of dependency. Lack of integrity at any point can compromise the integrity of the final product.

Information Flows and Application-based Controls

Consider, for example, the flow of information that goes into generating a financial statement. Of course, there are the basic sections of a financial statement, such as the balance sheet, cash flow statement, and income statement, but where does the raw data come from? For example, the value of assets on hand could include the value of inventory that is stored in a business partner's warehouse. Let's consider how a seemingly simple file transfer process between business partners could undermine the integrity of financial reporting.

In this scenario, a business has contracted with a business partner in Asia to store inventory close to the manufacturer, which is also in Asia. Orders to Asian customers ship directly from those warehouses, while orders destined for Europe and North America are shipped first to warehouses on the respective continents. The European and North American warehouses share inventory levels with the warehouse managers who maintain sufficient levels of inventory to meet local demand.

The business partner uses its own enterprise resource planning (ERP) system to manage its warehouses but it reports to the owners of the inventory every week on current inventory levels in the Asian warehouse, details about what was shipped directly to Asian customers, and products staged at European and North American warehouses.

From a financial reporting perspective, one company owns this entire inventory and must report it regardless of the fact that it is distributed around the globe. Let's assume that each of the business' ERPs is well designed, properly configured, and meet best practices for complying with accounting standards. A CIO could be reasonably confident that these systems meet SOX requirements and would be willing to sign off on the compliance report. What about the file transfer process that is used to move inventory data from the warehouse managers to the inventory owner?

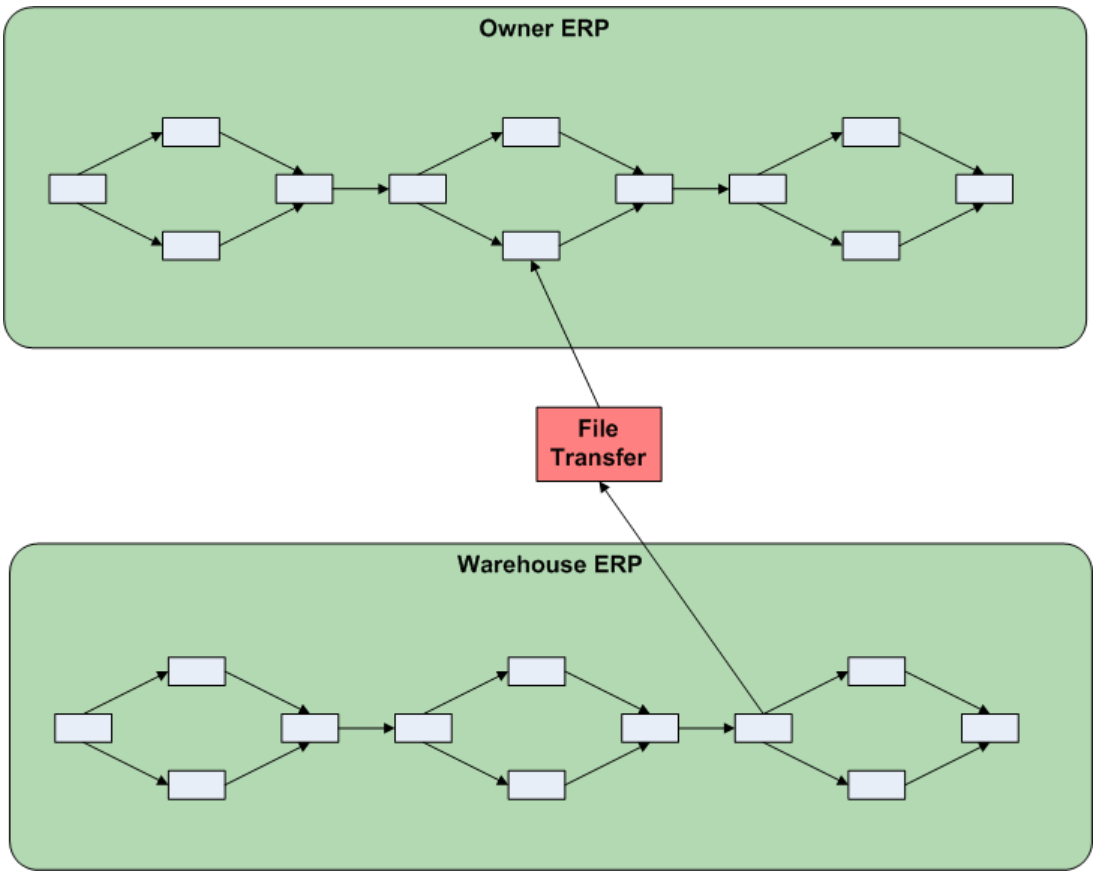

Figure 2.2: Source systems, such as ERPs, may be fully in compliance, but data exchange procedures between these compliant systems are not necessarily compliant.

The Weakest Link: File Transfers

ERPs and other enterprise-scale applications are designed from the start to provide an integrated flow of data through multiple processes. No matter how many ERP modules are licensed, there will be times when business requirements cannot be met within the ERP and data has to be transferred to some other system. In the case of the business and its warehousing partner, data exchange is needed to move data between ERPs.

Once the source ERP generates the data to transfer to the other ERP, the source system's controls no longer protect the data. The data is not in the target ERP, so that system's controls are not yet protecting the data either. The data has entered a veritable "no man's land" of integrity controls. To mitigate this situation, the workflow that transfers the data from one ERP to the other could employ several techniques:

- Writing the data to a file or set of files to a secure directory with sufficient access controls
- Encrypting the file to prevent unauthorized access to the information in the data files
- Calculating a message digest that is passed along with the file to ensure that there are no unauthorized changes to the files
- Logging any changes to the files and recording metadata about changes, such as the user ID making the change and time of change

The problem with this scenario is that developers writing ad hoc file transfer programs may not be aware of compliance issues or even the fact that the data being transferred is materially relevant to financial reporting in the first place. Line of business managers, for their part, may not be aware of the low-level technical details of file transfers and could be understandably unaware of the fact that a seemingly simple step in a complex information flow could leave essential data vulnerable to tampering. Although there are several ways to mitigate risks with ad hoc file transfers, there is no guarantee that those responsible for implementing them are aware of the need for them. This scenario depicts the limitations of ad hoc file transfer solutions with respect to SOX, but as the following sections will show, similar problems arise with other regulations as well.

PCI DSS

If you follow information security news, you have probably heard several times about the "largest data breach to date" involving either a retailer or a credit card processor. The credit card industry has stepped in with a bit of self-regulation to reduce the risk of credit card data breaches by establishing PCI DSS.

Resources

See the Privacy Rights Clearinghouse "Chronology of Data Breaches" at http://www.privacyrights.org/ar/ChronDataBreaches.htm for a sobering list of incidents; those that include credit card data as well as other types of data breaches.

The PCI DSS establishes several control objectives designed to minimize the risk to credit card information including:

- Building and maintaining a secure network
- Protecting cardholder data
- Maintaining a vulnerability management program
- Implementing access controls
- Monitoring networks
- Maintaining an information security policy

The second, third, fourth, and sixth control objectives are directly relevant to file transfers.

Protecting cardholder data includes encrypting that data anytime it is transferred outside of a secure network. Vulnerabilities can exist in workflows that include file transfers, but those vulnerabilities may be difficult to detect when custom scripts are used to transfer data files. Access controls may be the responsibility of systems administrators who manage directories where transferred data files are staged. The information security policy is highly relevant to the way file transfers are performed; however, that information may not be captured in design requirements provided to the programmer developing a custom transfer script. Unless there is a review and enforcement process in place, how can management be sure the policy is actually implemented?

HIPAA

HIPAA is legislation designed to protect personal health information. Part of the regulation, known as the Security Rule, specifies three types of safeguards:

- Administrative
- Technical
- Physical

The administrative safeguards describe policies that should be in place regarding governance and management oversight, access controls, training, and emergency response. The technical safeguards focus on access controls to data and protections of transmitted data. These include, for example, controls to prevent unauthorized changes to data, checks to verify the integrity of data, and the use of encryption to ensure confidentiality of private data. Physical safeguards address access concerns related to hardware and physical facilities.

With regards to file transfers, both technical and administrative safeguards must be in place. For example, if a file containing protected health information is transferred from one healthcare provider to another, several issues must be addressed:

- Is the file encrypted?
- Is a checksum, message digest, or other integrity check calculated for the file?
- Is the transfer partner authenticated using a digital signature or other means of authentication?
- Is a digital signature applied to the file so that the receiving party can verify the source of the transmission?
- Are access controls in place to prevent tampering with the file after it is generated but before it is transferred?

As with ERPs, data may be well protected when it resides within healthcare information management systems, but the process of exporting and transferring data, such as from a healthcare provider to an insurance company, can be the most vulnerable point in the process.

GLBA

In 1999, the United States repealed a long-standing law that kept commercial banks, investment banks, and insurance companies separate. The legislation, GLBA, included privacy safeguards to protect customer data that would be maintained by financial services companies. The most relevant part of the legislation to file transfer processes is the Safeguard Rule.

Under the Safeguard Rule, financial services firms are required to

- Evaluate the risks to private information maintained by the company
- Develop, implement, and monitor security practices to protect private information
- Establish management oversight for those security practices

Although the GLBA's safeguards may not be as precise as other regulations, they clearly dictate the need to protect customer information when stored and when transmitted. The GLBA opened the door to mergers of different types of financial services firms, which would clearly benefit from exchanging customer information for cross selling, marketing, and other business development purposes. Much of that exchange could be done using file transfers, thus creating the same potential security threats we have discussed in sections about other regulations.

Figure 2.3: Information sharing across financial service segments is allowed under GLBA, but private data must be protected, including during transmission.

21 CFR Part 11

21 CFR Part 11 is a section of the Code of Federal Regulations that specifies requirements for pharmaceuticals, medical device manufacturers, and others regulated by the Food and Drug Administration (FDA) with regard to electronic records. Clearly, the FDA has an interest in protecting the integrity of records related to the design and manufacture of drugs and medical devices. As with other regulations discussed in this chapter, the code specifies the need for access controls, audit trails, written policies, and so on. There is also special emphasis on electronic signatures in this regulation.

Data integrity is particularly important for the FDA. If, for example, there were a problem with a particular batch of a prescription drug, the FDA and manufacturer would want clear records of the components used to manufacture the drug, the location of the manufacturing, the distribution of the drug, and other information needed to recall the product. File transfer processes would once again be required to maintain sufficient information and enforce adequate controls to meet these requirements.

Compliance and File Transfer Requirements

Generalizing from the several regulations we have discussed, we can conclude that at minimum, regulations typically demand:

- Security procedures to protect the confidentiality of private data and the integrity of operational and financial data
- Effective controls to implement security policies
- Management and audit reporting sufficient to demonstrate compliance with the regulation

Repeatedly through this discussion, we have seen that file transfers can be a weak link in the overall security posture of a workflow. Enterprise applications, such as ERP and healthcare information management systems, may have sufficient controls in place to meet regulations; however, once data is exported for transfer to other systems, those controls are no longer protecting the exported data. At that point, the file transfer system should assume responsibility for securing data. Unfortunately for the developers of ad hoc file transfer solutions, that is a significant task to take on.

Developers should also keep in mind that security measures provided by a company's internal network may not be in place when working with public cloud providers. Consider the possibility that a custom file transfer program lacks sufficient security controls. For example, a custom program might have weak authentication, allowing someone to perform an unauthorized transfer. Fortunately, other controls are in place that log the fact that a file is created on the target server. If the transfer is made to a cloud storage provider instead of an internal server, no record of the transfer will appear in the log. Custom scripts that worked well with on-premise transfers might not be sufficient when working with cloud resources, including storage and software as a service (SaaS) providers.

Flexibility and Scalability

In addition to external drivers, like compliance, we have internal drivers that promote the adoption of managed file transfer methods. One of those is the need for flexibility and scalability. Flexibility is important because there are so many ways in which file transfers are used within an enterprise, between enterprises, and with external services, such as cloud providers. A file transfer method is ideally suited for multiple uses cases. Scalability has obvious implications for meeting service level requirements and continuing to meet those as demands change.

Three aspects of file transfer operations are relevant to understanding flexibility and scalability:

- Size of files transferred
- Volume of files transferred
- Number of transfer partners

These individually and collectively contribute to the need for managed file transfer solutions.

Size of Files Transferred

There is no "average" file size when it comes to managing file transfers. We could have small control files that pass between processes to indicate the state of one process or to pass a small number of parameters before initiating a process on another server. We can of course have quite large files. Some common large file exchanges include:

- Database dumps
- Multimedia files
- Extraction, transformation and load (ETL) files for data warehouses
- Input files to batch processing operations
- Network, server, and application log files

Difficulties can arise when transferring large files:

- Packets may be dropped and have to be re-transmitted
- Connections can be lost during a transfer leaving the target file partially copied
- Large file transfers can run the risk of running longer than the time window allotted to them

A robust file transfer solution will scale to large file transfers by incorporating a number of features, such as the ability to recover a disrupted transfer from the point of failure without starting over again and the use of efficient protocols that minimize overhead while preserving the integrity of the transfer process.

Volume of Files Transferred

In addition to the need to transfer large files, scalability requires the ability to handle a large number of individual files. As the number of files grows, so does the overhead associated with tracking each transfer. Even without the demands of regulations, it is a good practice to log information about each transfer; logging can include details about:

- The name of the file transferred
- File size
- File type
- Start and end of transfer
- Encryption status
- Authentication on the target server

As the number of transfers grows, so does the volume of logging data associated. Ideally, this information is centralized; that would ease management overhead.

It is especially important to ensure the integrity of file transfers when performing analysis on those files. For example, you might transfer a large number of application, server, and network log files to an on-premise cluster of servers or to a public cloud for analysis. The object of the analysis is to identify correlations between application performance and events on servers and the network. If data is missing because of a problem with file transfers, the analysis will not accurately reflect the actual performance of the application and correlated events.

Number of Transfer Partners

An increasing number of transfer partners, like the number of files transferred, can create scalability issues as well. Difficulties with increasing number of transfer partners tend to cluster around management issues:

- Scheduling transfers according to different partners' requirements
- Managing multiple authentication credentials
- Tracking different transfer requirements, such as whether to encrypt files before transferring them
- Providing custom reporting to trading partners

Flexibility and scalability are key drivers to adapting a managed file transfer solution. The demands for scalability manifest themselves in terms of file size, number of files, and the number of transfer partners.

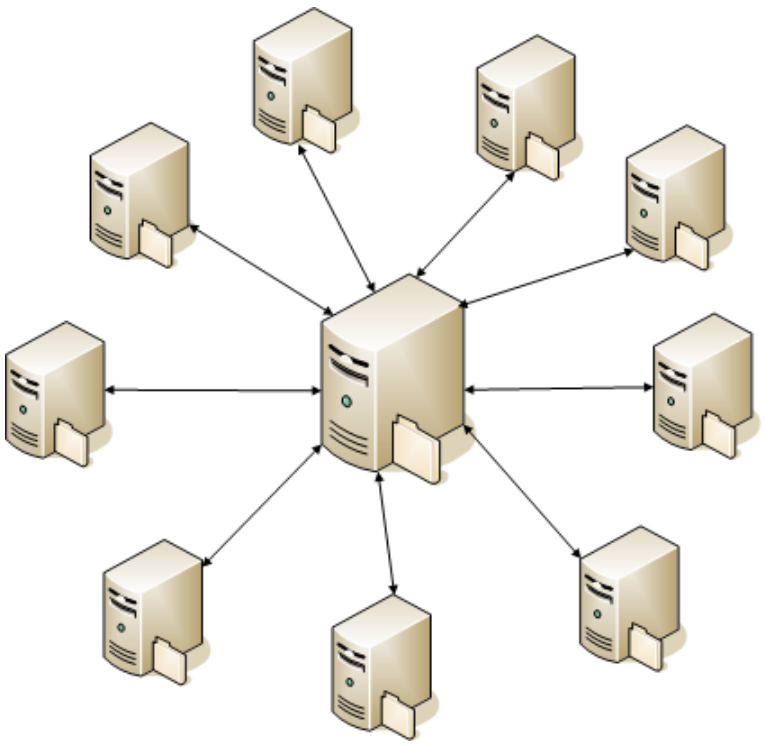

Figure 2.4: Increasing the number of trading partners increases the overhead of file transfer operations.

The need for scalability across these dimensions stems, in part, from the way we design applications and workflows today. They are highly distributed and can require large volumes of data transfers. Exchanging small amounts of data, for example, a single sales order, can be done with programmatic interfaces such as Web services; bulk transfers are still better done via file exchange.

Management Issues in File Transfer

As file transfer processes are so important to business operations, managing those processes has become a key driver to adopting managed file transfer practices and solutions. Three important issues in file transfer management are:

- Centralized control
- Monitoring and alerts
- Management reporting

Centralized Control

With so many applications of file transfers, from loading data warehouses and exchanging data sets for batch operations to sharing data with business partners and replicating data for disaster recovery protection, there is a growing need for centralized control.

Centralized management provides the ability to monitor:

- Status of file exchanges
- File exchange schedules
- Trends in growth of number of exchanges
- Trends in growth of data volumes
- Required procedures for exchanges, such as the use of encryption for exchange of confidential or private data

Of course, we could manage all these aspects in a more distributed fashion, but it would be less efficient and more prone to error.

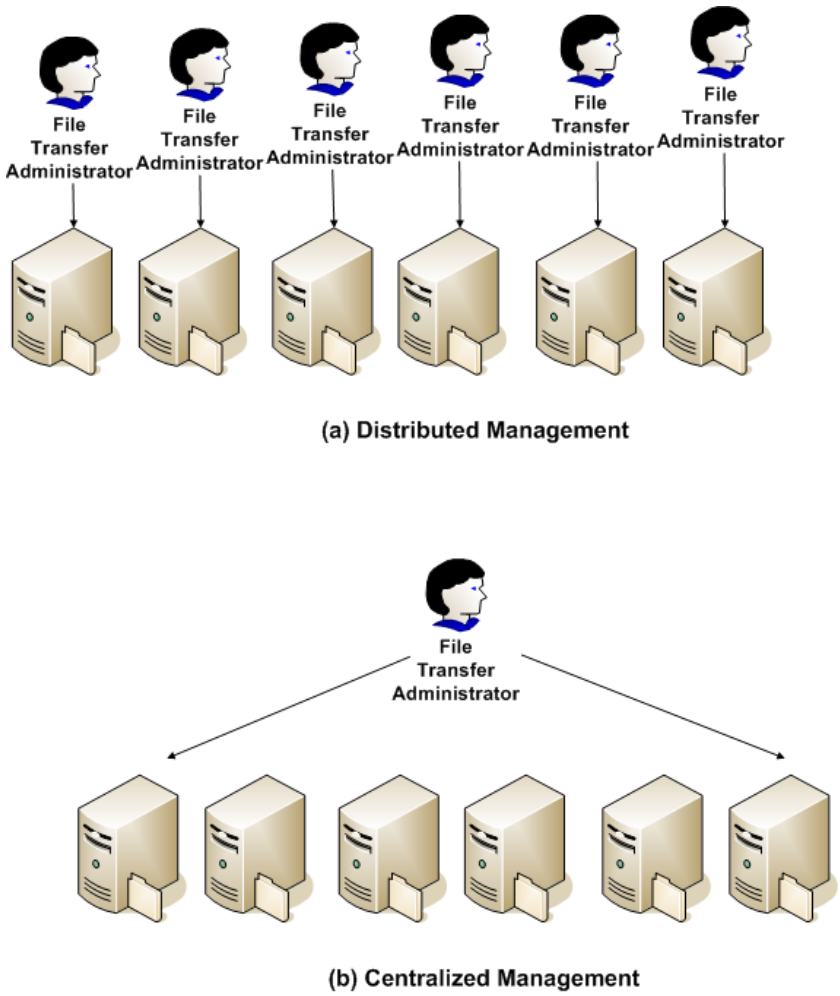

Figure 2.5: Distributed management is less efficient than centralized management.

Monitoring and Alerts

In addition to the normal, expected processes we need to manage within file transfer operations, we have to plan for the unexpected. Monitoring is the routine process of reviewing essential performance indicators:

- Time required to transfer files
- Average time to transfer per file size unit
- Number of encrypted/unencrypted transfers
- Number of transfer partners sending and receiving files

Monitoring helps to establish baselines and detect trends and changes in those baselines. This is especially useful when planning for future requirements.

Alerts keep us aware of unexpected events that need more immediate responses. Alerts may be triggered, for example, when:

- File transfers fail due to insufficient space on the target server
- Authentication fails on the target server
- Target server is unreachable
- Connection is refused by target server

With proper logging, alerts can also be tracked over the long term along with other areas covered by management reporting.

Management Reporting

Management reporting differs from monitoring and alerts in that the former provides a more aggregate view of the status of file transfer operations. Management reporting is especially difficult when ad hoc, homegrown solutions are used because each program may have its own method for structuring data, log entries, and management reports (if they exist at all). In a centrally managed system, a single log of events can be used to generate data about operations across multiple jobs, users, and even departments to provide a comprehensive overview of file transfer operation.

Cost Control

The discussion of centralized management and monitoring demonstrates the impact of distributed management on cost of implementing and maintaining multiple file transfer solutions. Certainly, different users will have different requirements, but a properly designed, centralized file transfer solution will be able to adapt to multiple requirements.

For example, one process may require a single file to be transferred from one local server to another at the same time every day. That is a pretty straightforward requirement. Another process may require multiple files to be transferred to a third party only after a particular event has been triggered. In addition, the transfer contains confidential information, so the files must be encrypted before transfer. These may sound at first like they need different programs to manage the transfer, but these tasks are easily accommodated by file transfer solutions that support job-specific configurations. A job configuration may include settings such as:

- Single or multiple files to transfer
- Transfer based on time or event
- Encrypt files prior to transfer
- Authentication required for external server

By using a single platform to perform multiple transfers and meet multiple use cases, we can reduce the cost of in-house deployment and maintenance.

Consolidating servers is an especially effective way to reduce costs. Consider the need for multiple security protocols. A business with multiple file transfer partners might have to support several security protocols, such as SFTP, FTPS, FTP with PGP, and HTTPS. Managed file transfer solutions can support all these protocols on a single server, reducing the need to procure and manage multiple servers to accommodate the varying requirements of different transfer partners.

Cost considerations play a major role in how you use cloud computing services. Many providers charge for servers by the hour. If you have 20 servers running but not enough data to keep them busy, you are paying for unused resources. It is important to have file transfer solutions that can transfer data to the cloud reliably and quickly. It is also important to have file transfer solutions that can recover from problematic transfer.

Some cloud providers charge for data transfers. How your file transfer solution handles errors can significantly affect your data transfer costs. Consider a large file download that fails after 80% of the file is downloaded. If the download must be restarted, you will pay for the 80% download as well as the next attempt to download the file. Assuming the second attempt is successful, you would pay 180% of the cost you would have paid had the file transfer not failed. Alternatively, if the file transfer solution detected a problem with the transfer and restarted the transfer from the point of failure instead of from the beginning, additional charges would be minimized.

A centralized system can eliminate the need for developing scripts targeted to single jobs that are managed by different administrators and debugged by different developers and that generate their own reports and log that are not easily consolidated for management reporting. Centralizing file transfers drives down the cost of developing, debugging, and maintaining file transfer solutions. The result is more efficient and cost-effective file transfer that also avoids the opportunity costs of having developers work on file transfer solutions when they could be working on other pressing business needs.

Workflow Efficiency

Another driver for improving file transfer efficiency is improving workflow efficiency. File transfers are typically one step in a larger workflow. If we ask the question “Why are we transferring these files?” we will likely find an answer that involves a complex series of steps implemented to support a larger business process. In a sense, no file transfer is an island unto itself. With this in mind, we can see two opportunities for improving workflow efficiency through improved file transfer.

Integration with Existing Workflows

The first is integration with existing workflows. Businesses are constantly executing workflows, from simple transaction-based workflows to complex multi-department operations. Often we find that workflows can be optimized and made more efficient by studying the global requirements of a process. For example, if we looked at just the first few steps in a workflow, we may miss the fact that later steps duplicate some of the same processing. By considering the global workflow, we may find new opportunities to optimize the process.

Similarly, if one workflow ends with the generation of a file to be transferred and another workflow begins after the file has been transferred, we may have the opportunity to optimize the overall flow by integrating the two workflows. The file transfer process becomes the lynchpin in this case; rather than having two workflows, and the separate management overhead, we can combine into a single workflow with consolidated management, monitoring, and development.

Support for Multiple Trading Partners

We can extend the idea of integrated workflows across organizational boundaries when we support the needs of multiple trading partners. File transfers are parts of larger business processes. If we were to custom design a file transfer solution for every trading partner (as some of us have), we reduce the efficiency of the file transfer process, introduce a weak link into the process, and make management reporting and operations management more difficult. A single file transfer solution for multiple trading partners gives us many of the benefits we have seen for other applications, including: centralized management, improved monitoring, alerts, and the ability to better monitor trends.

Summary

There is no single reason to improve file transfer, there are several. Compliance with regulations is a major concern and ad hoc, homegrown file transfer solutions may not meet all the requirements of regulations, and if they do, they are costly to build and maintain. Business is dynamic and workflows have to be dynamic too. Flexibility and scalability are essential characteristics of an enterprise-scale file transfer solution, especially with respect to the size of files transferred, the number of files transferred, and the number of trading partners. Management needs both tactical reporting, such as alerts, and longer-term management reports on trends. Here again, a single centralized file transfer solution is better able to provide that information efficiently than a large number of custom solutions. Cost controls and workflow efficiencies are also key considerations driving the adoption of managed file transfer solutions.

Chapter 3: Selecting a File Transfer Solution: 7 Essential Requirements

No one intentionally sets out to deploy an insecure and unreliable file transfer system. There are no business drivers that prompt IT professionals to look for a system riddled with security risks or likely to fail during the course of normal operations. Problems with security and reliability often creep in unnoticed when developers focus more on delivering a finished product on time and on budget. This is especially a problem when file transfer is seen as a minor requirement or a simple task that does not warrant as much attention and consideration as other aspects of system design. This is a myth.

As we have seen in the previous two chapters, enterprise file transfer is a complex process with multiple sources of requirements, a wide range of constraints, and a target for security threats. The purpose of this chapter is to identify essential requirements for a secure and reliable file transfer solution. This identification should help frame discussions about design around actual requirements instead of myths that can thwart your long-term objectives. Before examining seven essential requirements of a secure and reliable file transfer system, let's dispel misunderstandings that can cloud your understanding of file transfer.

Dispelling a Few Misunderstandings About File Transfer

The first two chapters of this guide dispelled common myths about file transfer: that it's easy, should not require much code, and is basically a glorified file copy operation. Myths and misconceptions such as these stem from three fundamental misunderstandings.

File Transfer Is a Simple Technical Operation

Misconceptions stem from a narrow focus on technical aspects of implementing file transfer. For example, the file transfer protocol (FTP) was obviously created to transfer files from one system to another. FTP includes simple commands, such as "put," which copies a file from the system running the FTP command to a remote system, and "mget," which copies multiple files from a remote system to the system running the FTP command. By thinking of file transfer in terms of implementation and simple protocols, you can easily fall into the trap of thinking design and development efforts should focus on determining which FTP commands to use and how to wrap them in a Perl script or any of the other popular programming languages.

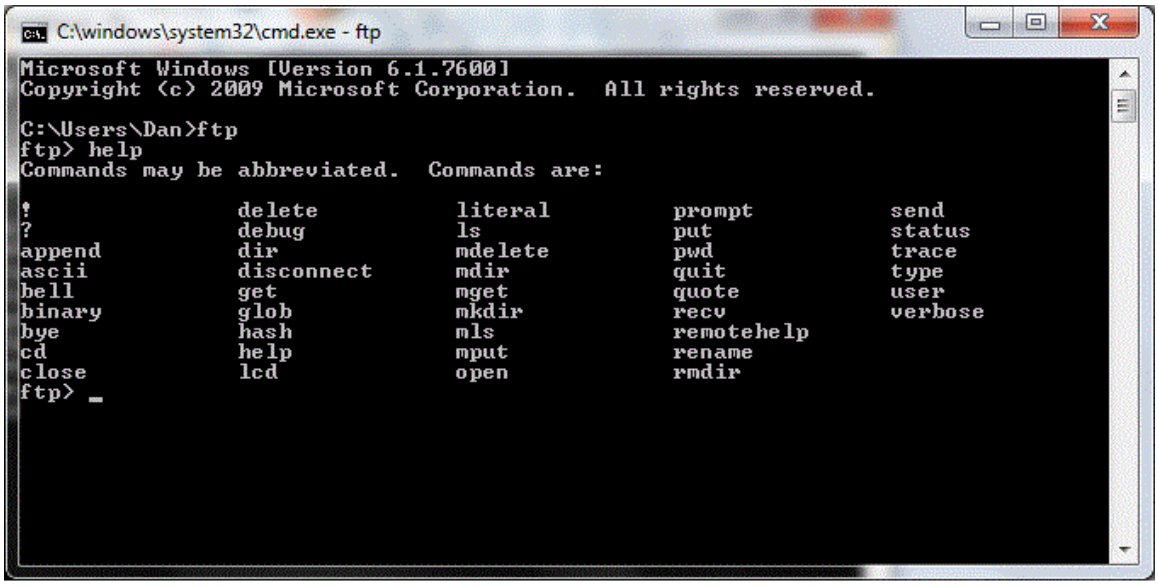

Figure 3.1: How complicated can file transfer be if a small set of commands can implement FTP? This is not a good indicator of the complex requirements of file transfer.

There Are Few Risks, Potential Errors, or Changes to Requirements

Second, many fail to account for the potential problems with file transfers. Perhaps designers and developers are inherently optimistic about the problems you work on and the applications you develop. This optimism is unwarranted with file transfers. The litany of potential problems includes:

- Insufficient storage space on the target server
- Access control restrictions that prevent successful transfer
- High-latency networks that slow the transfer to unacceptable levels
- Security risks such as file tampering and information theft
- Change in requirements that necessitate modifications to overly specialized code

These problems are obvious, especially when they disrupt file transfer operations. Less obvious are problems that stem from poorly integrated file transfer processes.

File Transfer Is an Isolated Operation

This is the most ironic of all misunderstandings about file transfer. After all, why bother transferring files if they were not needed for some reason? Perhaps this is not so much a misunderstanding as an under appreciation. Consider, once a file is transferred, what happens to it next? Some things you need to consider at this point are:

- Is the file sufficiently secured in the directory to which it was transferred?
- Could another user accidently overwrite this file before it is copied, processed, or otherwise operated on?
- What process will remove the file from the directory when it is no longer needed?
- How will other processes be notified that a file has arrived and the next stages of processing should be initiated?

No file is an island unto itself. Once it is transferred, there are likely other operations that have to be performed. Even when post-transfer requirements are minimal, there will likely be some requirements for post-processing. For example, when a file is transferred for backup purposes, the event should be logged for management purposes.

The 7 Essential Requirements of Secure and Reliable File Transfer

To transfer files in a secure and reliable manner requires more than may be apparent at first. Once common misconceptions are dispelled, you can see that common file transfer requirements span organizations of all sizes. Seven of the most important are:

- Support for both internal and external information flows
- Support for multiple protocols
- File transfer security
- File transfer optimization
- Reliability and transaction control
- Alerts and process reporting
- Event processing and automation

These requirements address the way file transfer operations are used in today's businesses. They start with the business driver perspective—not a technical orientation that drives the myths we saw earlier. These requirements can be contrasted with those myths and misconceptions to see how the reality of business operations is served by these requirements and not by common misconceptions.

Myth	Reality	Relevant Requirement
File transfer is a simple technical operation	There is more to file transfer than copying bytes. Business requirements dictate the need for performance, security, and reliability across a range of applications—many with varying technical requirements.	Support for internal and external information flows Support for multiple protocols
There are few risks, potential errors, or changes to requirements	There are many ways a file transfer operation can fail. Private and sensitive information may be at risk; security controls are required.	Security File transfer optimization Reliability and transaction control
File transfer is an isolated operation	Files are transferred for a reason. Other processes may need to process the data transferred or log the fact that it was transferred.	Alerts and process reporting Event processing and automation.

Table 3.1: Common myths and misconceptions undermine business objectives with regards to file transfer. The seven essential requirement outlined here are meant to support those business objectives.

Essential Requirement 1: Support for Internal and External Information Flows

Internal and external information flows are different enough that you need to select file transfer solutions based on their ability to support both. The differences stem from the fact that with internal transfers, you are typically, but not always, working with shared identity management systems, common access controls, and transmissions over secure routes. Support for both internal and external information flows means there is support for:

- Common requirements
- Multiple access controls
- Encrypted transmissions

Common Requirements

The common requirements for internal and external information flows include:

- Sufficient performance
- Adequate logging of events in the transfer process
- Ability to schedule jobs
- Ability to recover from errors and restart transfer jobs

Sufficient Performance

Performance is a fundamental requirement for any IT service; file transfer is no different. The specific level of performance will vary by application, but in many cases, it is determined by windows of time in which files can be transferred and the number and volume of files that must be transferred in that time.

Adequate Logging of Events in the Transfer Process

File transfers can entail multiple events. Many of these are expected, such as initiation and termination of connections as well as the start and completion of a transfer. Others events are not unexpected but they are unwelcome. These include errors, such as insufficient storage space or overly restrictive access controls. They also include reports of failed services, such as a lost network connection that prematurely terminates a file transfer. All of these should be logged. Log information is especially useful for diagnosing problems with individual transfer operations, but they can also provide information on trends, such as increasing time required to complete transfers or increasing frequency of dropped packets. This type of information can be useful to track down emerging problems or plan for future capacity requirements.

Ability to Schedule Jobs

We often repeat the file transfers over and over again. Copy the database export to a report server that is used to off load reporting from the production server. Transfer backup files to the disaster recovery site. Upload data from branch offices to headquarters. Send business partners nightly updates to the product catalog. As business processes span departments within an organization as well as across organizational boundaries to business partners, there is the potential for more file transfers. To ensure these transfers occur as needed, a file transfers solution should provide for scheduling these tasks.

Ability to Recover from Errors and Restart Transfer Jobs

One thing about file transfers is virtually certain: If you do enough of them, you will eventually run into errors. This is true for both internal and external transfers. Of course, how you deal with them can be very different. For example, when an external server runs out of space, you cannot always submit a service desk ticket to get more space. (That may be true of internal processes as well). Once the problem is resolved, the transfer process should resume with minimal intervention. The file transfer system may be configured to attempt the transfer again at regular intervals, ideally starting for the point of failure and not restarting from the beginning of the file. Each attempt should be logged as well; the number of attempts and time required to resolve the problem is useful information.

Support for Multiple Access Controls

When transferring files across organizational boundaries, you have to support multiple forms of access controls, including different types of authentication. Many organizations can meet their security requirements using username and passwords to authenticate file transfer users. Others may have stricter controls and require that all users authenticating to a file transfer service use a digital certificate-based method for authentications.

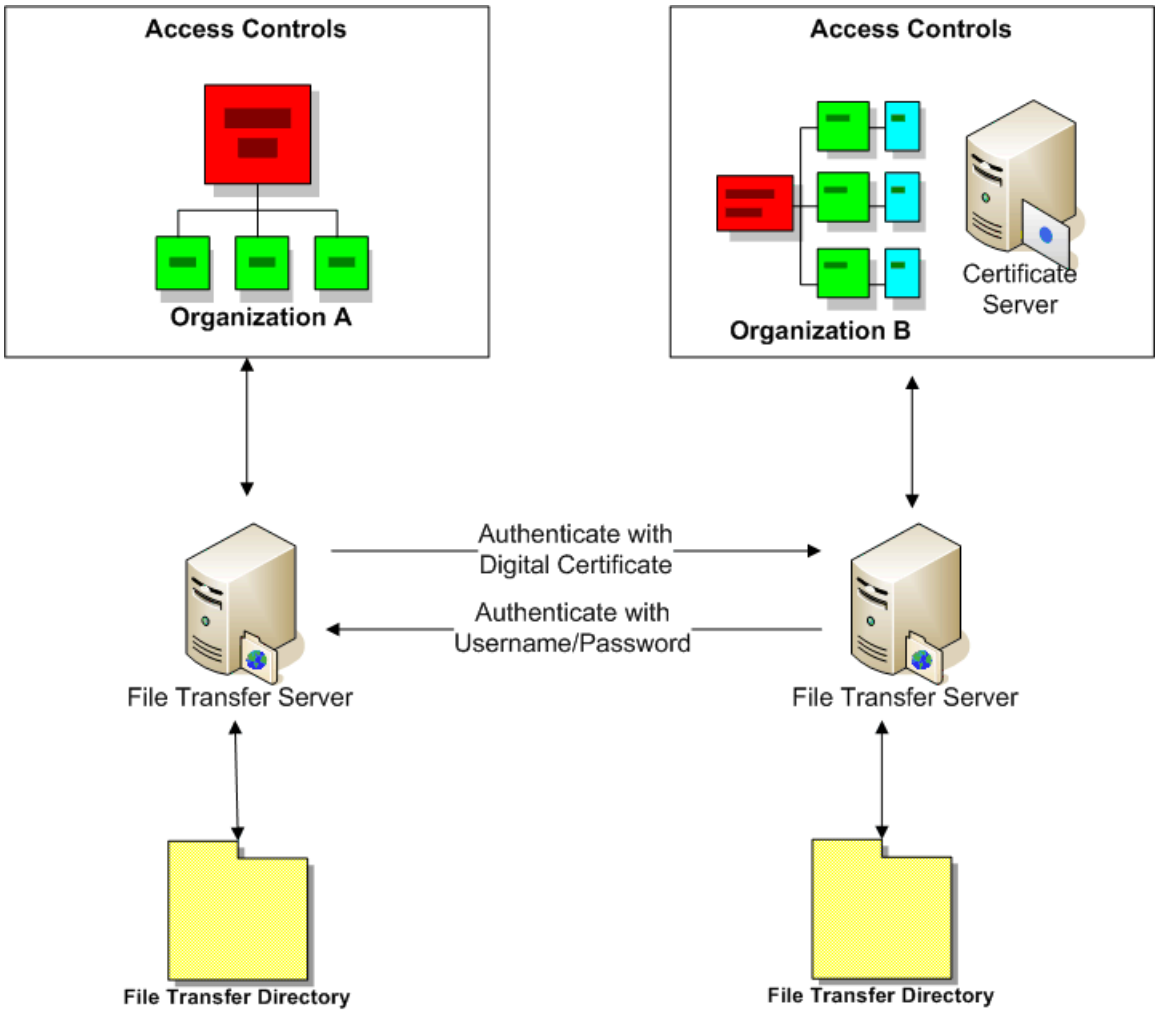

Figure 3.2: Different organizations may use different methods for authentication, such as username and passwords or digital certificates.

Support for Encrypted Transmission

Transferring files across the public Internet exposes them to threats such as tampering and divulging confidential information. To mitigate the risk from these threats, file transfers should be encrypted. There are several ways this can occur, although they all rely on the same basic encryption technologies:

- Encrypting each file individually as part of the file transfer process
- Using an encrypted file transfer protocols, such as SFTP, which encrypts all traffic related to the file transfer
- Using a virtual private network (VPN), which encrypts all traffic, including file transfers

When evaluating encryption applications, make sure the applications are using sufficiently strong encryption. Encryption strength is something of a moving target. An old encryption standard, DES, was once considered state of the art; today, DES encryption could be easily broken. Look for encryption programs that are Federal Information Processing Standard (FIPS) verified; use strong encryption algorithms, such as the Advanced Encryption Standard (AES); and employ sufficiently long encryption keys.

At this point, we are starting to move from the high-level requirement of supporting both internal and external information flows to discussing lower-level implementation details. This brings us to the second essential requirement, the ability to support multiple protocols.

Essential Requirement 2: Support for Multiple Protocols

There is an old saying in the Perl community, "There is more than one way to do it." The sentiment reflects the fact that there are often many ways to use the tools and techniques available to us to accomplish a specific task. That sentiment applies equally well to file transfers. Take, for example, the number of protocols available for file transfer. There are a number of open standards as well as proprietary protocols that may offer added benefits, such as accelerated file transfer.

The flip side of having many protocols to choose from is that your transfer partners may use any one or more of these protocols. That, in turn, requires you to support what your trading partners support. Let's look at the variety of protocols and the advantages and disadvantages they present.

Open Standard Protocols

As a general rule, open standards mean greater interoperability than proprietary standards. When it comes to file transfers, there are several protocols to choose from; some are relatively old (at least relative to the age of the Internet) and others are more recent advances with improved features.

The most important protocols for file transfer include:

- FTP—Dating from 1971, this is the earliest file transfer protocol that provided a basic client/server model for file exchange. The protocol has been updated since then with the latest version defined in RFC 959 adopted in 1985.
- SFTP—Secure File Transfer Protocol brings basic file operations to the Secure Shell Protocol (SSH). The underlying SSH protocol provides security features, such as authentication and encryption.
- FTPS—This protocol builds on FTP by adding security features to the protocol with the use of Secure Sockets Layer (SSL) and Transport Layer Security (TLS).
- HTTP—The Hypertext Transfer Protocol is best known for transferring Web page content but has been used for basic file transfer.
- HTTPS—HTTP over SSL provides the same basic functionality as HTTP but adds the security features of SSL, such as encryption, as well.
- AS2—Formerly known as Applicability Statement 2, the AS2 protocol builds on the secure mail protocol S/MIME to incorporate encryption and digital signing to files that are transferred as "attachments." AS2 also provides for message disposition notifications, which can indicate the status of the message and file after reaching the recipient.

It is worth noting that file transfer protocols are adapting to meet the essential requirements of complex business operations by incorporating security and notification services. They do this, in part, by employing file transfer protocols with other existing protocols.

Proprietary Protocols

FTP is designed to use TCP for transmission. TCP is a general purpose transmission protocol that ensures reliable, in-order delivery of data packets. Web pages and emails are sent over TCP as well. TCP is a general solution that can meet a wide range of requirements. A drawback of TCP, though, is that the exchange of control information between a client and a server can impose substantial overhead on a transmission.

Some managed file transfer vendors implement proprietary techniques to improve performance of file transfers. For example, a vendor may use UDP and implement additional functionality on the client and server to ensure reliable, in-sequence transfer. As with any proprietary solution, the client and server will have to agree on use of this protocol and possibly install proprietary software. Another potential advantage is that vendors may incorporate other features, such as compression, that can further improve overall performance.

Anyone who has lived through complex application development and deployment efforts that have been hampered by proprietary protocols may have something of a knee-jerk reaction against proprietary protocols. For example, a proprietary protocol within a company may function without a problem because all parties are using the same protocol; however, when you try to use the same protocol with a business partner that does not support that protocol, problems arise.

You just need to remember that open protocols are valued for the way they allow you to meet business requirements; there is nothing inherently valuable about an open protocol. It is a means to an end. Similarly, there is nothing inherently wrong with a proprietary protocol; but it will have to prove itself. That is, whatever the proprietary protocol has to offer must outweigh the lost flexibility, interoperability, and other benefits of open standards. These benefits can be in the form of accelerated file transfer, improved recovery after an error in transmission, greater control over the transfer process, or some other improvement to the overall process.

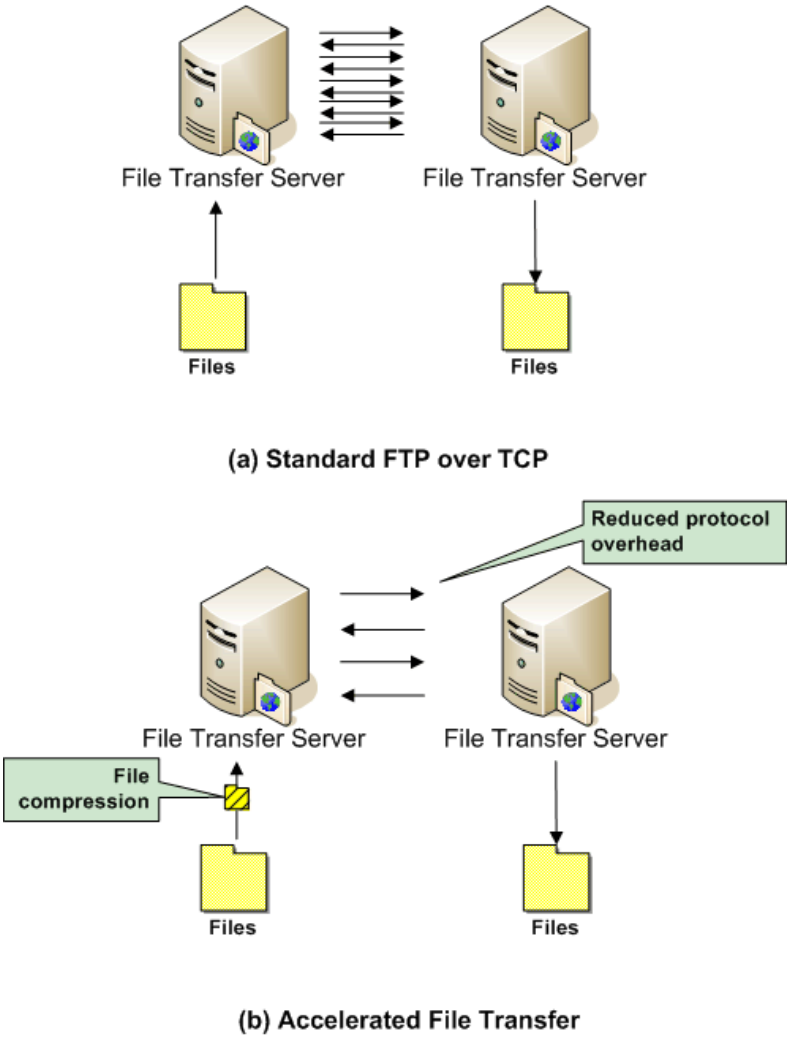

Figure 3.3: Proprietary protocols can accelerate file transfer through a combination of compression and reduced overhead in the transmission protocol.

Essential Requirement 3: File Transfer Security

Data managed by applications, stored in database, and stored in user directories can be protected with a variety of access controls. When data is in transit from one system to another, there are risks. In particular, confidential or other protected data is at risk if it is staged in the DMZ and if it is transmitted in unencrypted form.

Staging Data for Transfer

Staging areas are shared resources where you temporarily store files so that they can be transferred to the final target server. By definition, multiple users put files into the staging area. This typically implies fairly open access controls; at the very least, many different users can write to the staging area.

In some cases, you might be able to treat the staging area as accessed by a trap door: you can write data but cannot read it. Data goes in one way but cannot be accessed in the same way. This helps reduce the risk of another staging area user reading your files, but this setup creates other difficulties. For example, without read access, you cannot readily confirm the file was written correctly by calculating a checksum on the written file. You do have other methods for protecting confidential data, though.

Confidential information should be encrypted before it is copied to a shared staging area. In addition, files should be deleted from the staging area after they are transferred to their target locations. In the event that a file is not successfully transferred after some reasonable period of time or number of attempts, the file should be purged from the staging area. This will prevent a file from remaining in a staging area for extended periods of time because of errors or problems with the target server. This is especially important when the target server is controlled by another party that might not have the same level of concern about your confidential information sitting in a shared staging area.

File Transfer Security in the DMZ

The DMZ part of a network is semi-trusted. The DMZ is protected by a firewall, which offers basic security controls such as blocking ports to unused services. At one time, that actually provided a fair degree of protection. Today, however, such is not the case due to the amount of traffic that moves into and out of the DMZ over HTTP, including security threats. In general, it is not reasonable to assume confidential data is protected in the DMZ. When compliance is a concern, confidential and proprietary data should not be stored in the DMZ. In fact, a good rule of thumb is to not store data for file transfer in the DMZ, period.

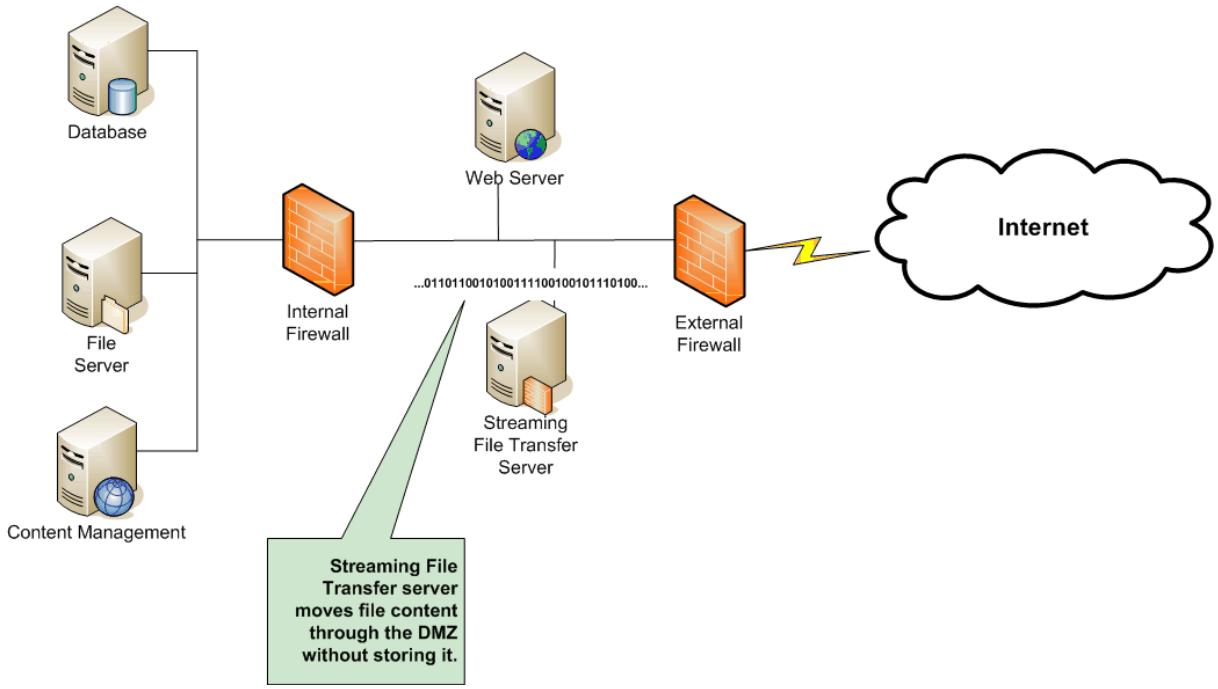

Figure 3.4: Files are vulnerable to compromise if stored in the DMZ. File streaming moves file data immediately through the DMZ without exposing it to risks associated with persistent storage in the DMZ.

The alternative to streaming is a store and forward model. Consider a simple example. An insurance company and a hospital have established a procedure for transferring patient billing information on a daily basis. The hospital transfers patient and procedure information to the insurance company, and the insurance company provides coverage and payment information in return. The hospital has a DMZ and an internal firewall that blocks any direct access to the trusted network by any outsider, including business partners. To enable file sharing, the hospital IT department decides to set up an FTP server in the DMZ. Outsiders, such as the insurance company, can access the ftp server, pick up patient and procedure information, and leave payment and coverage information. In an ideal world, this may work, but not in our world.

This store and forward approach leaves confidential data vulnerable to a number of security weaknesses with FTP. By leaving data in the DMZ, any attacker with the ability to tunnel threats over HTTP can get into the DMZ, access the Web server, and then exploit vulnerabilities on that server to gain access to other systems in the DMZ, possibly the file transfer server. Streaming data through the DMZ does not sacrifice the advantages of having a DMZ, but it avoids the risks associated with storing data there as well.

File Transfer to the Cloud

Data transfers to the cloud also require attention to security concerns. The level of security provided by your network can vary by the way you use the cloud. Let's consider a couple of different scenarios.

In one scenario, you use public clouds on occasions when you need a large number of servers to analyze or process a large amount of data. This situation can take place when you are doing a market analysis study and you need to run data mining and statistical analysis programs on years of customer transaction data and third-party demographic data. For occasional uses like this, you might transfer data from your on-premise servers to cloud storage. You are responsible for securing communications between your on-premise servers and the cloud. You should ensure authentication information is protected and that confidential and private data is encrypted before it is transferred.

In the second scenario, your company uses cloud servers almost continuously. You treat the cloud as an extension of your internal resources. In this case, you might implement a virtual private network (VPN) between your corporate network and the cloud provider's network. File transfers are encrypted by the VPN software, and application-specific encryption is not needed.

A well-designed file transfer solution should accommodate either scenario; custom scripts that assume a secure network might not adequately protect confidential data when used in the first scenario.

Essential Requirement 4: File Transfer Optimization

As file sizes and volumes of transfers grow, it is imperative that a file transfer solution be able to optimize transfer operations. Optimization in this context can be thought of in terms of automation and efficiency.

File Transfer Automation

File transfers should require minimal manual intervention. This automation is achieved through a combination of robust scripting and scheduling features and policy definitions governing the transfer process.

A robust scripting environment is required in file transfer operations. (This is probably one of the reasons Perl is such a popular choice for ad hoc file transfer programs). Scripts allow you to collect the files needed in a transfer set, perform pre-processing operations, and respond to events in a file transfer workflow. In cases where there are frequently required operations, such as transfer all files in a directory or transferring all files with names that match a regular expression, the file transfer system may provide that as a standard operation.

Scripts can become complex when automating file transfers to the cloud. You pay for cloud computing resources based on the time you run virtual machine instances, so you will want to minimize the time your virtual machines are waiting for files. You might, for example, script your file transfers in a way that triggers another script to start virtual machines in the cloud only after the transfer is complete. Ideally, a file transfer solution will support workflows that allow for scripts to perform tasks in additional to transferring files.

Policies are templates for operations and workflows that can be applied in multiple instances. For example, the sales department may need to upload daily sales data to the data warehouse every night. The data is actually distributed over a number of geographically disperse servers, but they all use the same file structures to store the data intended for the data warehouse. A policy could be defined to transfer files based on the current date found on the files in the staging directory on each server. The policy, which may be implemented as a script or a configuration file executed by a script, may include commands to trigger other events once the file transfer is complete, such as starting a load process in the data warehouse.

Automation reduces the need for manual support in file transfer operations. This opens the possibility of performing more and more file transfers. That is, of course, unless the file transfers system cannot keep up the volume of data.

Efficiency of File Transfers

The efficiency of a file transfer is a measure of how much data can be transferred in a given amount of time. One obvious way to improve this metric is to increase the bandwidth of the network and improve the I/O throughput of servers; however, we are more concerned with the file transfer software and processes than the infrastructure they run on. That leaves two options to improve throughput of file transfers: reduce the amount of data transferred and reduce the overhead associated with each transfer.

Reducing volumes can be done with compression. Standard compression algorithms for data transfers do not lose information but reduce the size of the representation required to encode that information.

Reducing the overhead associated with the transfer can be done with accelerated file transfer protocols. As noted earlier, these proprietary protocols use different techniques than those used in TCP to transfer data while still ensuring delivery of all data in the proper order.

Accelerated file transfer protocols are especially important to transfer large files to the cloud or to business partners. Big data analytics is a promising evolution of business intelligence practices; however, it can require significant computing resources. Public cloud providers can readily offer the computing resources needed, but you have to get your massive data sets to the cloud first. Optimized file transfer can help reduce a barrier to entry to big data analytics by making public cloud providers a viable option for computing resources.

Essential Requirement 5: Reliability and Transaction Control

Additional key requirements for file transfer solutions are reliability and transaction control. The two are closely related, so we will consider them together.

Reliability pertains to the ability to transfer files consistently over time. It also implies a degree of robustness; simple errors should not completely disrupt or disable the transfer process. A reliable file transfer system will be able to handle a range of operating conditions and requirements:

- Transferring large files
- Transferring a large number of files
- Completing transfers under adverse network conditions, such as high latency
- Recovering from disrupting errors, such as lost connections

Some of these requirements are met in part by other features, such as accelerated file transfer that contributes to transferring large files and operating on high-latency networks. Others, such as recovering from disruptive errors, require additional functionality.

One way to implement a recovery mechanism is to use checkpoints. Checkpoints are control mechanism used to ensure the transfer process up to a point has been successful. Metadata about the transfer process and the state of the transfer is also recorded. In the event of an error, the file transfer system will not have to start the transfer all over again from the beginning; instead, the process can restart from the last checkpoint. Checkpoints are essentially the last known good state of the transfer.

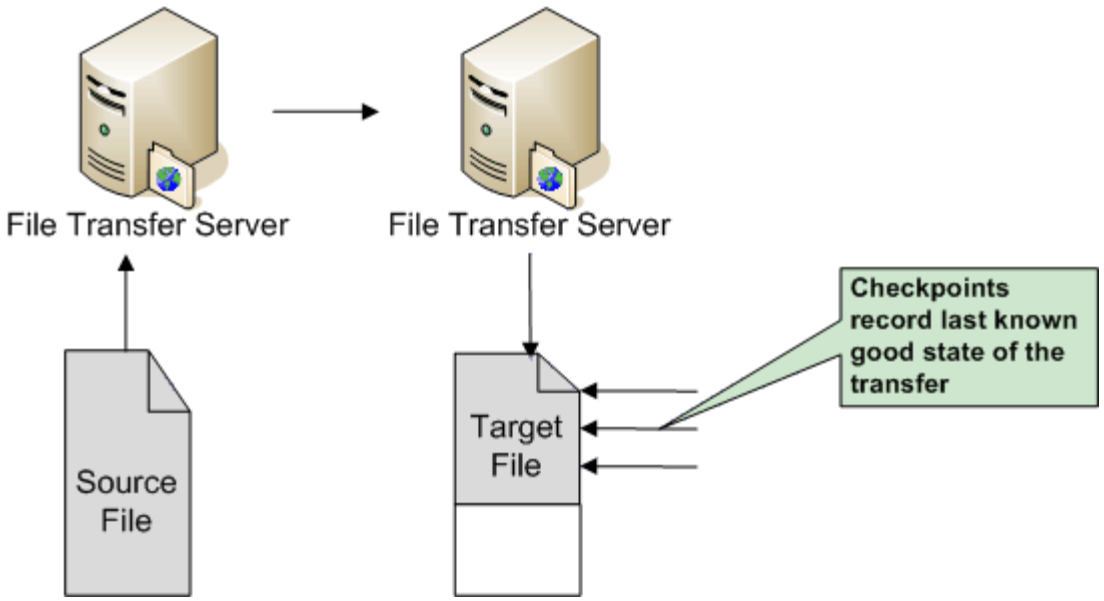

Figure 3.5: Checkpoints are metadata structures that record information about the file transfer process that allow a disrupted transfer to resume from the latest checkpoint rather than starting from the beginning of the file.

Essential Requirement 6: Alerts and Process Reporting

It is not enough to reliably and securely transfer files between systems, you must be able to manage the process as well. This brings us to the sixth essential requirement: the need for alerts and process reporting. For the purposes of this discussion, we distinguish alerts, which are notifications of some unexpected event, from process reports, which are reports about the status of normal operations and activities.

File Transfer Alerts

We have discussed in some detail the various ways things can go wrong with file transfers and the need to handle these situations. One of the elements of effective error management is reporting when a problem occurs. A problem report should include enough information for a systems administrator to understand which file transfer is involved and what the technical aspects of the problem are (at least to some degree). This is accomplished by including information such as the following in error reports:

- Name of the job that failed
- Severity level
- Name and size of the file that was in the process of transferring when the failure occurred
- The source and target destinations of the file transfer
- Usernames of authenticated users on both sides of the transfer
- Summary of error message returned by subsystem that detected the failure, such as the operating system (OS), Java runtime environment, and so on

Note that alerts do not have to contain all relevant information about an error, just enough to make the systems administrator aware of the problem, its severity, and a basic description. Alerts can be delivered on a number of platforms, such as email and text messages, so you must be careful not to overload an administrator's cell phone with a message too long to read on a mid-range mobile device. Details of the failure are best tracked in log files that can be analyzed in more detail as needed.

Process Reporting

Alerts tend to be bad news; process reporting is where you get the good news, and occasionally some bad news. Process reporting provides details to administrators so that they can (1) ensure operations are completing as expected on a day-to-day basis and (2) track trends over longer periods. The level of detail can vary in process reports. In some cases, a simple message that a transfer is complete is enough. In other cases, administrators will want detailed reports about all transfers over a given period of time. These may include details such as:

- The number of transfers in and out
- Total volume of data transferred in and out
- Time required to complete transfers
- Breakdown of transfers by time, department, transfer partner, and so on
- Number of errors by severity level
- Number of restarts due to errors in transfer
- Percentage of successful restarts

Process reporting is essential for any business operation and file transfers are no different.

Essential Requirement 7: Event Processing

The last of the essential requirements we consider is event processing. Transferring a file from one system to another is often a multi-step process. Furthermore, it is not uncommon for a single file transfer job to include multiple files. You can roughly divide event processing into two types: pre-processing activity and post-processing activity.

Pre-processing activities are those required to prepare files for transfer. These activities include tasks such as:

- Collecting files and copying them to a staging area
- Verifying the structure and content of files
- Applying basic transformations on files, such as reformatting to meet the needs of the target system
- Logging details about file preparations
- Encrypting files prior to transfer
- Compressing files prior to transfer

Post-processing events typically trigger tasks that make use of the recently transferred files and include:

- Triggering the next processes in the workflows, such as importing files into application databases
- Logging data about the size and number of files transferred
- Copying files from staging areas to long-term storage directories with appropriate access controls
- Decrypting files
- Uncompressing files
- Generating process reports

Pre- and post-processing operations can be complex, especially when dealing with data imports and exports that require extensive data quality checks, error reporting, or trigger processing and analysis workflows in the cloud. A file transfer solution should include sufficient support for process automation to meet the pre- and post-processing needs of the organization.

Summary

File transfers are used many ways, but the essential requirements outlined here are broadly applicable. To summarize, the seven essential requirements are:

- Support for both internal and external information flows
- Support for multiple protocols
- File transfer security
- File transfer optimization
- Reliability and transaction control
- Alerts and process reporting
- Event processing and automation

These requirements grew out of the way file transfers are typically done in businesses and other organizations. File transfers are common operations in more elaborate workflows and involve sensitive and confidential data that must be protected. The file transfer processes themselves are often part of mission-critical operations and therefore must be secure, reliable, scalable, and manageable. The seven essential requirements translate these high-level requirements into functional requirements that should be available in enterprise-quality file transfer solutions.

Chapter 4: Planning and Deploying a File Transfer Solution

Throughout *The Shortcut Guide to Eliminating Insecure and Unreliable File Transfer Methods,* we have considered the limitations of traditional file transfer methods, analyzed key business drivers motivating improvements to file transfer methods, and assessed essential requirements for a secure and reliable file transfer solution. In the final chapter of this guide, we will turn our attention to planning and deploying a secure and reliable file transfer solution. We continue our methodical approach to understanding file transfer issues and solutions with a structured approach to deployment.

The chapter is organized into several sections, each describing a core step in the deployment process:

- Assessing the current state of file transfer methods used in an organization
- Identifying business requirements for a file transfer solution
- Inventorying hardware and assessing repurpose potential
- Prioritizing replacement of existing solutions with an enterprise file transfer solution
- Establishing policies and procedures for enterprise file transfer
- Rolling out a file transfer solution across the enterprise

By the end of this chapter, the reader will have concise description of critical questions that should be answered, methods for assessing the answer to those questions, and guidelines for constructing policies and procedures to promote the most efficient and effective use of an enterprise file transfer solution.

Assessing Current State of File Transfer Methods

Once you have determined that your organization would be better served with a secure, reliable file transfer solution, the next step is to catalog existing methods for file transfer and business processes dependent on those methods. The goal at this stage of the process is not to change any of the existing processes; we simply want to understand the extent to which ad hoc file transfer solutions are used and where they are used.

Identifying Homegrown File Transfer Solutions

The difficulty in identifying where in an organization homegrown file transfer systems are used can range from fairly simple to time consuming and complex. There are two basic strategies for identifying homegrown file transfer solutions: a top-down approach and bottom-up method. They are not mutually exclusive and often we can draw from both. The top-down approach is simple and direct; it takes advantage of work done in the process of formally managing software assets. The bottom-up method works for environments without centralized and detailed information about software deployments.

It is worth noting that even in organizations with formal software asset management, there might be custom file transfer solutions. Such is especially the case when departments or other small groups can use public cloud providers for special projects, like an analytics project or new business intelligence initiative. These kinds of projects can emerge outside of normal project management and software development procedures, and they can be particularly difficult to track down.

Working Top Down with Formal Enterprise Software Management: The Best of All Possible Worlds

At one end of the spectrum, we have organizations with formal software management policies that are strictly followed. In these ideal organizations, all production software is managed in code repositories. Approved copies of commercial software are maintained in a central repository along with tested and validated patches for that software. Copies of the software are deployed from the repository, information about the deployment is maintained, and one can readily report on where the software is deployed, which version is in use, and similar management-related data. Custom applications are similarly managed with the additional benefit of having copies of source code.

In this ideal situation, assessing the use of homegrown solutions becomes a matter of reviewing the contents of the software repository. Custom applications will likely have descriptions and other metadata that indicate whether file transfers are part of the services provided by the software. One should search for terms such as:

- ftp, ftps, and sftp
- extraction, transformation, and load (ETL)
- extract, dump, and backup
- replicate and synchronize
- file transfer, file copy, and so on

Although we are concerned here with "homegrown" solutions, we should not ignore commercial software. For example, a commercial ETL tool may have an ftp module that is used in ETL scripts to transfer files. The software repository may not have all scripts currently in production, but it will at least point to business processes and application users that might have developed such scripts.

Distributed Management Models and the Bottom-Up Approach

Not every organization has an enterprise-wide formal software management process. This is not necessarily a problem; different businesses have different drivers and requirements. A multinational bank might not hesitate to spend the time and resources required to maintain a formal software management system. A rapidly growing startup, in contrast, might have to focus on more immediate demands on its time. Many organizations will likely fall somewhere in between where the desire to adopt ideal software engineering practices are tempered by constraints of time and budget.

The bottom-up approach starts with low-level details of file transfer implementations. Let's assume a worst-case scenario in which we are working with an organization that has no structured information on the use of file transfer programs. The company has grown quickly and had to focus on delivering IT services on tight schedules. Corners were cut and documentation is limited. The organization's drive to deploy services was successful and IT management, systems administrators, and developers are all in agreement that the quick and dirty homegrown programs for file transfer need to be replaced.

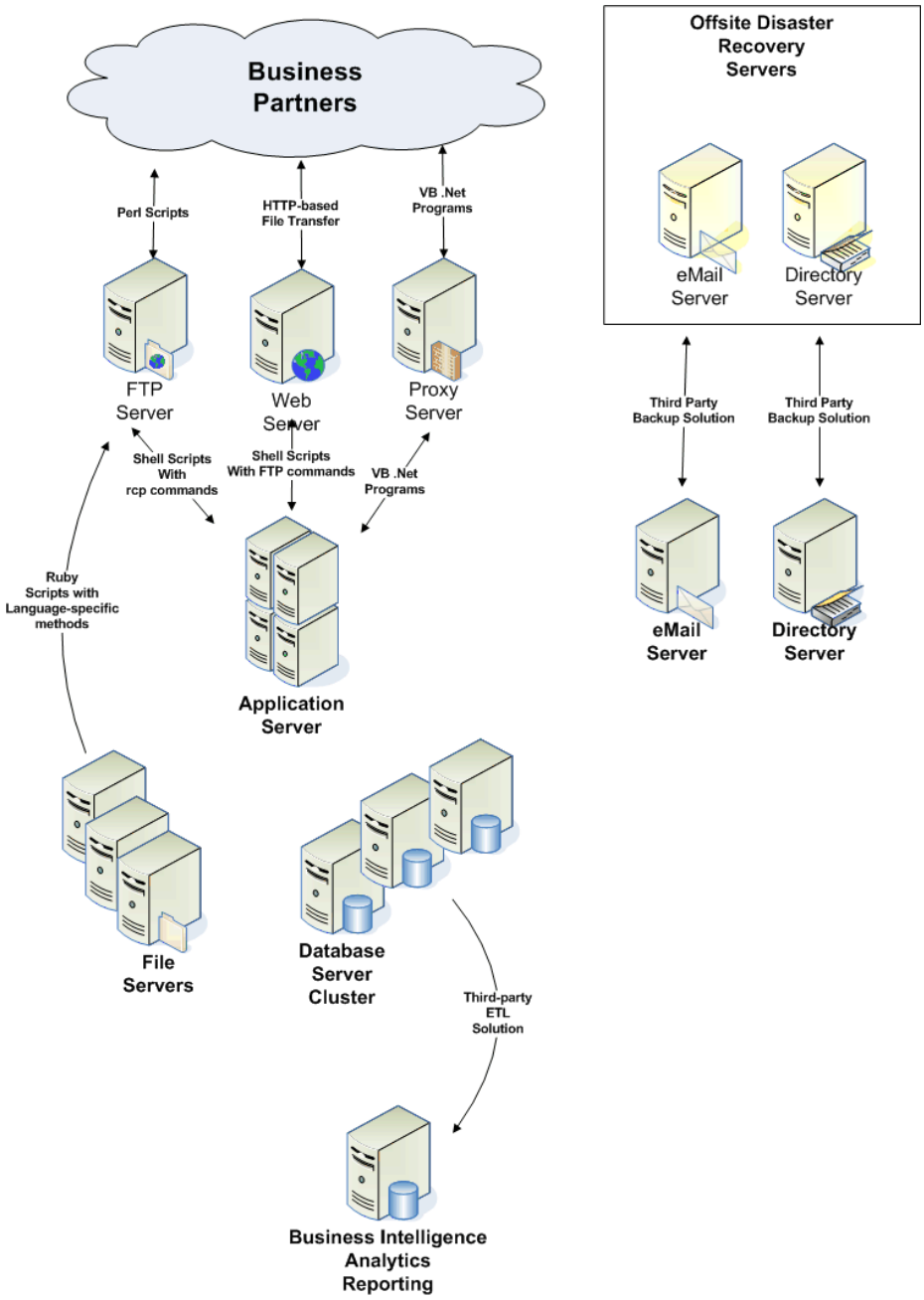

Figure 4.1: Different methods of file transfer may be used throughout an enterprise when each file transfer process is treated as a new requirement rather than an instance of a generalized problem.

The bottom-up approach begins with searching servers for scripts used to transfer files. There are many ways to transfer files (see Figure 4.1 for some examples), so there will be a variety of patterns that one should look for when searching code directories. Some example patterns indicative of file transfers include:

- Perl scripts that uses the module Net::FTP
- Python scripts that use the low-level sockets module
- Ruby scripts that use the FileUtils module, and the copy method in particular
- Visual Basic programs referencing the System.IO object, especially those that use the FileExists and FileCopy methods
- Unix/Linux shell scripts using the remote copy (rcp) or ftp commands

This is not an exhaustive list, but it does give an indication of the range of ways in which we can implement homegrown file transfer solutions. In addition, one should search for scripts or configuration files for ETL or backup programs that may be used to transfer files created specific to those operations.

> **Which Search Tool?**
>
> Window's file searching service provides basic search functionality and is suitable for simple search requirements. Linux and Unix users have the advantage of the grep command and its support for regular expressions. For a little bit of Unix/Linux in a Windows environment, use the Cygwin suite of tools (http://www.cygwin.com/), which offers a Linux-like command-line environment for Windows users.

Of course, this method assumes we have read access to all production code directories. In practice, these search scripts will likely be run by several systems administrators and application managers who would have appropriate access to the various servers.

In the case of small groups using public cloud providers, you might find an ally in the finance department. Cloud usage will leave a trail of transactions. If payments are made to cloud providers through invoicing, purchase orders, or other company payment processes, then they should be relatively easy to track down. If small payments are made on credit cards, you will need access to all the transactions on those cards to find all cloud users.

Gathering information about homegrown file transfer solutions, whether it is done in a top-down, bottom-up, or combination method is the first step in the assessment phase. The next step is assessing business processes dependent on homegrown solutions.

Inventory Business Process Dependencies on Homegrown File Transfer Solutions

Every file transfer program in production use is running to support one or more business processes. For each of these programs, we should document key characteristics of the business process, including:

- Whether it is an internal or external transfer
- How frequently the transfer is executed
- The number of files typically transferred
- The volume of data typically transferred
- Constraints on when the program can execute

We should also categorize the business processes in terms of how important the process is to overall business operations. For example, file transfers from an order entry system to an order fulfillment application are critical. File transfers from an enterprise resource planning (ERP) system to a data warehouse are important but less critical than customers' orders. File transfers to support updates to news stories on the internal employee portal fall into the least important category. This information is useful for planning migration from existing solutions to an enterprise file transfer system.

At this point, we have information about what homegrown solutions are in use and what business processes are supported by them. Next, we need to assess the actual business requirements of the business processes using file transfer. As noted earlier in this guide, homegrown solutions may not actually meet all the business requirements of business operations. It would be a mistake to analyze existing file transfer programs and assume we have a handle on the business requirements.

Identifying Business Requirements for File Transfer Solutions

It is safe to assume that even the most rudimentary homegrown file transfer solution meets some business requirements. At the very least, such programs copy files from one device to another. Usually there is more functionality provided as well. We can often find information about other requirements within existing solutions, specifically:

- Information about programs that invoke the file transfer program
- Scheduling details, such as cron job entries that define when the programs are run
- Information about programs that are run after the transfer is complete
- Details about file transfer processes that are recorded in log files
- Authentication requirements for running the file transfer program

There may be additional facts about file transfer operations that are not apparent in the scripts and supporting code that implement homegrown solutions. For these details, we need to research the underlying business processes that might be either internal or external processes.

Internal file exchanges are less complex than external ones with regards to management and oversight because all responsible parties are within a single organization. In many cases, the same authentication and authorization systems are used for both the file transfer source and target systems. There may be greater latitude with regards to access controls as well. For example, an internal file transfer process may use an ftp server configured to allow reading as well as writing, thus allowing users to list the contents of the ftp directory and confirm the file transfer completed successfully. Transfer partners that receive files from multiple external sources in the same ftp directory may be hesitant to give outsiders, even business partners, read access to a directory that is shared with others.

Big data analytics will drive additional requirements for file transfer solutions. Those kinds of projects can drive processes that work with a wide range of files, such as log files, data base extracts, and third-party data sets, such as demographic information. Be prepared for potentially large file transfers to support analytics efforts. Also consider how frequently the file transfer processes will need to run. If you are collecting data from a heavily used Web application, you might want to download application logs every few hours. When you are downloading third-party demographics data, you might find monthly updates are sufficient. Optimizing file transfers will depend, in part, on understanding the frequency with which various jobs need to be run.

When assessing requirements for both internal and external file transfers, consider the following:

- Number of files and volume of data transferred
- Bandwidth required to complete transfers in time window allotted
- Retention policies on files transferred
- Need for encryption, both during transfer and while the files reside in ftp sites or other staging areas

In the case of external file transfers, one must also look into authentication and authorization issues. In cases where transfer partners have close relationships, there may be trust agreements between the organizations that allow identities on one of the partner's systems to be trusted by the other partner.

> **Note**
>
> These are known as federated identity management systems; they come with their own host of complex management challenges.

In other cases, a trading partner may simply embed a username and password in a transfer script. Although this is clearly a security risk, it is the kind of shortcut that is sometimes taken in homegrown file transfer solutions. An important requirement for any enterprise-scale file transfer solution is the ability to securely manage user credentials, such as usernames and passwords or digital certificates.

Existing file transfer programs are a starting point for quickly assessing basic file transfer requirements; however, they should not be the sole source. These programs likely do not capture the full extent of business requirements, so it is important to consider the broader business processes that include the execution of these programs.

Inventory Hardware and Assess Repurpose Potential

A fragmented approach to file transfers can lead to a fragmented infrastructure when dedicated file transfer servers are used. It can also promote virtual server sprawl in a virtualized environment.

Multiple Dedicated File Transfer Servers

Consider a typical project involving a multi-tiered architecture. Servers are needed to host a Web server, another to support an application server, and yet another for database services. The project also requires file transfer between the application server and a business partner's server. The system architect on the project is concerned about security vulnerabilities that have been found in ftp software and decides to host the file transfer application on a separate server in the DMZ (see Figure 4.2).

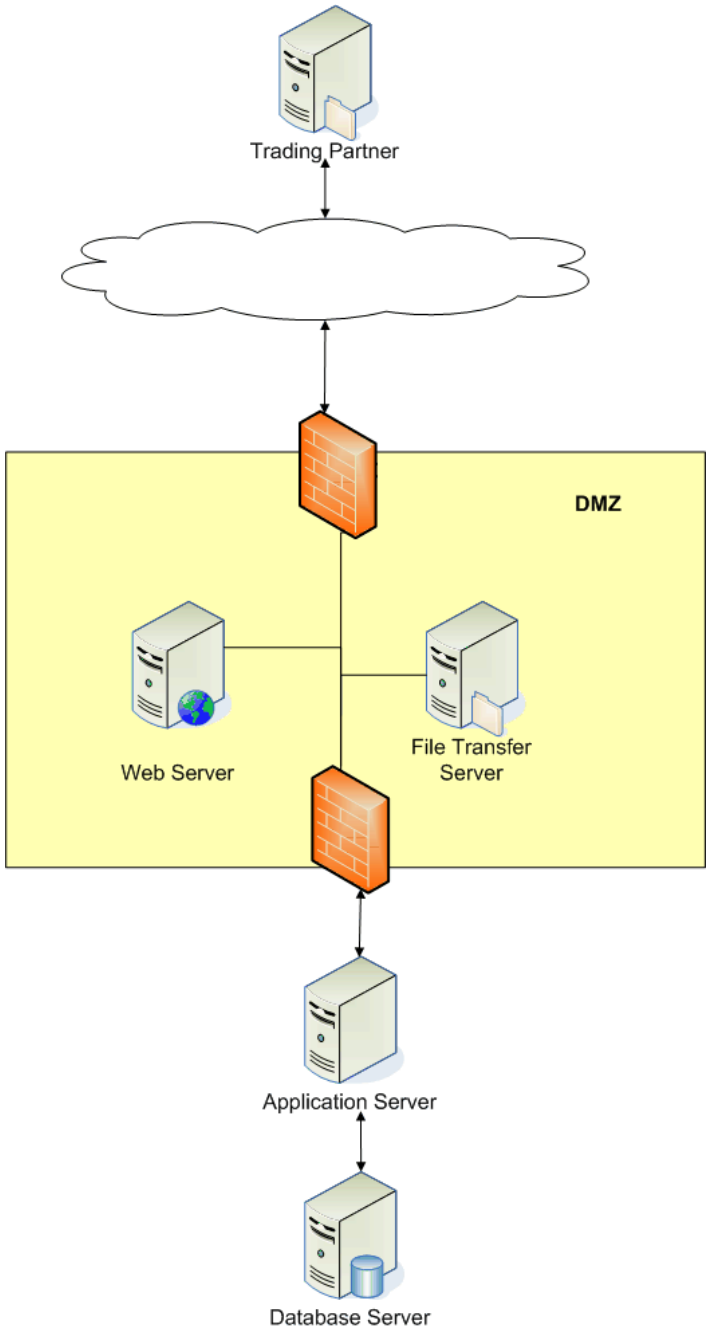

Figure 4.2: A single project solution to file transfer results in inefficient use of hardware.

If we multiply this scenario by the number of homegrown solutions in use within an organization, we can see the potential for reclaiming underutilized servers. Repurposing can directly benefit the business processes that use these homegrown solutions by allowing dedicated file transfer servers to be used for a number of alternative purposes:

- An additional server in an application server cluster
- An additional server in a federated database system
- An additional server in a Web server load-balance configuration
- A failover server in a disaster recovery situation

Consolidation can also benefit virtualized environments.

Countering Virtual Server Sprawl

Virtual servers offer many advantages of physical servers without the dedicated hardware. This makes virtual servers an ideal solution for isolating file transfer operations. The security concerns with ftp described in the previous section could be addressed by deploying a virtual server. In fact, one way to deal with the need for multiple file transfer methods (for example, ftp, sft, ftps, and https) is to host multiple virtual servers.

Another potential advantage of virtual servers is that a single file transfer server could be deployed with multiple virtual servers each of which is dedicated to a different business process. This configuration has several advantages:

- Different file transfer protocols may be configured on each virtual machine
- Each virtual machine can be administered by a different business process owner
- Access controls can be configured according to the needs of each business process
- Virtual machines can be configured with only the application stack needed by each business process, such as Perl, Python, and Ruby interpreters

The drawback of this approach is the additional management overhead that comes with deploying multiple virtual machines.

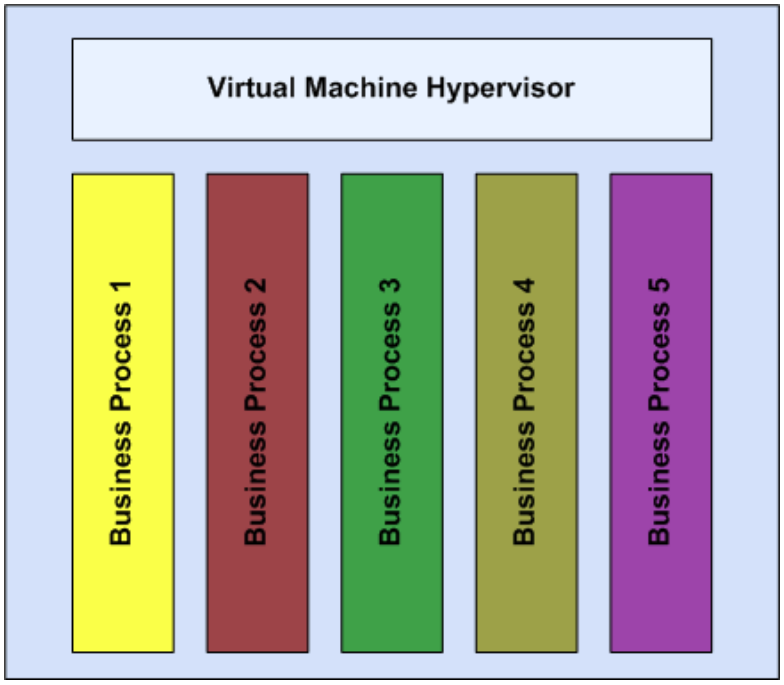

Figure 4.3: Deploying dedicated file transfer servers on virtual machines reduces the number of dedicated servers but maintains the management overhead of multiple servers.

Enterprise file transfer solutions can dramatically improve on these two scenarios. Rather than force architects to implement and enforce different file transfer policies and management schemes at the server level, enterprise file transfer solutions allow system designers to define logical policies and workflows that execute within a single file transfer application. There is no need for dedicated multiple servers or virtual server sprawl because file transfer processes can be implemented directly as a manageable, logical entity. The first step in the move away from fragmented architecture and collections of homegrown solutions is to prioritize the replacement of existing programs.

Prioritizing Replacement of Existing Solutions with an Enterprise File Transfer Solution

When faced with a wide array of existing homegrown solutions, it can be difficult to know where to begin with the migration to an enterprise file transfer solution. In such cases, it helps to consider several factors when prioritizing the move to a centralized solution. These factors include:

- Criticality of business process
- Frequency of transfer
- Volume of transfers
- Reliability of existing transfer solution
- Security requirements
- Compliance requirements

When considering these factors, the foremost consideration should be how well the existing file transfer solutions meet the business needs embodied in the factor.

Criticality of Business Process

File transfer processes are not created equal. The impact of a failed file transfer can range from a minor inconvenience to a costly one-time mistake to a long-term setback. Consider a few examples.

A business provides an employee portal for self-service human resources administration. Employees can perform routine tasks, such as submitting time cards, changing payroll deductions, and keeping up with company news. The portal administrator has established a simple mechanism for department managers to submit news to the employee portal. Managers email content to a designated email address, and a custom scripts extracts those emails, restructures them into an RSS feed format, and transfers them to the portal server. It's not difficult to imagine how such a script could fail: error message formats, changes to access controls on target directories, servers down for maintenance, and so on. Assuming the complete failure of the transfer script, there is still minimal impact on business. Customers are not adversely affected, business partners would not know of the minor glitch, and the business would not lose money due to the failure.

Now consider a more costly example. US businesses with sufficiently large payrolls are required to submit payroll taxes on a weekly basis. Companies that grow through mergers and acquisitions may find themselves continuing to operate multiple payroll systems for some parts of the payroll process while using a consolidated reporting system for tax purposes. Information about payroll taxes are collected from division-level payroll systems and a single payment is made based on the aggregated information. If a file transfer process fails or succeeds in transferring only partial information, there could be an error in tax payment. Penalties for such errors can be significant.

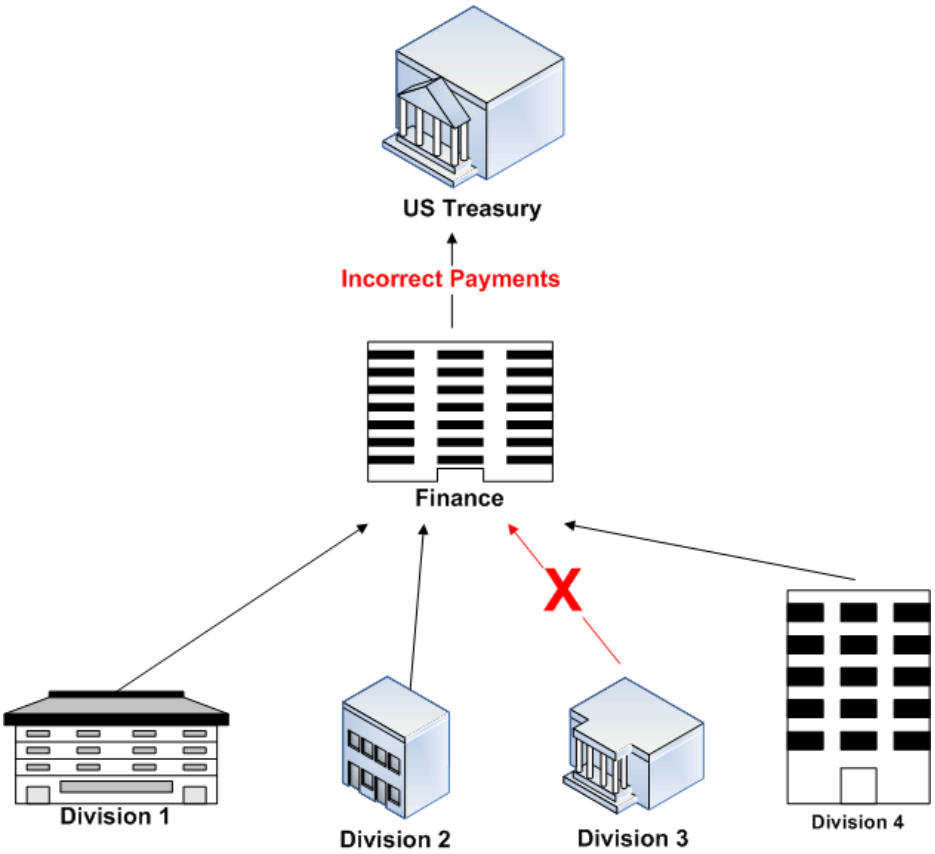

Figure 4.4: A failed file transfer can adversely affect business processes downstream of the transfer.

Frequency and Volumes of File Transfers

Another consideration when prioritizing new implementations of file transfers is the frequency with which the transfers are made. Frequent transfers are often found in core business operations, such as order fulfillment. As a general rule, if a file transfer is important enough to be done frequently, it is probably important enough to do it right all the time. This makes it a high priority to migrate to a reliable, secure enterprise solution.

The volume of data transferred is another good indicator of the relative priority of a transfer operation. Large volume transfers may be required for backup and replication processes. For example, full backups may be transferred to offsite disaster recovery centers on a weekly basis followed by smaller incremental backups performed during the week. In other cases, a large amount of data may be transferred as part of an ETL process for updating a data warehouse. Business intelligence reporting may be less critical than ensuring up-to-date disaster recovery capabilities; thus, as these examples show, volume alone is not enough to establish a priority order.

Reliability of Existing Solutions

File transfer programs that frequently fail drive up business operation costs. Failures require the time and attention of systems administrators and developers to debug and correct problems. Such failures entail opportunity costs as well: When IT professionals are chasing down problems with file transfers, they are not attending to other pressing business needs.

Reliability problems are best avoided with a combination of robust software and adequate management reporting. Dedicated file transfer solutions are more likely to be rigorously tested and provide error handling to respond to common problems, such as insufficient disk space. Reporting and alerts can be used to keep application administrators aware of changing conditions, such as low storage space or changes to access controls. This information is useful for preemptively correcting conditions before they adversely affect a file transfer operation.

Security and Compliance Requirements

Insecure homegrown solutions should be considered high-priority targets for migration to a more secure, enterprise file transfer application. Practices, such as embedding user names and passwords in clear text within a script, present security risks that should be eliminated. Quick and dirty techniques like this can compromise both internal security and that of file transfer partners.

Compliance is also a factor to consider with file transfers. Regulations governing the integrity of business data often require we demonstrate that we are employing sufficient controls to prevent tampering. That is a tough requirement to meet if we are using ftp sites to which multiple users can write. A host of compliance problems can arise with custom solutions:

- What is to prevent one user from maliciously or accidently overwriting another user's data on the same server?
- Do our homegrown solutions perform basic checks, such as checksums, to ensure files transfer correctly?
- If so, how are errors reported?
- Are logs that record details of transfers tamper-proof?

Security and compliance are critical considerations in file transfer operations that are easily overlooked with potentially costly consequences.

Prioritizing business process migration to a managed file transfer solution is just the first step to comprehensive management of file transfer operations. Many of the factors we consider when prioritizing migrations, such as volume of transfers and security considerations, also play an important role in file transfer policies and procedures.

Establishing File Transfer Policies and Procedures

One of the benefits of a consolidated file transfer solution is that the same system can be applied to a wide variety of business requirements: internal transfers, file exchange with business partners, encrypted or unencrypted transfers, and so on. Mature managed file transfer applications have many features built-in and ready for use. Our job is to use them most efficiently and effectively to do what we need within established policies and procedures.

Policies and procedures should be in place to guide the implementation of individual transfer operations as well as the management and governance of the enterprise service. In particular, we should consider:

- Standard operating procedures (SOPs) for managed file transfer operations
- Policies governing reporting requirements
- Process monitoring procedures
- Security policies
- Storage and other resource management policies

Together, these constitute the core management subject areas of enterprise managed file transfers.

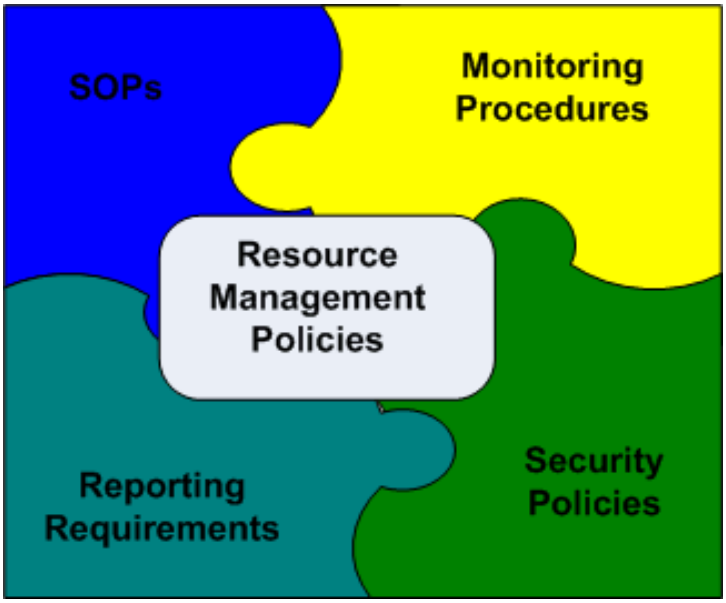

Figure 4.5: Managed file transfer requires a set of policies and procedures that defines how to operate and maintain the enterprise application.

SOPs for Managed File Transfer Operations

SOPs for managed file transfers are procedures that should be followed for all, or most, file transfer operations. The first procedure is creating a managed file transfer process. Doing so establishes jobs that run to execute the transfer. For each such process, we should collect information about:

- The business process supported by the transfer
- The business owner of the process
- The security requirements for the process, such as the need for encryption
- The frequency of the transfer
- The expected initial number and size of files transferred
- The expected growth rate of the number and size of files
- The time window in which transfers are to occur
- An indication of the criticality of the process

This type of data is important for day-to-day management as well as long-term planning. By storing this data in a centralized repository, we can have easy access to a comprehensive picture of managed file transfer processes. This, in turn, will support other management efforts, such as audits and other compliance reviews.

Reporting and Monitoring Policies

With a centralized system, we have the ability to report on the status of all file transfer operations. This is especially useful for logging details about transfer operations and notifying administrators about failures or other issues with transfers. Policies should be defined to:

- Establish rules for notifying administrators based on severity of issues
- Define types of information to routinely report on, such as number of files transferred, volume of data transferred, number of failed or interrupted transfers, and so on
- Establish response procedures based on types of errors and business importance of file transfer processes

The objective of these policies is to ensure that transfers are executing as expected and problems are detected rapidly and addressed in a standard manner. There should be no need for ad hoc responses, which are sometimes required with homegrown transfer applications.

Security Policies

It is especially important to have security policies in place to define acceptable use of file transfer services. Managed file transfer applications can automate the movement of large amounts of information within an organization and across organizational boundaries. This situation is itself a potential security risk if these services are not used according to established rules.

Security policies should address the need for:

- Patch management to ensure the managed file transfer application code is kept up to date and known code vulnerabilities are corrected.
- Vulnerability assessment because managed file transfer applications have access to significant amounts of data, making them targets for attack. Vulnerability assessments should examine configurations to detect improper settings that could be exploited by an attacker.
- Audit controls to securely log significant events, such as changes to system parameters, privilege elevation, or other events that alter the security stance of the application.
- Separation of duties with regard to supporting and maintaining file transfer operations. For example, administrators with the ability to create or delete transfer jobs should not have privilege to overwrite the event log.

Security policies such as these are designed to ensure the confidentiality, integrity, and availability of file transfer services.

Storage and Resource Management Policies

Preserving the availability of file transfer services also requires attention to limited resources, such as storage. Another set of policies should be in place to establish the use of quotas on bandwidth usage, the use of staging area storage, and rules governing the purging of data left in file transfer areas. Security policies are focused on preventing malicious disruption of services while these resource management policies are designed to avoid unintended loss of service due to resource limitations. Once policies and procedures are in place, a business can begin to effectively and efficiently use managed file transfer applications across the enterprise.

Rolling Out a Managed File Transfer Solution

Throughout, this chapter has outlined the steps required to migrate from homegrown file transfer programs to a managed file transfer solution. To deploy a managed solution, we need to:

- Assess the current use of file transfer methods
- Identify business requirements, especially those that are not addressed by existing file transfer programs
- Inventory hardware and assess how best to repurpose redundant servers
- Prioritize the order of replacement of existing solutions with an enterprise file transfer solution
- Establish policies and procedures for enterprise file transfer

As we replace existing custom solutions, we should verify several aspects of the new implementations. First, we need to able to account for all file transfer operations. They should be linked to business processes and business owners. As part of verifying the link to business processes, we need to verify that the new file transfer functions properly within larger workflows. For example, once a file transfer is complete, are follow-on events properly triggered and executed?

Second, we need to verify that the security requirements are met. This is one area where existing homegrown solutions are not necessarily good guides for judging the completeness of the new system. Authentication, authorization, and the use of encryption should all be verified.

Finally, we need to ensure that external file transfers are tested to ensure transfer partner's requirements are met. Ongoing management practices should be implemented according to the policies and procedure defined earlier.

Summary

File transfers are commonplace in today's enterprises. Data is constantly moving within and across organizational boundaries. Managed file transfer applications have taken their place alongside other enterprise applications as core business infrastructure. Moving away from homegrown, ad hoc solutions can and should be done in a methodical manner. The benefits of the approach outlined here include the ability to detect and implement business requirements, efficient use of hardware resources, a policy and governance structure for long-term efficient use of manage file transfer applications, and security practices that promote the confidentiality, integrity, and availability of file transfer services.