Realtime publishers

Tips and Tricks Guide™ To

Managed File Transfer

sponsored by

Don Jones

Copyright Statement

Volume 2

Each volume of this Tips and Tricks Guide will present a series of tips, tricks, answers, and best practices around Managed File Transfer.

Tip, Trick, Technique 8: How Does a Managed File Transfer System Integrate with My Other Business Processes?

File transfer—whether managed or not—often occurs at the beginning or end of a much more complex and involved business process. For example, if you receive a file from an external business partner, that file's receipt may kick off a business process that involved importing the file, updating databases, interacting with line of business (LOB) applications, and so forth. Companies traditionally hand off the data from something like a Managed File Transfer (MFT) server, and either use in-house applications or scripts to coordinate the remainder of the business process, or they use commercial automation tools to coordinate the process. Figure 8.1 illustrates this, with an MFT server accepting an incoming file and a separate coordination application handling the file's import, database updates, or whatever.

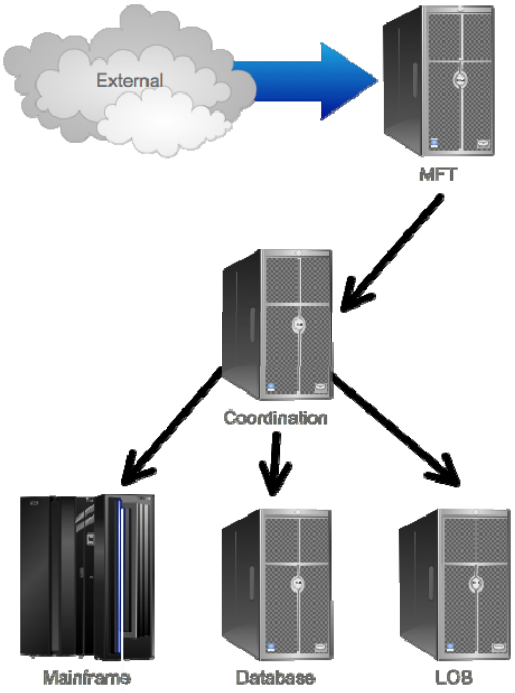

Figure 8.1: Handing off a file for coordination within a larger business process.

There's nothing specifically wrong with this technique, although it can have a few weaknesses. To begin, this approach involves moving files across several boundaries: between the MFT server and whatever is doing the coordination, between that application and whatever systems it interacts with, and so forth. All of those boundaries can be observed by whatever's doing the coordinating, so it's able to log each interaction within the data's life cycle. The coordination component misses one significant boundary, though—the one between the MFT server and the external partner. Because that happens before the coordination element is introduced to the life cycle, that interaction will be captured in the MFT server's log. The upshot here is that you're not getting a consolidated log of the data's entire life cycle.

> **Note**
>
> I'll use terms like *coordination element* or *coordination application* to generically refer to the piece of your infrastructure that coordinates the overall business process: moving data to different internal locations, updating databases, and so forth. This might be implemented as an application, a script, a server-based product, or in some other form.

Another downside is that you have to manage two sets of security for data transfer. After all, the coordination element *is* transferring data throughout your business; hopefully, it can do so in a secure fashion. The MFT element is obviously involved in a file transfer and will support security options for that transfer. But you'll be managing those two sets of security in two different places because the coordination element and the MFT server are separate components.

A significant drawback to this approach is that the coordination element almost always seems to require custom programming—either a complete coordination application written in-house or application programming interface (API)-level programming that commands a dedicated coordination system. In other words, you're going to have software maintenance to deal with. Worst off are those companies who've built their business coordination in the form of system scripts (such as Unix shell scripts, Microsoft VBScript, and so on) because those scripts are often more delicate, and harder to maintain, than a traditionally-written software application.

Finally, keep in mind that most business processes require a significant amount of data movement—something an MFT server excels at, even for internal transfers (see Tip 12 for more details). Coordination applications vary—especially if they're written internally—in their ability to offer the guaranteed delivery, secure protocols, robust logging, and other features that are common in MFT servers.

The solution, as you might expect, is to find a server application that's capable of being both a top-notch MFT server and a coordination component. Figure 8.2 illustrates this revision to our process.

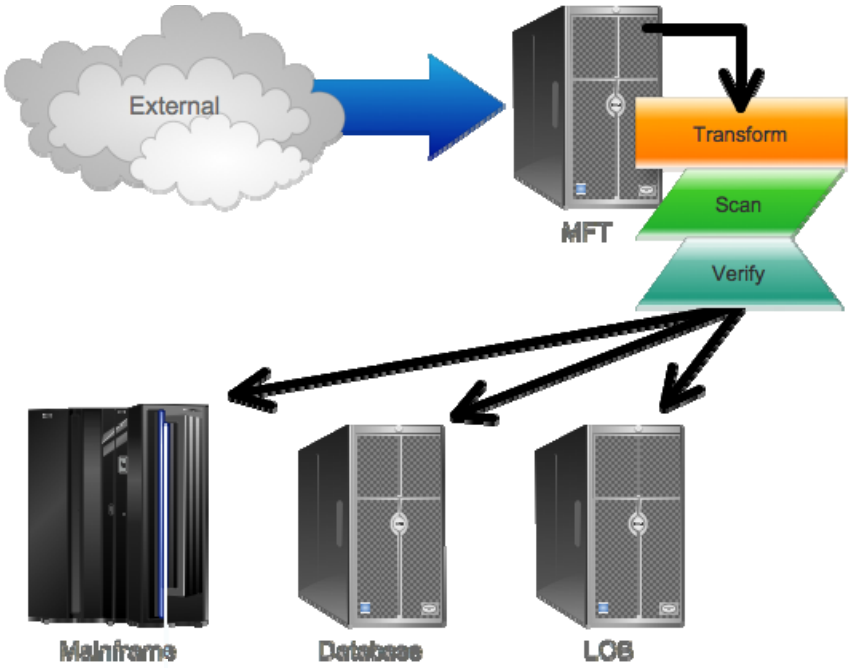

Figure 8.2: Coordinating with the MFT server.

Of course, whether you "coordinate with an MFT server" or "transfer files with a coordination server" is a matter of semantics: What you're looking for is a hybrid solution that can handle both functions. There are a few specific features and capabilities that such a solution should exhibit—some of which are alluded to in Figure 8.2. Look for:

- The ability to parse and validate incoming files. Before data starts hitting your business servers and databases, you should make sure that the data meets your formatting requirements, contains valid data, and so forth. Using data maps and lookups, the solution can help improve data integrity by kicking out unacceptable data early in the process.
- Data mapping and transformation. The incoming data might be valid, but it might not be entirely in the form that you prefer. For example, your internal systems might abbreviate US state names, while the incoming data uses full state names. Your solution should offer data transformation (also called *mapping* or *translation* by vendors) to put the incoming data into the proper form.

  Exactly *how* you define these data transformations becomes a significant point of difference between solutions. Will you have to program or write custom scripts? Can you define data mapping graphically, using a drag-and-drop interface? The easier the process is, the less time, effort, and skilled personnel you will need to implement and maintain business processes.
- Full logging. You want to log not only want what data goes where but also all internal operations—such as data translation and parsing—for maintenance, auditing, and troubleshooting purposes.

- Native code. Transforming and parsing data can be extremely resource-intensive, so focus closely on the code that a solution uses to accomplish its work. Although frameworks like Java and .NET are popular for rapid application development, bulk data processing routines typically run much faster and more efficiently when written in lower-level languages such as C or C++.
- Workflow. Businesses processes invariably involve flowchart-like, multi-step processes, and your coordination element needs to be able to handle those. You might scan, parse, and validate data first. Next you might transform elements of it. Then you might feed the data to an LOB application, and then update a database or produce a report. Several business systems might become involved in the process at various points, and their specific responses might drive branching operations within the workflow. The ability to interact with these various systems in a prescribed sequence is an important one.
- Robustness. A good integration/MFT solution should be able to (for example) checkpoint its progress and restart from that checkpoint in the even of a failure or interruption. High availability should be at least an option offered by the solution.
- Full-feature MFT. This includes all the features that you would expect from a top-notch MFT solution:
    - Secure, multi-protocol communications (FTPs, HTTPs, SSH-FTP, AS2, and other protocols)
    - Encryption, certificate, and other security-related technologies
    - Auditing, logging, and reporting

In February 2010, research firm Gartner released a note entitled "MFT Manages More Than You Think, and Governs Too." In it, Gartner suggests that MFT solutions can fit with integration middleware (what I've referred to as "coordination" elements) as a service:

> MFT is one category of integration products and services, including enterprise service buses for back-end integration, B2B gateway software and integration as a service. These integration products and services are used to securely and reliably exchange transactions, files, messages, and transactions between application systems, external business partners, and cloud services with the same level of governance and compliance as MFT. What differentiates MFT from other forms of IaaS is (1) its unique focus on particularly large files, and the scheduling and management of moving of very large numbers of files and bulk data between applications and businesses, and (2) the movement of files and data in usage scenarios not typically addressed by many integration solutions, such as enhancing the performance of file attachments in e-mail. Note, however, that all integration solutions are rapidly converging such that, for example, MFT solutions continue to incorporate more-general-purpose integration capabilities, such as enterprise service buses or IaaS, and vice versa.

That last sentence is the key: MFT solutions are increasingly taking on the middleware "coordination" role and integrating features that were traditionally performed by separate applications. It makes perfect sense because so much of that middleware's role involves file transfer and data parsing or transformation. In fact, even the terms "file" and "data" (as in, the things you transfer from place to place, and which become involved in your business processes) are being replaced by the term *message*, a broader word that encompasses all kinds of data moving around and participating in business processes.

Bolt-On vs. Suite

Be careful to distinguish between solutions that offer integrated MFT and middleware capabilities and those offering an MFT bolt-on to an existing middleware application. In a fully-integrated application, you might still purchase specific capabilities a la carte, but the connectivity between the components tends to be more seamless. You also tend to get more robust integration of solution-wide features such as monitoring and logging.

Bolt-ons typically involve an existing middleware solution having some MFT capabilities added through an add-on or plug-in. The depth of the bolt-on integration varies from solution to solution, so investigate closely to ensure you're getting truly-integrated feature sets. You don't want to be in the position of buying what is essentially a standalone MFT solution with a few specific integration points, as you'll find yourself in substantially the same scenario as depicted in Figure 8.1, with many or all of that scenario's disadvantages.

Also keep in mind that this kind of transfer/integration process will often involve data *leaving* your organization—the opposite of what Figures 8.1 and 8.2 picture. In those cases, you may want additional capabilities—such as data filtering, to remove or redact sensitive information or at least report or log the transmission of sensitive information. Figure 8.3 illustrates this variation to our scenario, and you might imagine a global filter that redacts any information meeting the pattern for a US Social Security Number. Using this kind of global filtering, you can be better assured that any outgoing data will be scrubbed of sensitive information that appears to meet this pattern. Of course, you could disable that filter for specific processes that were permitted to send that sensitive information, after ensuring that they were doing so using the appropriate measures for security and privacy.

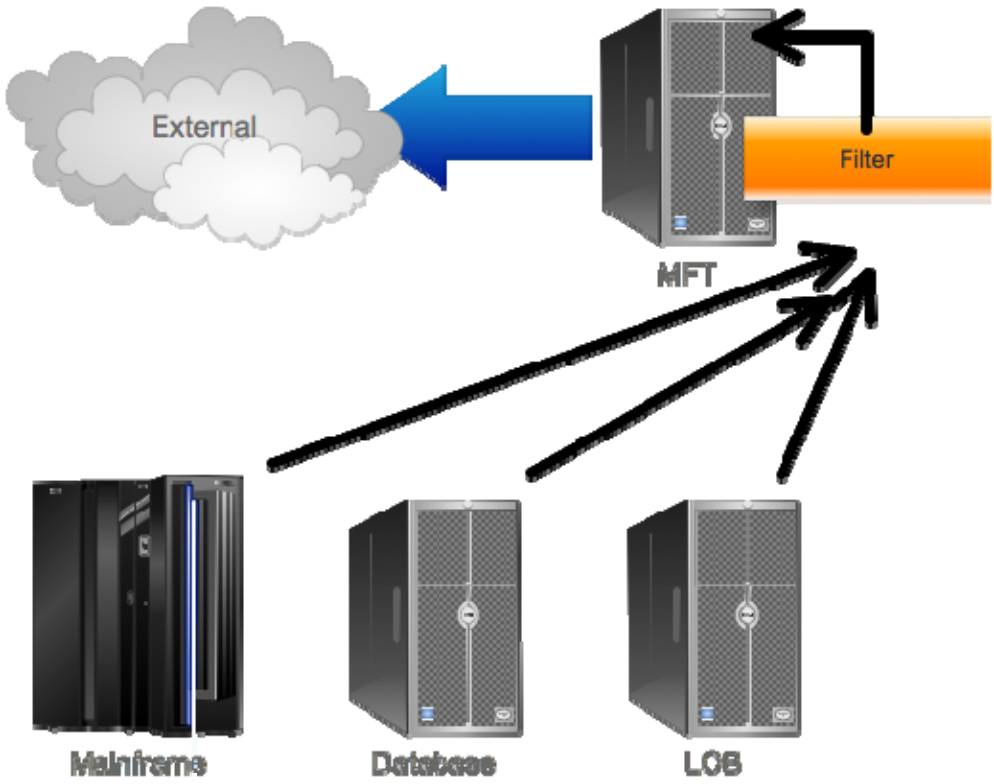

Figure 8.3: Adding filtering to outbound data as part of an overall business process and data transfer.

The moral here is that the more you can have these "coordination" middleware capabilities integrated as part of an MFT solution, the more manageable, securable, and efficient your business processes will become.

Tip, Trick, Technique 9: Can a Managed File Transfer System Enable Central Management and Control?

It should. Let's define what "central management and control" means:

> The ability to, from a single location, define configuration settings and rules that govern multiple discrete functional elements.

You might also see this referred to as *policy-based management,* the idea being that you configure a central policy that describes your desired configuration, and the individual elements of your infrastructure configure themselves to meet that policy. Change the policy, and their configuration changes. Change an individual element, and it ideally reconfigures itself to remain compliant with the central policy.

Your company's personnel management practices are probably a good example of central management and control. Individual managers don't set vacation policies, determine employee benefits, or establish rules for things like dress code. Typically, those things are all defined centrally, as part of a companywide series of policies that all employees are expected to follow. The concept, as applied to human beings, is age-old. It's odd that we've taken so long to apply the concept to computers.

Managed File Transfer (MFT) is a relative newcomer to the IT world, so let's focus on something a bit older as a better example: the Windows operating system (OS). When Windows first started making its way into businesses, networks were a rarity for most businesses, and computers operated entirely on their own. Without any central communications network, there was no reason to expect computers to be centrally controllable. Yet even though networks were common by the mid-1990s, we were still manually configuring every client computer and server with the appropriate settings. Maintaining and enforcing those settings was typically impractical. It wasn't until Microsoft introduced Group Policy in 2000 that we started gaining the ability to establish central configuration policies, and to have those policies automatically apply to our computers. Today, that technology continues to grow more robust, including additional elements that can be controlled by central policy.

That's the exact pattern that most IT efforts seem to follow: We start out with one of something, so there seems to be little need to control it centrally. As Figure 9.1 shows, when you only have one of something, it *is* the central point of control. MFT began in much the same way: Companies tended to have one MFT server, so controlling it centrally was the same as controlling it directly.

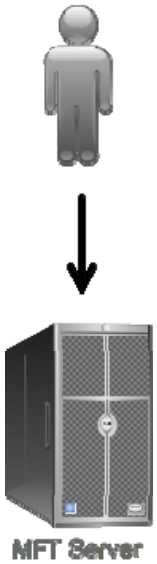

Figure 9.1: Control is easy with only a single element to manage.

But, like all things in IT, growth tends to occur. You add a second MFT server to provide high availability. A third and fourth are added to support a new business effort. A fifth, sixth, and seventh server are added to support higher file transfer data volumes and to act as business process middleware. Before long, you're not managing a single point any more—you're managing a bunch. As Figure 9.2 shows, maintaining consistent settings between these servers becomes more labor-intensive, more error-prone, and less reliable. It's as if every boss is making up his own vacation policies, causing chaos and strife in the ranks.

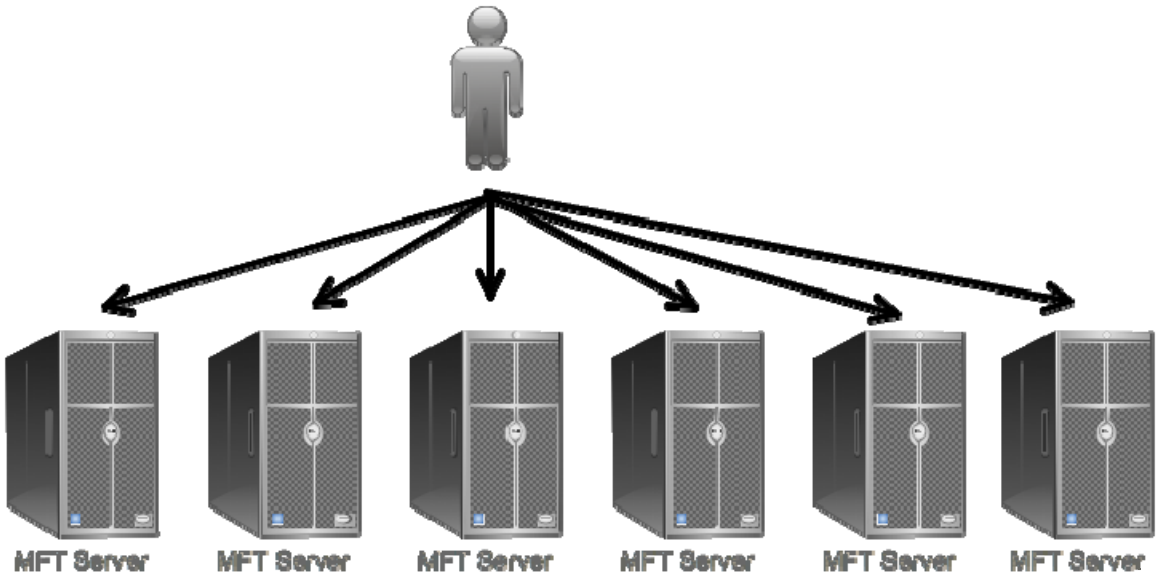

Figure 9.2: Managing multiple elements becomes more labor-intensive and error-prone.

The solution—and something a good MFT solution will support—is to move to policy-based management, just as we always eventually do with IT infrastructure. With policy-based management, a single policy administrator can centrally control policy for the entire infrastructure, from a single place—not unlike Microsoft's model for Group Policy, or for that matter any model for central, *policy-based governance*.

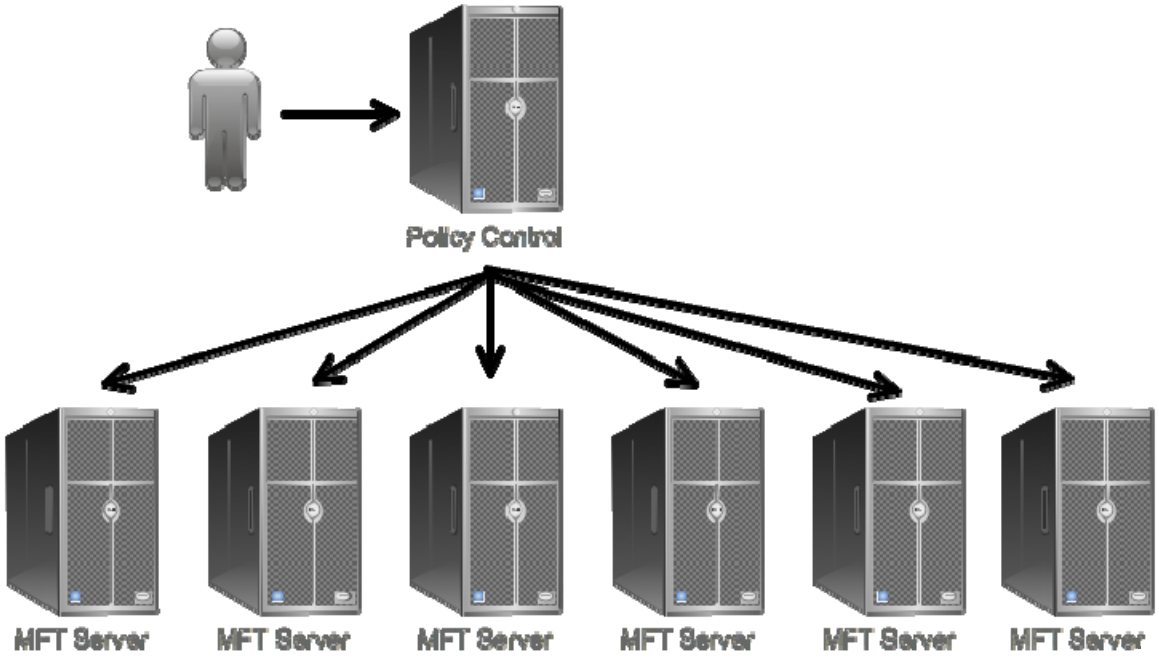

Figure 9.3: Policy-based control is centralized, easier, and less prone to error.

There are two other significant benefits with this approach:

- It's no longer as critical that individual servers' configurations be manually audited to ensure compliance with corporate configuration policies. Instead, you simply audit the central policy, knowing that all functional elements are being governed by that policy. Audits also become centralized, more efficient, and even more broadly-implemented (you're no longer, for example, doing spot-checks on individual servers—you're effectively checking them all at once).
- You gain better separation of duties. As Figure 9.4 shows, front-line administrators (shown in blue) can continue to manage the servers they're responsible for, but policy managers (in grey) have overriding control on configuration policies. Administrators lose the ability to override policy, ensuring a compliant environment at all times.

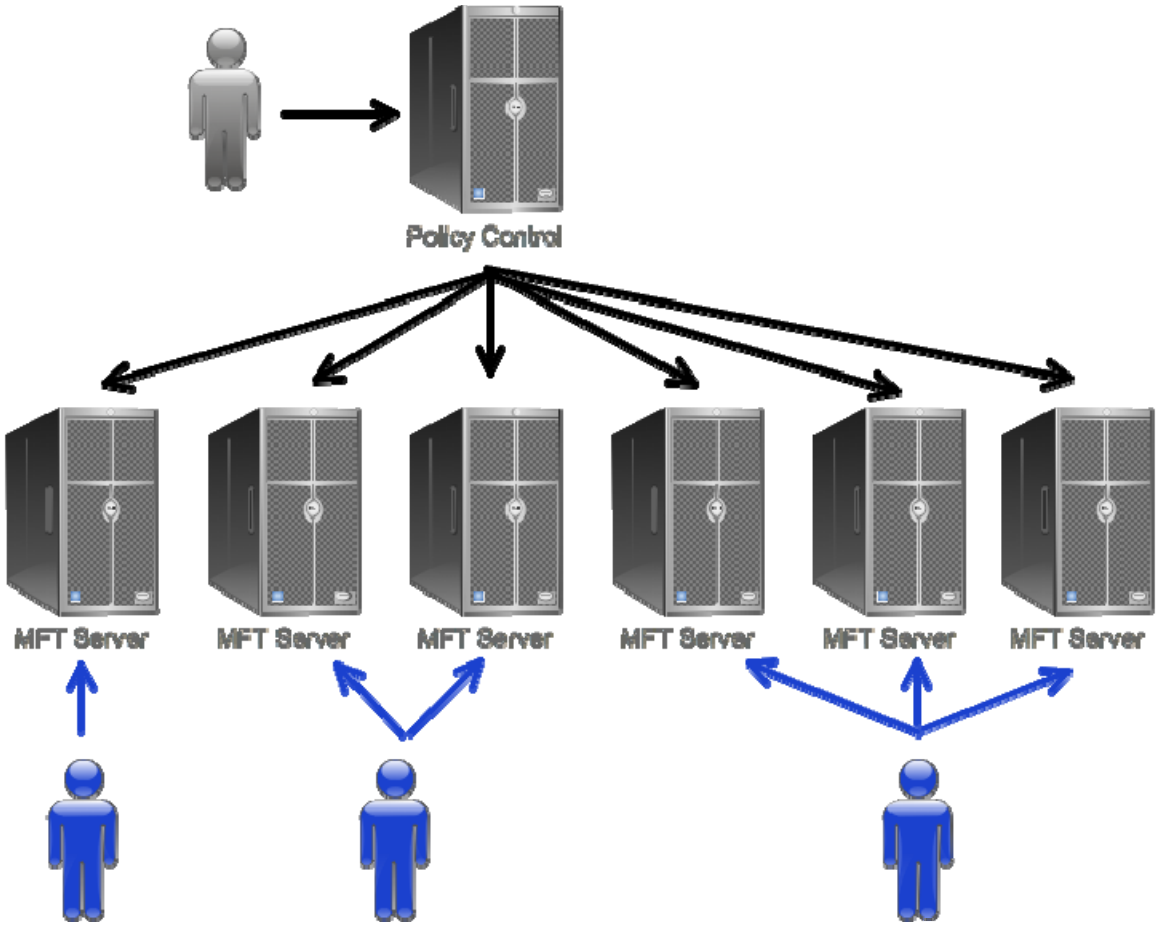

Figure 9.4: Policy-based control offers separation of duties between administrators and policymakers.

The ability to offer this kind of policy-based control is a major argument for moving to a single high-end MFT solution for your file transfer and data movement. Obviously, each vendor in the market offers their own proprietary management scheme, so they can't manage each others' solutions. By moving to a single platform or brand, you gain central control over all your MFT resources—which means you *gain central control over all file transfers.* That latter part is the important one because it enables you to (for example) ensure that all file transfers are properly secured. You can ensure that only authorized individuals can transfer files, even through ad-hoc mechanisms (which a good MFT server will provide). So you're not just gaining central control over a bunch of servers, you're gaining central control over critical business processes and activities.

Tip, Trick, Technique 10: Do I Need File Transfer Protocols Other Than FTP?

The short answer is "Yes. Absolutely."

The longer answer, which I'll obviously give you, entails the "why" as well as which other protocols you think you might want in a Managed File Transfer (MFT) system. Let's start with the protocol list first. Here are the minimum protocols I think you should look for:

- AS1, AS2, and AS3. These "applicability statement" specifications describe how to reliably and securely transport data over public networks, such as the Internet. They specify the use of digital certificates and encryption for security. AS1 is based on SMTP and S/MIME, essentially making it a kind of "secure email" method; AS2 is based on HTTP and S/MIME, and encodes files as "attachments;" AS3 is currently a draft specification and utilizes FTP. Because these specifications are not successors to each other, but are rather unique specifications in their own right, having support for all three offers maximum flexibility.
- Local file system copy as well as support for Server Message Block (SMB) network file transfer. You may also want support for additional network file transfers, such as NFS.
- FTP. The good old file transfer protocol still isn't dead, although it doesn't offer anything in the way of security by itself. For that reason, you'll also want the following additional protocols.
- FTPS (FTP over an SSL) connection, SFTP (FTP over SSH), and SCP2 (Secure Copy Protocol 2), all of which are alternatives to FTP and in most cases use a standard FTP connection that is secured by means of a wrapper connection—either SSL or SSH.
- HTTP and HTTPS (HTTP over SSL), which are often used to exchange data with things like Web services or to simply transfer data.
- SMTP, POP3, and IMAP4, all of which are email protocols. You'll often want the additional ability to automatically sign and encrypt emails, deal with file attachments, and so on. Encryption can be provided by standard digital certificates, or you may opt for industry-standard encryption modules like OpenPGP.

Of these, the AS*x* specifications are often the hardest to come across, although they've become standards for the exchange of structured data in Electronic Data Interchange (EDI) environments. Some vendors may only offer the AS*x* protocols as an option. Obviously, if you have a specific need for them, you buy the option; make sure you buy a solution that *has* the option.

Why do you need *all* of these, or at least the option to add or enable them to your MFT system? There are several reasons, the first of which is *flexibility.* Here's the thing: Most companies seem to acquire an MFT solution in response to a specific project. You have an external business partner, you need to securely and reliably exchange data with them, and an MFT solution is the quickest way to get up and running. What you *don't* want to do is get an MFT solution that *only* meets that specific project's requirements. I've never seen a company that didn't eventually start using their MFT system for other projects, so getting a system with maximum flexibility will help ensure that the system can grow to handle whatever other needs crop up in the future. The alternative—which, sadly, I've seen a lot of companies do—is to buy a different MFT system for each project. Companies who do that inevitably feel the pain of configuring, monitoring, managing, patching, and controlling those disparate systems—and most of them eventually start looking for a replacement solution that can consolidate all of their operations into a single platform. My advice: Go flexible, and buy that single platform at the outset, even if it's a bit more feature-rich that you specifically need at the moment.

The second reason is security. Although you might start out with simple needs that are met by basic FTP, odds are those needs will change. Security is becoming a mandate for more and more companies, with legislation and industry rules increasingly requiring encryption and other security measures. Even if you only need "plain" FTP today, odds are that you'll need a more secure protocol in the future—so why not get an MFT solution that supports it out of the box?

There are a few "second-tier" protocols that I'll mention. These are ones that you may already be using, or may already be familiar with, that I don't expect you'll find in most MFT solutions. Or, if they are present, they don't support the MFT solution's full range of features. I'll explain why with each:

- 9P. I mention this one only because I once had three different consulting clients specify it within the span of 2 weeks, and I actually had to look it up. Originally developed for the Plan 9 OS from Bell Labs, versions are available for both Unix and Linux. It's unusual to find MFT solutions that support it because it doesn't do much that simple FTP doesn't offer, and it isn't inherently secured.
- HFTP. This is a protocol used to pass FTP through an HTTP proxy. MFT solutions are more likely to support FTP over SSL, and most MFT solutions are directly compatible with proxy servers for most protocols. For that matter, it's more common to put a specialized MFT server in your network's DMZ, rather than passing traffic through a proxy, so that the MFT solution can maintain full track and control of your data through every connection.
- WebDAV. Although this HTTP-based protocol can certainly transfer files, it isn't as efficient as the protocols more commonly supported by an MFT solution. WebDAV was designed more for individual file "put" and "get" operations, and less for bulk transfers.

- RCP. This "remote copy" protocol is usually provided with Unix distributions. It does not inherently provide security, and it is typically not used over public networks like the Internet—so you won't often find support within MFT solutions. That said, SCP, or *secure* copy, uses a secured variation of RCP and you'll often find SCP or SCP2 support in an MFT solution.
- SFTP. This alternative use of the "SFTP" acronym refers not to *secure* FTP (FTP over SSH) but to *simple* FTP. This is an unsecured protocol, and was never widely accepted; the Internet Engineering Task Force (IETF) officially ranks this as a "historic" protocol. For those reasons, you won't see it in an MFT solution.
- TFTP. *Trivial* FTP is useful for quick file transfers that don't require security, do not need guaranteed delivery, and often don't need authentication. In other words, TFTP is pretty much the opposite of MFT, so don't look for it in most MFT solutions.
- UFTP. This is a UDP-based variant of FTP, meaning it uses the User Datagram Protocol (UDP) rather than the Transmission Control Protocol (TCP). As a result, UFTP doesn't guarantee delivery and it doesn't natively provide any security, so it's rare to see it in an MFT solution.

So most of these aren't in MFT solutions because they lack the security, the reliability, or both, that an MFT solution requires to operate. Almost all of them have more commonly-used alternatives, which I've outlined, that you *will* find in a good MFT system. Another takeaway is that if you have any internal processes using any of these less-suitable protocols, then moving to an MFT solution will also mean migrating to more secure, more manageable, and more reliable protocols.

Tip, Trick, Technique 11: How Can a Managed File Transfer System Reduce Overhead on My IT Staff?

Through three features common to most Managed File Transfer (MFT) solutions: guaranteed delivery, non-repudiation, and automation. These not only save time and overhead for your IT staff but the rest of your personnel as well.

Guaranteed Delivery and Non-Repudiation

Guaranteed delivery typically involves the ability to detect a file transfer that has become interrupted, or has failed to start, and to automatically retry or resume such transfers. If the other end of the transfer isn't another MFT solution, the other end must at least support transfer-resumption, which most FTP and Web servers do support these days. Guaranteed delivery is also closely tied to the concept of *non-repudiation.*

Non-repudiation is the ability for your MFT system to authoritatively determine when a file was successfully received, in its original condition, by the recipient. Typically, this information is saved into a log and can often be viewed either in a log viewer or as a status of whatever transfer job you're interested in.

Not all low-level file transfer protocols provide native non-repudiation features. Instead, MFT solutions often extend these protocols to include post-transmission integrity checks, whereby the receiving system calculates a checksum, hash, or some other verification, and transmits it back to the sending server. That means that, in some cases, non-repudiation is not available for all protocols and all transfers. A transfer from an MFT system to a relatively low-end FTP server, for example, might not support non-repudiation.

Non-repudiation can also involve an authentication element, whereby the MFT system keeps track of who sent the file, when they sent it, and the fact that the file was not altered in transit. This additional capability often requires a compatible MFT server or client on the other end of the transfer. For example, one place where you'll see this in action is when a user creates an ad-hoc file transfer using a file transfer client to a compatible MFT server. The client is able to send the user's identity, along with a cryptographic hash of the file being sent; the server can receive that information, verify that the hash matches the file that was received, and store that collected non-repudiation information as part of the transfer's log.

> **Note**
>
> Non-repudiation, transfer retry, and transfer resume are three features often lumped together under the category *guaranteed delivery;* some vendors break non-repudiation out into a separate feature.

To quickly summarize the MFT features we're discussing:

- Transfer retry: The ability to automatically restart a transfer that has failed to start.
- Transfer resume: The ability to resume sending a file whose transfer was interrupted, starting from the point of interruption rather than starting from the beginning.
- Authentication: The ability to know who sent the file, or who is receiving the file.
- Integrity checking: The ability to determine whether the file was altered in transit.
- Audit trail: The ability to preserve authentication and integrity checking results.

Collectively, these guaranteed delivery features can save your staff a lot of time and effort:

- You'll never have to have the IT team spend time tracking down a file that seems to have gone missing in transit; you'll have a clear audit log that tells you whether the file was received on the other end.
- You'll never have to re-send a corrupted file because the MFT solution can detect that corruption and automatically retry the transfer.
- You'll never have to hear, "oh, but we didn't get that file" again, because the MFT solution can confirm, upon completion of the transfer, who sent (or received) the file, and authoritatively determine that the file was, in fact, received.

You can kind of think of this as similar to sending a message through the postal service or sending it via an express courier. With the postal service, you drop the message into a mailbox and wait a few days for it to arrive. You have no confirmation that it did so, and you have few means of determining whether it arrived intact. With an express courier, in contrast, you have the signature of the recipient, a sealed box (ideally) that tells you the package made it intact, and the identity of the sender printed right on the box. How often would your company send critical business information via a postal service rather than an express courier? Probably pretty rarely—because you end up wasting time re-sending things, tracking down results, hoping nothing was altered, and so forth. An MFT system is the express courier of the file transfer world, and it can save you time and frustration in a very similar way.

Automation

There's no question that automation can save time—but aren't your administrators handling automation on their own, through scripts? Do you *need* an MFT solution?

Well, yes. I believe—and most industry analysts back me up on this—that scripting can be a wonderful way for administrators to automate something that they would normally do manually. But administrators wouldn't normally complete your company's file transfer needs manually, in part because they can't provide guaranteed delivery, guaranteed security, extensive logging, and all the other features that you need to accompany file transfers. If an administrator can't do it manually, you shouldn't attempt to do it with a script.

Most companies start looking at an MFT solution for one, or both, of two reasons: better security and/or more reliable and manageable automation. That automation often involves workflow that goes beyond the basic file transfer because companies need that file transfer to kick off (or be kicked off by) other business processes that deal with (or generate) the transferred data.

In fact, *workflow* is probably a better way to describe this MFT time-saver than *automation.* Organizations' systems and processes tend to grow somewhat organically, and administrator-created scripts and other "hacks" were the primary means of automation simply because they could be created and maintained dynamically. Eventually, however, organizations want to settle on more formal, thought-out processes, and they want to move away from workarounds like scripting. Businesses don't *want* their processes to be continually in flux, but they *do* want them to be *reliably executed.* Businesses don't want to have to re-script everything just because a new file transfer protocol comes into play or because a skilled administrator leaves the team and nobody can understand the scripts he or she wrote.

Let me draw an example (excerpted and modified from http://eziniearticles.com/?Workflow-Automation---Replacing-Scripting-Or-Code&id=4679582). Here's a shell script that automates a transfer between a host and a client, and back again, with a log entry:

```
#!/bin/bash

DATE=`date +%d.%m.%Y-%H.%M`
SRV=sftexa

#scpg3 put
echo "/opt/xxxxx/bin/scpg3 -B -q testfile $SRV:test" >> scpg3_put_$DATE
/opt/xxxxx/bin/scpg3 -B -q testfile.dat $SRV:test
echo $? >> scpg3_put_$DATE

#scpg3 get
echo "/opt/xxxxx/bin/scpg3 -B -q $SRV:test test" >> scpg3_get_$DATE
/opt/xxxxx/bin/scpg3 -B -q $SRV:test test
echo $? 0>> scpg3_get_$DATE
```

That's a wonderful, concise piece of work, but did it save time? Yes, it will accomplish the automation, but an administrator had to *write* it, and an administrator will have to *maintain* it whenever the operating conditions change. An MFT solution would normally eliminate this scripting step, and instead use a workflow engine to describe the desired process. This automation could then potentially be created by a skilled knowledge worker—taking the IT staff out of the loop completely. That knowledge worker might simply have to designate the source folder, the destination folder, the file(s) to be transferred, and the schedule on which to complete the transfer. Problem solved—with no administrator time. Start adding options like guaranteed delivery, encryption, security, and so forth, and the script starts becoming even more complex, but the MFT solution accomplishes it with a few extra button clicks from the knowledge worker.

Tip, Trick, Technique 12: Do I Need a Managed File Transfer System to Move Files Around Internally?

Need? Perhaps not. *Want?* Almost definitely, and for two main reasons: security and business integration.

More and more companies are concerned about information security, whether due to external mandates (like legislation or industry group rules) or internal policies. Yet these companies too often focus on their network firewall as the only security boundary. Take a look at Figure 12.1, which outlines a business process that involves moving data from an external business partner into several internal business systems.

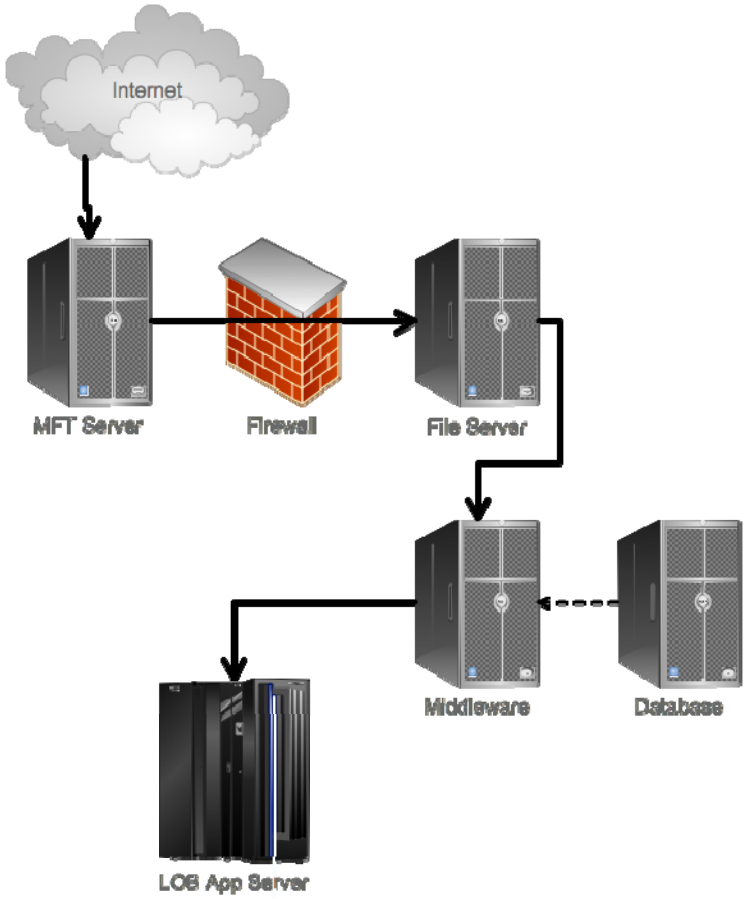

Figure 12.1: Example business process involving data movement.

Here, a Managed File Transfer (MFT) solution in the company's DMZ receives data from an external business partner. It transfers that data to an internal file server. A middleware application picks up the deposited file and processes it, perhaps using a companion database for data mapping and data transformation. The middleware server then feeds the data to a line of business (LOB) application.

The problem with this approach is that, in most cases, only the initial transfer to the MFT server from the business partner, and possibly the transfer of the file to the file server, is secured, authenticated, guaranteed, and logged. As Figure 12.2 shows, the other steps—which are outside the purview of the MFT solution—use unsecured, unauthenticated, unlogged transfers (shown in red) handled by the middleware. In some cases, such as on the file server, the data may well sit unencrypted, further exposing it to potential compromise.

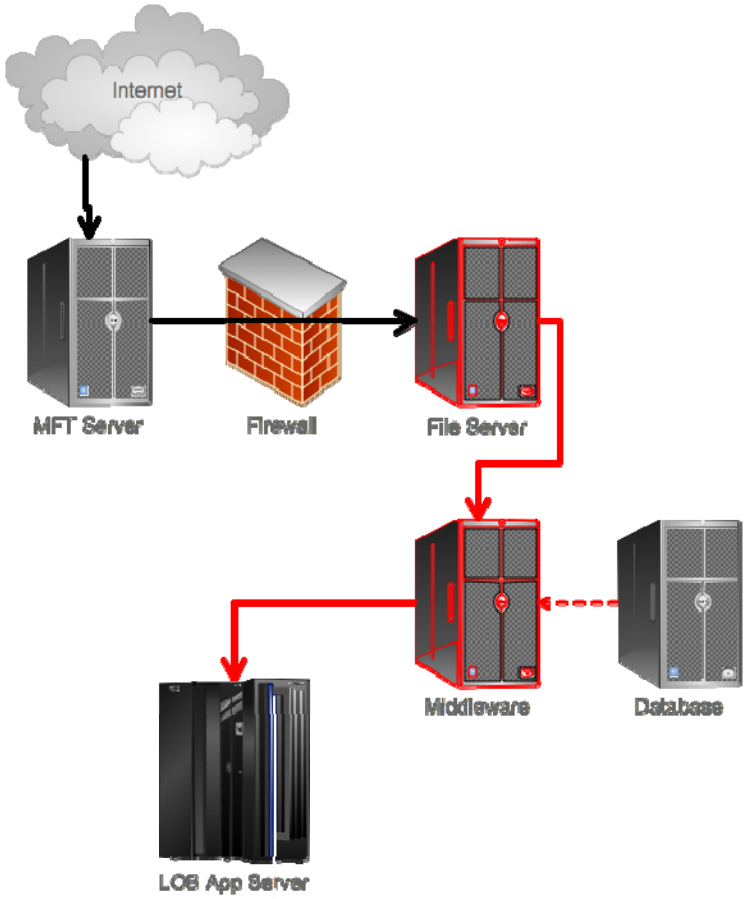

Figure 12.2: Highlighting the weak points in the scenario.

This is why so many MFT solutions (as I discussed in Tip 8) are adding process-integration capabilities and replacing middleware. Consider Figure 12.3:

1. Here, a DMZ-based MFT solution receives the file. It's secure, logged, and authenticated.
2. The file is securely transferred to a second MFT server inside the firewall, which will coordinate the remainder of the business process.
3. Perhaps some middleware is still used for data transformation. No problem. The internal MFT server transfers the file to the middleware, keeping the MFT solution in control. The internal file copy can be secured, logged, and authenticated.
4. The results are then transferred to the LOB system *by the internal MFT solution,* again ensuring that the file is secured and that the transfer is logged. In addition, the entire data life cycle is under the control of a single system.

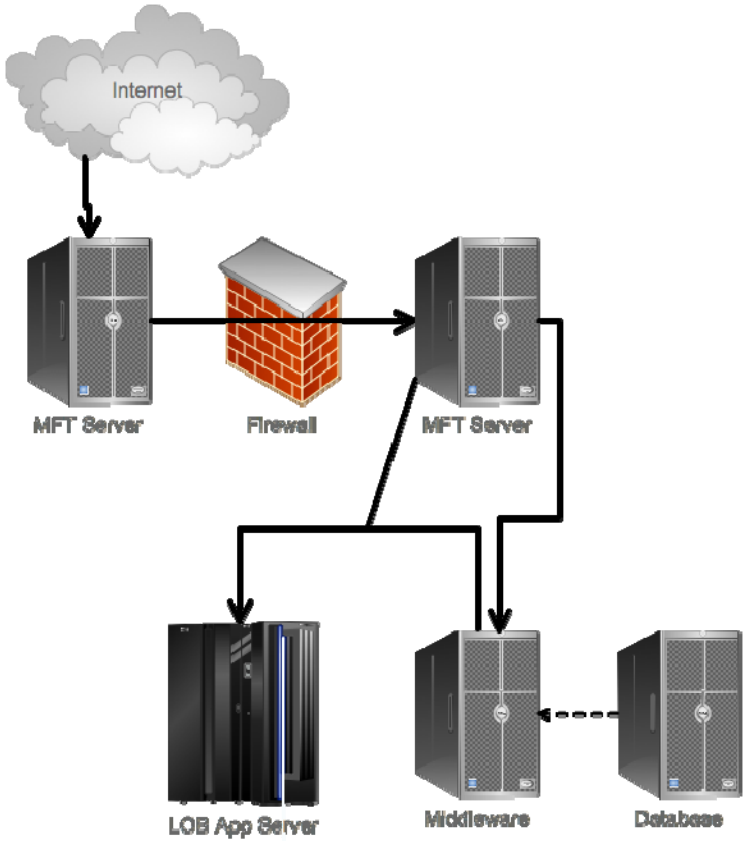

Figure 12.3: Using MFT to secure the entire life cycle and process.

MFT can even be useful for internal file transfers between users in an ad-hoc fashion. Rather than using email file attachments—which, frankly, do nothing but bog down the mail system and present opportunities for data to be exchanged with less security—users utilize a client or Web site to transfer files to each other using an MFT system. It's much the same model that they might use to transfer files to an external user, having the same security, authentication, logging, and policy-based control that MFT offers.

Download Additional eBooks from Realtime Nexus!

Realtime Nexus—The Digital Library provides world-class expert resources that IT professionals depend on to learn about the newest technologies. If you found this eBook to be informative, we encourage you to download more of our industry-leading technology eBooks and video guides at Realtime Nexus. Please visit http://nexus.realtimepublishers.com.