Realtime publishers

Tips and Tricks Guide™ To Managed File Transfer

sponsored by

IPSWITCH FILE TRANSFER

Don Jones

Copyright Statement

© 2011 Realtime Publishers. All rights reserved. This site contains materials that have been created, developed, or commissioned by, and published with the permission of, Realtime Publishers (the "Materials") and this site and any such Materials are protected by international copyright and trademark laws.

THE MATERIALS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. The Materials are subject to change without notice and do not represent a commitment on the part of Realtime Publishers or its web site sponsors. In no event shall Realtime Publishers or its web site sponsors be held liable for technical or editorial errors or omissions contained in the Materials, including without limitation, for any direct, indirect, incidental, special, exemplary or consequential damages whatsoever resulting from the use of any information contained in the Materials.

The Materials (including but not limited to the text, images, audio, and/or video) may not be copied, reproduced, republished, uploaded, posted, transmitted, or distributed in any way, in whole or in part, except that one copy may be downloaded for your personal, non-commercial use on a single computer. In connection with such use, you may not modify or obscure any copyright or other proprietary notice.

The Materials may contain trademarks, services marks and logos that are the property of third parties. You are not permitted to use these trademarks, services marks or logos without prior written consent of such third parties.

Realtime Publishers and the Realtime Publishers logo are registered in the US Patent & Trademark Office. All other product or service names are the property of their respective owners.

If you have any questions about these terms, or if you would like information about licensing materials from Realtime Publishers, please contact us via e-mail at info@realtimepublishers.com.

[**Editor's Note:** This book was downloaded from Realtime Nexus—The Digital Library for IT Professionals. All leading technology books from Realtime Publishers can be found at http://nexus.realtimepublishers.com.]

Volume 4

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 19: How Can I Identify Areas Where Managed File Transfer Can Help My Company?

As I've written in prior tips for this book, most companies adopt a Managed File Transfer (MFT) solution in response to a specific project or immediate business need. I've seen a lot of companies pick a solution based only on immediate needs, only to pick *another* system when some new need comes along, then *another* system to support the next project, and so on—until they have a whole array of different, competing solutions that all require their own maintenance, have their own overhead, and so on. In the end, the companies spend more money acquiring and supporting point solutions than they would have if they'd just bought one really flexible system in the first place. So with that lesson in mind, it pays to think about the different places an MFT solution might fit in eventually, if not today. That way, you can select a solution that can grow as your needs evolve.

Why MFT Is Traditionally Deployed

As Figure 19.1 shows, the traditional first use of MFT within an organization is often to transfer files between the organization and an external system, such as a business partner.

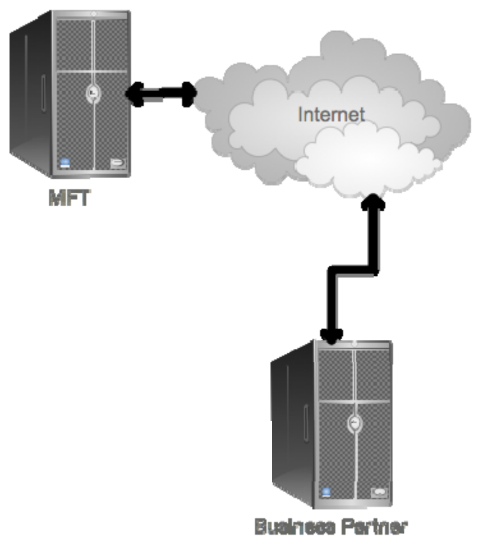


Figure 19.1: Transferring data between business systems.

Many companies implement an MFT system for this purpose after experimenting—and usually becoming frustrated with—homegrown solutions, such as scheduled FTP scripts and the like. As I've repeatedly emphasized throughout this book, MFT systems provide features above and beyond simple file transfers: They offer guaranteed delivery, delivery tracking and auditing, non-repudiation, and other features that are often desirable when moving data between organizations. These transfers are typically referred to as *server-to-server* or *system-to-system* transfers because, after full implementation, there's often very little human interaction on either end of the exchange.

Securing User-to-User Transfers

At some point, the organization tends to realize that just as much data is being exchanged directly between users as is between systems—and those *ad-hoc,* or *person-to-person* transfers are typically conducted through much less-secure means, such as email. That's when everyone starts looking to the MFT system to see whether it can manage those ad-hoc transfers as well, offering the same guaranteed delivery, auditing, security, and other features it does for server-to-server transfers. Figure 19.2 illustrates this new kind of need.

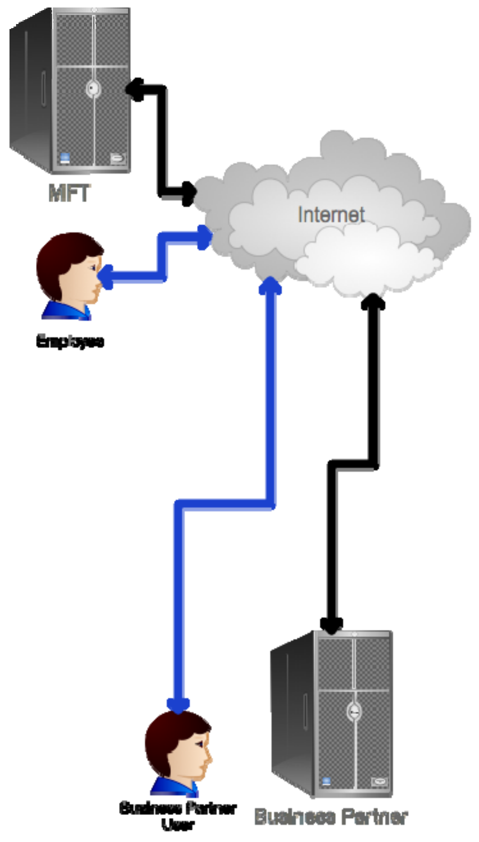


Figure 19.2: Adding user-to-user transfers.

Here, the entire file transfer is still completely managed, but it's initiated on an as-needed basis by individual users rather than being initiated on a schedule as part of an ongoing business process.

Organizations often realize soon thereafter that their own employees are moving a lot of data back and forth. Although things like file servers often facilitate that kind of transfer, for highly-sensitive or mission-critical data, a flexible MFT system can replace the file attachments usually sent via email. As Figure 19.3 shows, using the MFT system for intra-organizational ad-hoc transfers can be beneficial (and in a well-designed MFT system, the process of doing so isn't that different from traditional email file attachments).

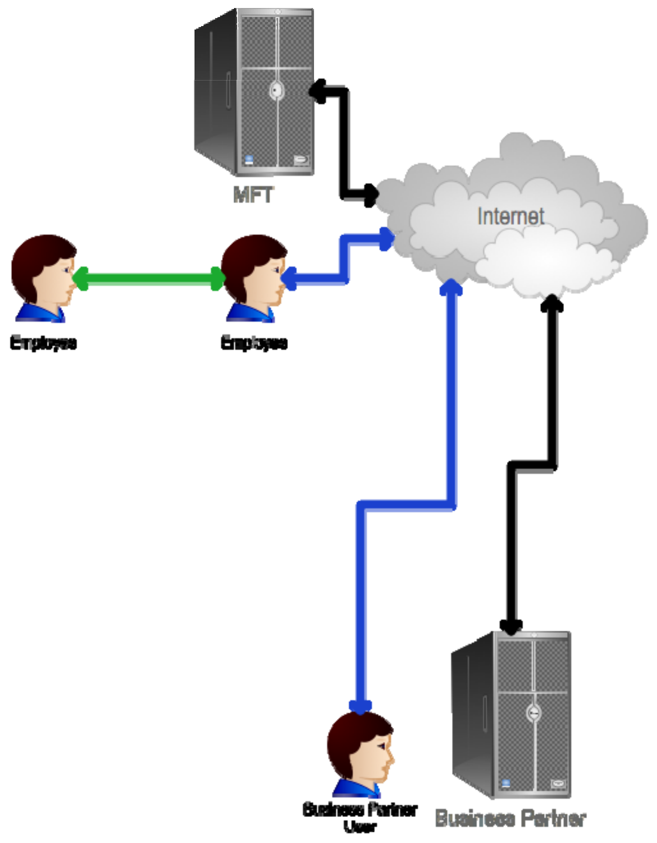


Figure 19.3: Intra-organizational, ad-hoc file transfer.

Intra-Application Transfers

Having dispensed with homegrown scripts for external file transfers, and having reduced the use of email file attachments as a means of moving critical data between users both internal and external, the organization now looks to other places where data is being moved around by homegrown means—and they start to wonder whether the MFT system couldn't handle those transfers better as well. After all, as Figure 19.4 shows, this movement isn't that different from transferring data to and from external partners—you're just moving it between *internal* systems and applications.

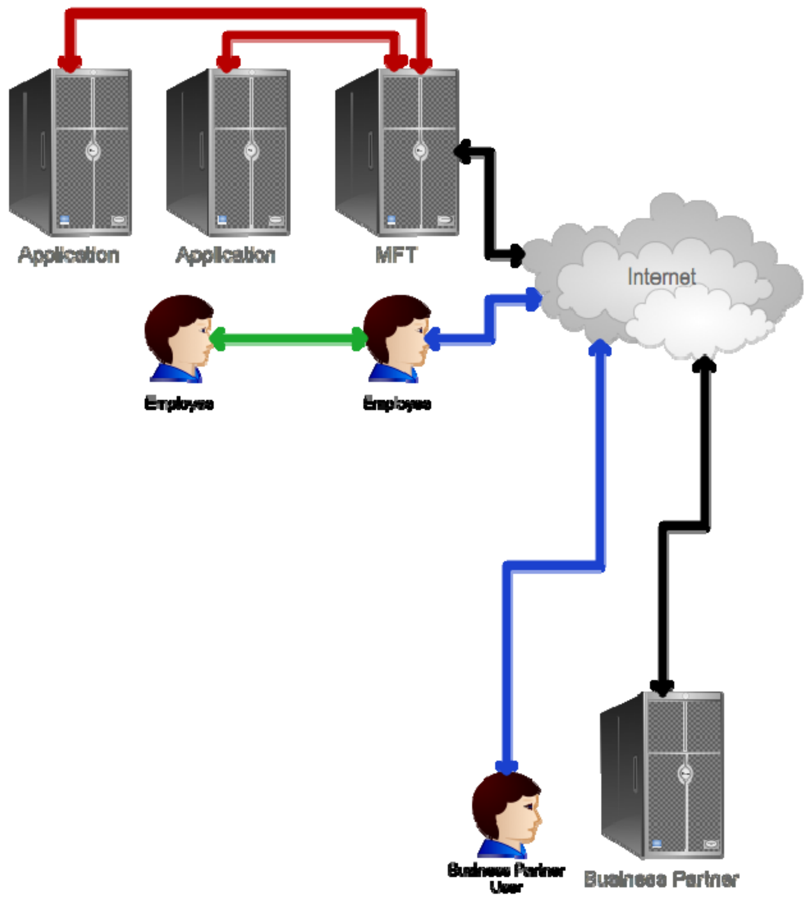


Figure 19.4: Transferring data between internal systems.

MFT systems can offer an advantage here in that they can not only move data but also transform and translate it between systems. MFT sort of moves beyond mere data movement and into a complete system for orchestrating the integration of data between systems internal and external.

Look Carefully to See Where MFT Can Be of Use

The moral of the story is to think about *all* the ways in which your company currently moves data around, both internally and externally, and to consider the potential advantages of using an MFT system to facilitate those transfers. Even if you decide not to *use* the MFT system for all of those tasks right away, you'll want to select a system that *can* be used for those tasks so that you're not shopping for yet another MFT system a few months down the line.

Tip, Trick, Technique 20: Isn't Email a More Secure Way of Sending File Attachments?

Email is "more secure" than posting the data in clear-text on a Web site perhaps—but that's about it. Yes, email messages can be encrypted—although users seldom do so—but the messages are simply passed through too many hands to be considered "secure." Consider Figure 20.1, which shows the path of a typical email.

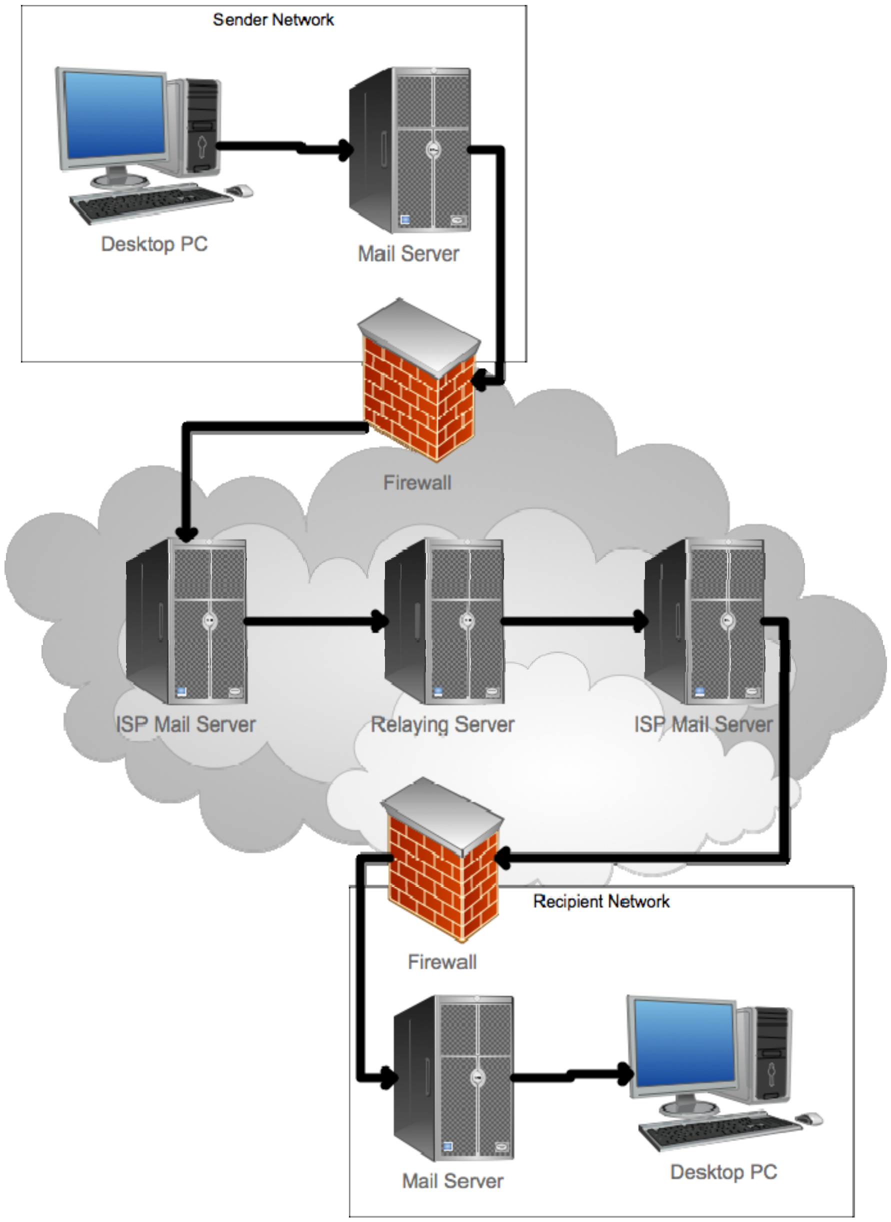


Figure 20.1: Typical email delivery.

The user's computer transmits the email to the company mail server. That passes the message out—generally through a firewall or proxy server of some kind. The message travels on to the company's ISP mail system in many cases (although some companies' mail servers do attempt direct delivery). The email may bounce through one or more relaying mail servers before it reaches the recipient's ISP mail server, passes through the recipient's firewall, into the recipient's mail server, and finally into the recipient's computer.

Every step along the way can keep a copy of the message. Even if encrypted, there can be copies of the message lying around who-knows-how-many servers on the Internet. That's not even counting the number of routers and other devices the message's Simple Mail Transport Protocol (SMTP) traffic must pass through, and *any* of those devices could transmit copies of the message to anywhere. SMTP traffic is rarely encrypted, so even if the email itself is encrypted, a snoop could see that there *was* a message, who it was intended for, and so forth. Now, let's contrast that with Figure 20.2, which shows how an MFT system talking to another file transfer server (or Managed File Transfer—MFT—system) would handle the transfer.

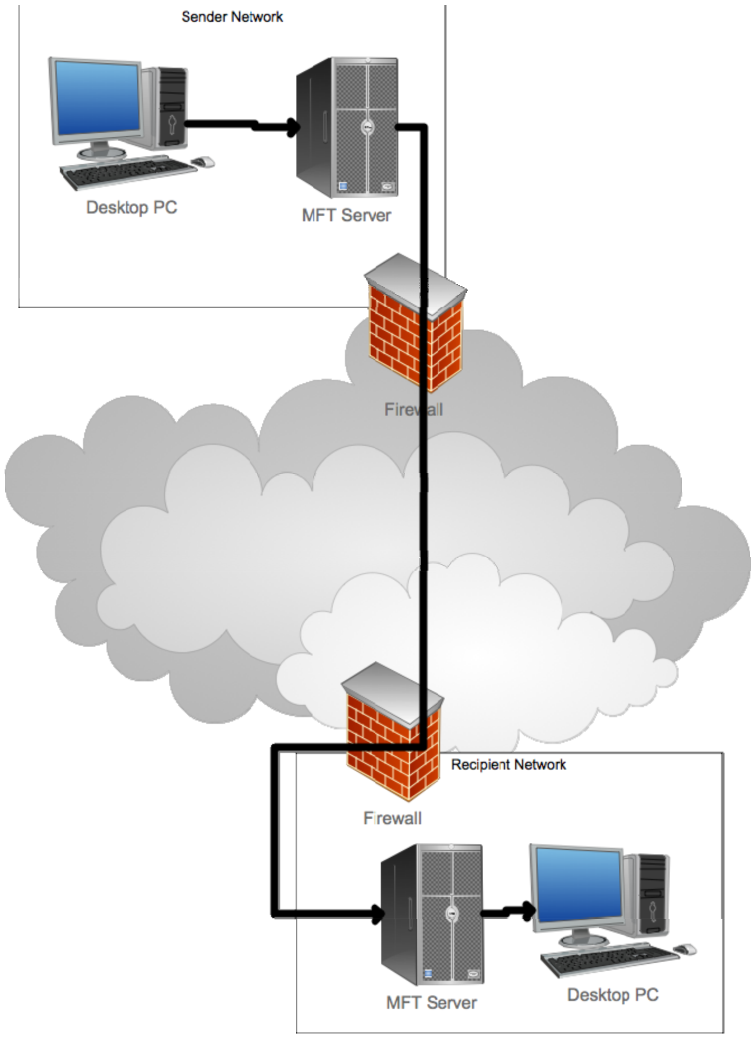


Figure 20.2: Transferring more directly via MFT.

The sending MFT system establishes a direct connection to the receiving system. That connection *may* pass through a firewall, but the firewall in this case won't be relaying the message or even seeing it, as the *entire communications channel* can be encrypted. Routers and other devices will certainly need to be involved in this transfer, but because the entire channel is encrypted, no other device can even see what's going on. For that matter, intermediate devices wouldn't even be able to distinguish between multiple file transfers occurring between the same sender and recipient at the same time, making it impractical to piece together the traffic and attempt to decrypt it.

So, no, email is not "more secure" for transferring data; the reason is that it simply doesn't typically rely on encrypted communications channels (even if the message itself is encrypted) and because too many players are involved. A point-to-point, MFT-managed transfer *is* more secure.

Tip, Trick, Technique 21: We're Using a Free FTP/SFTP Server. Why Would We Want a Managed File Transfer System?

Well, maybe you wouldn't. Let's discuss it. FTP stands for *File Transfer Protocol;* the "S" in SFTP adds "Secure," meaning the FTP channel is usually encrypted by SSH or SSL (there are a bunch of varieties of "Secure" FTP that we won't go into here, and some folks do improperly refer to all of them as "SFTP" as a kind of verbal shortcut). MFT stands for *Managed File Transfer.* Two words in common with FTP: *File* and *Transfer.* So in either case, you're obviously moving files from one place to another. With MFT, however, you're getting management for that transfer.

In other words, if absolutely all you need is to move a file from one place to another, your free FTP or SFTP server is probably just fine. However, if you want to apply management to that transfer process, you probably want to upgrade to a full MFT system.

Capabilities Available in an MFT System

What kind of management would an MFT system offer? There's a whole list of capabilities generally found on most decently-made MFT solutions:

- Guaranteed Delivery. The idea here is that you're not just throwing the file out to the Internet and hoping for the best. You actually get confirmation that the file got to its destination in one piece, and if the transfer is interrupted for any reason, the MFT system can automatically resume and/or retry.
- Non-Repudiation. Just because *you* know you've transferred a file doesn't mean the person on the other end will agree. They might claim the file was damaged in transit, for example, or might even claim that the data looks different than you know it did. Non-repudiation removes that chance by providing evidence that the file arrived in one piece, often through means like cryptographic hashes, signatures, and the like.
- Auditing. Know who transferred what, when they did it, how it was sent, when it was received, and so on. In companies burdened by regulatory and industry compliance requirements, this kind of auditing isn't a nicety, it's a necessity.

- *Certified* Military-Grade Encryption. It's all well and good for a free FTP server to claim that it uses 256-bit AES encryption (or whatever), but have they *proven* that their algorithms are operating according to the standard? With FIPS certification—something many better MFT products carry—you *know* that the encryption is actually up to snuff.
- Workflow. It's rare that the business process stops when a file transfer completes. Usually, something needs to be done with the transferred data—or something needs to be done to *get* the data that needs to be transferred. In either case, an FTP server isn't going to help you much, but an MFT system *can* help orchestrate the business processes that initiate or depend upon the file transfer. In some MFT systems, that can even include translating or transforming data so that it can connect to the necessary systems.

More Protocols Than Just FTP

In addition, the acronym MFT, unlike FTP or SFTP, doesn't refer to a particular *protocol.* That's because most MFT systems support *multiple* protocols—FTP, SFTP, FTPS, HTTP, HTTPS, SCP2, AS2, AS3, and many, many more. So on top of all those management features, you'll also pick up a lot more flexibility, enabling you to exchange data with a wider variety of systems and partners without anyone having to jump through hoops to support some weird new protocol that they've never heard of. Keep in mind that some of an MFT system's management features *require* the use of one of these protocols (FTP, for example, is a pretty awful protocol when it comes to any kind of management or security, so if you want to use management or security features you probably won't be using FTP).

The Business Connection: Cost, Reliability, and More

In addition to the feature set offered by MFT, there's something more to consider: How important is this file transfer to your business? In other words, consider this: If you needed to get an important business document to your company's legal firm, how would you do so? Would you trust the document to a bonded courier or to a well-known document-shipping company? Or would you hand the document to some random stranger passing your company's office, with detailed instructions on what to do with the document and perhaps a $5 bill for their trouble?

I'm pretty sure I know the answer, so here's the follow-up question: How is a free FTP server any different from entrusting your data to a passerby? You know nothing about the code inside that FTP server. You know nothing about the people who wrote it. They have no liability for their product operating correctly or even reliably. I know of at least one instance where malicious programmers took a popular, open-source FTP server solution and modified it (which is legal with open source code) to send copies of every transferred file to themselves. They got quite a few people to download their version of the software. Where exactly did your free FTP server come from?

And what does that "free" solution actually cost you? How many do-it-yourself scripts, hacks, and add-ons are needed to fit that server into your business processes? How much time does someone spend babysitting those processes, scripts, and hacks? How much time is actually spent supporting the "free" solution, and what does that time cost? MFT systems historically show a pretty low total cost of ownership (TCO), meaning that once implemented, they require little in the way of day-to-day care and feeding. Unlike the guy who wrote those hacks and scripts to make the free FTP server fit in, the MFT solution isn't going to get hit by a bus, go on long vacations, or decide to leave the company for greener pastures.

You're Buying a Relationship, Not Software

This discussion actually brings up a good point: When you decide to use a piece of software, whether commercially-produced or free, you're also deciding to enter into a relationship with the authors of that software. Are they the kind of people you'd like to do business with? Will they be around in 5 years when you need support? How quickly will they address any problems you might find in the software?

What, frankly, are their motives for helping you should you need their help? Are they helping you because you paid for a support contract that legally requires them to help you—or are they helping you because they're "passionate" about the free software they've produced? How long will that passion last?

This isn't intended to be a dig on open source or other free software, but these *are* valid business questions. After all, some open source software *isn't* free, and it *is* supported by companies who are committed to supporting it. Your company may have numerous on-staff experts who are comfortable providing you with all the support you'll need for an open source project, so you're not relying on the goodwill of that project's authors. So long as you can confidently and comfortably answer the question "Are you comfortable being in a business relationship with the authors of this software?," then you're fine. But you *have* to ask, and answer, that question.

One good exercise is to create a flowchart of exactly how data gets from one system to another. Figure 21.1 shows one that I helped a recent consulting customer put together, based on their own homegrown solutions. Looking at the flowchart, they realized how cobbled-together the system was. I asked them to start making little tick marks next to each component every time it failed or needed human attention—that's the version you're looking at in Figure 21.1.

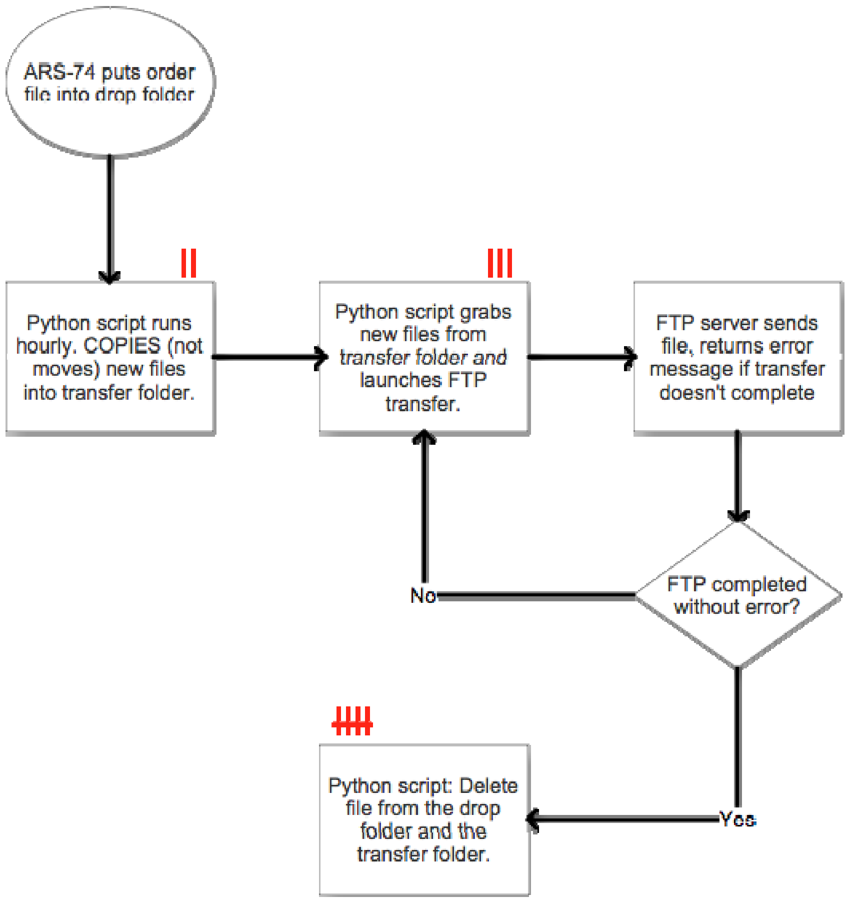

ARS-74 puts order file into drop folder
II
Python script runs hourly. COPIES (not moves) new files into transfer folder.
III
Python script grabs new files from transfer folder and launches FTP transfer.
FTP server sends file, returns error message if transfer doesn't complete
FTP completed without error?
No
Yes
卌
Python script: Delete file from the drop folder and the transfer folder.

Figure 21.1: Documenting your DIY process.

What the customer realized in this case is that their do-it-yourself (DIY) process had a few more moving parts than they were comfortable with, and that they really were spending more time supporting those cobbled-together scripts than they had realized. The problem most organizations have is that the folks who created this script and other hacks often just support them "in the corners," meaning the time isn't officially accounted for in any way, so it's really difficult to see how much it's costing. By forcing themselves to document the process, the moving parts, and the maintenance, they realized that their reliance on the "free" solution wasn't, in fact, all that free—nor was it something they remained comfortable with for their mission-critical business processes.

Tip, Trick, Technique 22: We're Evaluating Managed File Transfer Systems. What Should We Be Aware of as We Compare Different Solutions?

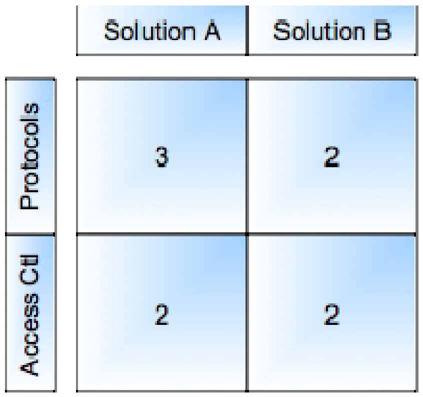
This tip falls under the category of "becoming a better shopper." It can be incredibly difficult to do a real apples-to-apples comparison between solutions just by looking at vendors' Web sites; vendors, quite naturally, want to highlight the good bits of their products and aren't necessarily trying to align to *your* exact business needs.

When you get them on the phone or into your office, of course, is where you get specific about *your* needs. I find that my customers' needs typically break down across six distinct categories. By focusing your evaluation criteria in these categories, you can help yourself do a better apples-to-apples comparison between different products.

I'll often put together a very simple grid, like the one I've started in Figure 22.1, to help compare solutions. For each major business need, I'll give each solution a score from 0 to 3:

- 0: Doesn't meet our need at all
- 1: Doesn't directly meet our need but offers workarounds
- 2: May meet our need with some customization or workarounds
- 3: Directly meets our need

Ratings scales of 5 or 10 or more get too subjective, so I like to keep my scale short. This makes it easier, at the end, to see which solution I consider the best for my situation.



	Solution A	Solution B
Protocols	3	2
Access Ctl	2	2

Figure 22.1: Starting a comparison grid.

Now, let's look at my six major comparison categories.

Protocol Support and Parallel Protocols

The ability to handle not only every protocol you need, but every protocol you think you might ever need in the future, is a big one. A score of "3" is earned by products that do everything I currently need *and* offer a good selection of additional, common, standard protocols.

You also need to dig a little to determine whether a solution has any restrictions on operating *multiple protocols at once*. It's great if a solution supports FTPS, HTTPS, and AS2; it's less great if it can only do one at a time. That's not a piece of information often brought out on vendor Web sites, and it's something I've been burned on a couple of times, so I always make a point to ask about it.

I generally consider the following to be a "complete" set of standard protocols—anything less than this set gets a lower score from me:

- AS1, AS2, and AS3, which are Electronic Data Interchange (EDI)-specific protocols
- HTTP and HTTPS
- SMTP and POP
- FTP and its various secure variants, particularly FTP over SSL (FTPS) and FTP over SSH (usually referred to as SFTP)
- CIFS and/or SMB, which is sued to access local network file shares

Because cryptography comes into play with many of these protocols, you're also going to want to look at the products' crypto certifications. I look for FIPS 140-2 validated crypto modules and standard AES encryption capabilities. It's becoming common to also see OpenPGP used for file encryption, so if that's something you want, look for it—some vendors offer it as an optional module.

Access Control

You don't want just anyone sending and receiving files, so the ability to control access to the MFT system is pretty important. You need to decide how you want that access controlled, though: Do you want the MFT system to have its own user database or do you need it to integrate with something like Active Directory (AD)—or would you prefer a mix of the two?

Access control should be pretty granular, too. You should be able to delegate control over specific tasks, MFT servers, and so forth to specific users or groups. Administrative access to the system should be separate, and should ideally take place over some kind of encrypted communications channel.

Auditing

These days, pretty much any system—such as an MFT system—that touches data should be expected to log its every interaction with that data. That can become complex in systems that also offer powerful workflow capabilities, but it's simply a must-have feature.

Every time someone changes something in the MFT system—adding a user, changing a configuration setting, whatever—and any time the MFT system executes any action, there should be a log entry in some kind of tamper-evident auditing database. Ideally, in MFT systems that span multiple servers (such as for workload distribution or high availability), that audit log should be centralized and shared by all MFT servers so that you can get all of your auditing in a single place.

Some MFT systems might accomplish this by simply sending log messages to a RADIUS or syslog server, and that's fine. Other vendors choose to implement their own proprietary auditing, which is also fine if it meets your needs.

Process Automation and Workflow

This is where the "managed" part of MFT really comes in, and it's something you should pay close attention to. There are basically four broad ways in which an MFT system can let you build automated business processes:

- By programming against an application programming interface (API) provided by the MFT system. This might be as simple as writing system scripts, or it might involve full-on software development.
- By writing scripts in a proprietary scripting language, or in a standard scripting language that's been specifically extended to support the MFT system. This is often easier than API programming while still offering a high degree of flexibility and power.
- By "clicking together" pre-made process task units. Essentially, the MFT system gives you a library of things it can do (the larger that library, the more flexibility you have) and you stack the tasks up in whatever order you need to match your business process. This requires little or no programming, but you're limited to doing only what the pre-made task units can do.
- By using pre-built, mildly-customizable process workflows. This is the least flexible of the three techniques but is suitable for simple processes.

None of these is inherently right or wrong, and a really good product will provide more than one of these so that you can provide customized process automation in a variety of ways. I find that *most* companies really want the third technique, which is usually implemented as a graphical "process builder," such as the one shown in Figure 22.2.

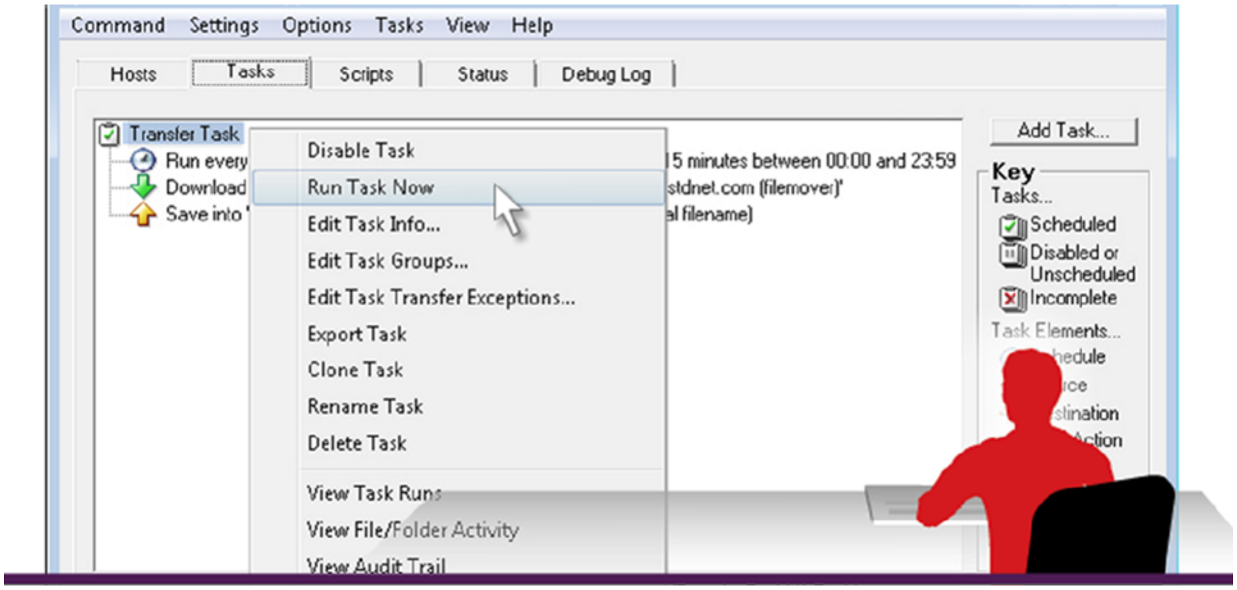


Figure 22.2: A GUI-based "process builder" for process automation workflow.

These kinds of builders are often accessible and usable by non-programmers and provide a faster way to get an MFT system up and running. I don't like not having the option to do full custom programming via scripting or an API, though, because you never know when some weird need will come up that the process builder's task library doesn't accommodate. As I mentioned, though, it's not unusual for MFT systems to provide a process builder *and* an API or scripting language of some kind:

Ask questions about the solution's ability to execute these custom processes:

- Can it run more than one process at a time?
- Can a given process run multiple tasks in parallel, such as transferring multiple files at once?
- Can tasks be run on a schedule, in response to a predefined event (like a file showing up in a drop folder) or on-demand?
- Can tasks be built with conditional logic so that they can adapt to different anticipated conditions?
- Are basic tasks—such as synchronizing files between two systems—easy to set up, or would they require custom scripting or programming?

Checkpoint Restarts, Reliability, and Guaranteed Delivery

Next up are what I consider the core features of any solid MFT system. First, does the system support guaranteed delivery and non-repudiation? Any MFT system should, and I've discussed in several other tips why those are important.

Second, can the system easily restart a failed or interrupted transfer? Restarting from a checkpoint is especially useful when you're sending large amounts of data because restarting the transfer from scratch each time will obviously take longer.

Third, ask if the system can be built to be highly-available. Even if you don't need that capability immediately, you'll want the option for the future. Ideally, an MFT system can be built across two servers that can load-balance each other and provide redundancy for each other. This may be packaged as a separate option, which is fine; you can decide whether you need it right away.

Centralized Policy-Based Management

Centralized policy-based management is becoming a bigger and bigger deal for companies, especially those who have regulatory or industry rules to comply with. Essentially, you want to be able to configure all of an MFT's major settings in one, central location—establishing an operating *policy* for the system, in other words. That top-level policy should then apply to any MFT servers subject to the policy, and the policy should be *enforced.* In other words, if an administrator reconfigures a single MFT server, the policy should step in and remediate the configuration, taking it back to the top-level, policy-based standard.

Auditors *love* this kind of setup because they only need to audit *the policies themselves,* rather than checking up on each and every server you have. Policy-based management also makes for a more agile system. Need to change something to meet a new business need? Just change your policy; doing so causes the system to reconfigure all of your servers automatically. So you're using policies to manage the *business service,* not individual pieces of hardware and software—much easier, much faster, and much more intelligent.

Tip, Trick, Technique 23: We've Created Our Own Managed File Transfer System Using Several Tools. Why Should We Consider an Integrated Managed File Transfer Solution?

Start by reading Tip 21 from this book, which covers some of the downsides of using free solutions in combination with do-it-yourself (DIY) scripts and tools. Much of what I wrote there will apply to this question as well. This question, however, isn't specifically about DIY tools or free tools, so there are some other considerations. Specifically, what's the difference between an integrated Managed File Transfer (MFT) system and a collection of non-integrated tools that perform the same tasks?

Holes Between Pieces

It's unusual for a collection of non-integrated tools to completely fill every need you've got—but once you're using those tools, it's difficult to spot the missing functionality simply because you've become accustomed to it. As an exercise, I'll work with my consulting clients to help them visualize all the pieces that go into their file transfer processes. Figure 23.1 is one that I developed with a recent customer. The idea is that the outer, red circle represents their entire need, and the white circles represent individual tools and elements that they've built or acquired.

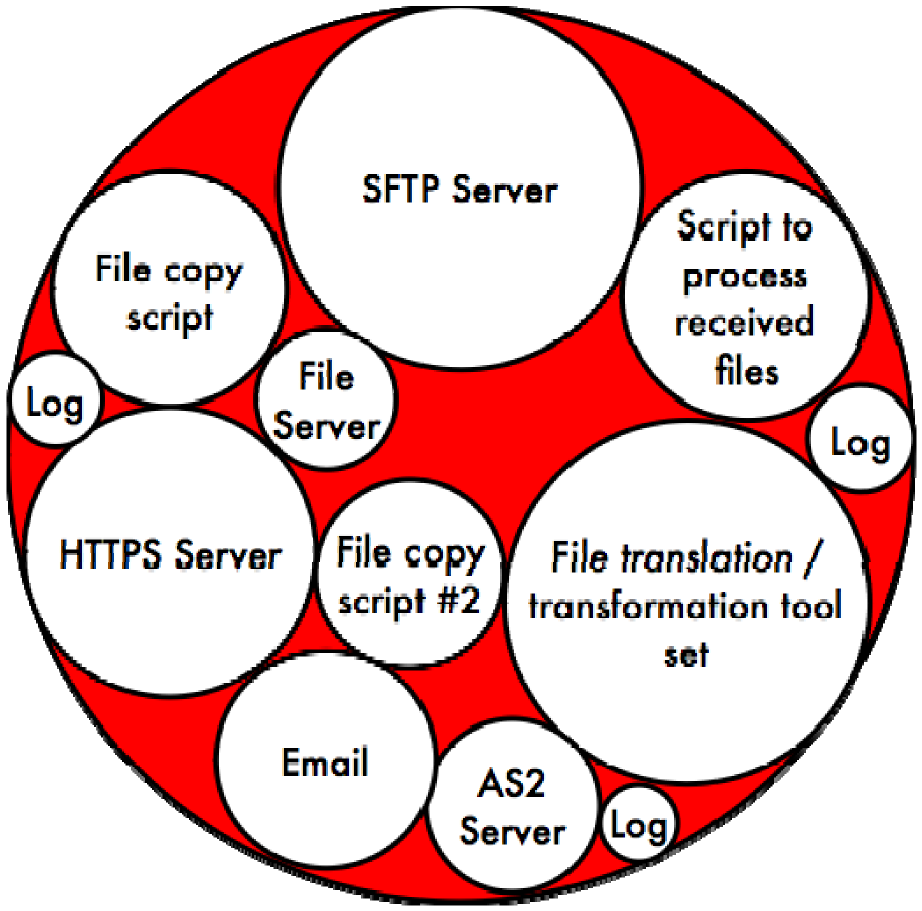


Figure 23.1: Mapping functionality to find "holes."

Each white circle's size is based on how much it's used within the overall system (so, it's relative importance), and the circles are positioned next to related components. For example, the HTTPS server has a logging component so that log is shown next to the HTTPS server's circle. Obviously, being circles, the white circles won't completely cover all the red. That's where we start discussing what the system doesn't do.

It was in this discussion that the customer acknowledged that the SFTP server they were using didn't create any kind of audit log—something that troubled them but that they'd simply become used to. We also noticed that they were really managing several logs from different systems—and admitted that it was a bit painful to construct the reports their security auditors needed.

I pointed out the DIY components—a handful of scripts—that were providing connections between these components they'd acquired. We talked about the support costs of those scripts. Management tended to think of those costs as "zero" because management didn't see an invoice for those costs, but the administrators who wrote those scripts pointed out that the scripts almost always had to be rewritten every time one of the connected systems was patched or upgraded because script interfaces changed. We estimated the time on that, and that "admin time" started to fill in some of the red.

We also looked at elements of the system that were perhaps less-than-perfect. For example, nobody in these discussions ever points out email as a means of file transfer, so I always make a point of asking about it—and it usually winds up being a pretty big circle. But it's not logged, and not managed, and not terribly secure, so it's sub-optimal—and that "less secure" note, for example, accounted for more of our "missing functionality" red space. Figure 23.2 shows how the diagram evolved, as the team started to realize some of the downsides to their pieced-together approach.

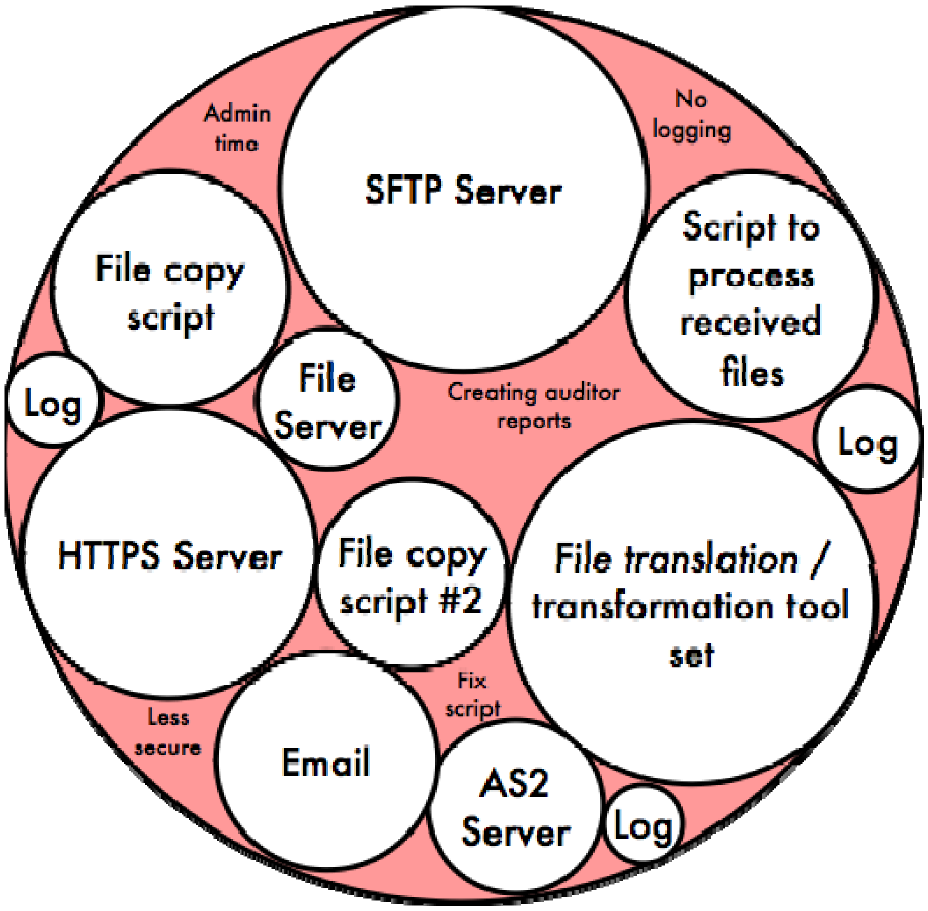


Figure 23.2: Filling in the "missing pieces" with what's really getting the job done, and with weaknesses and unfulfilled needs.

This is one way to look at an existing system to become aware of its weaknesses. You can start to build a better list of business requirements, and begin to understand how a single, fully-integrated system can better meet your needs—often at a better overall price.

Integrating the Pieces: A DIY Effort

The diagram also helps to highlight how much work it can really be to help integrate non-integrated systems. A better approach is to construct a diagram like the one in Figure 23.3. Here, I've walked my customer through the various components involved in sending drop-ship orders, which are entered into their order-management system, to the external vendors who will fulfill those orders.

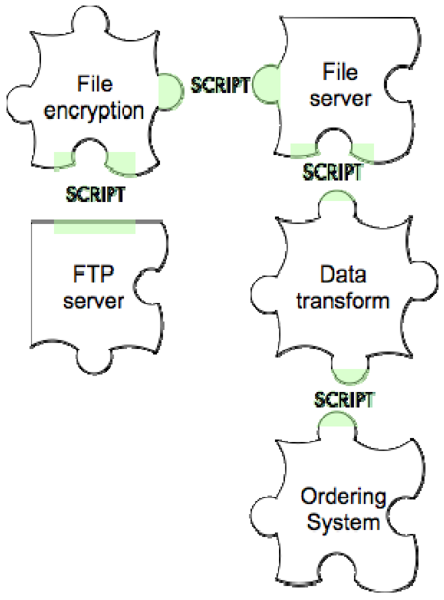


Figure 23.3: Connecting the non-integrated pieces of the "solution."

Then I walk them through the process of discovering how the systems communicate with each other. A script pulls data from the ordering system and feeds it to a data-transformation tool they bought. Another script takes the output and copies it to a file server. A third script encrypts the file before sending it to the FTP server, which sends it to the business partner. That's a lot of moving parts—and that's where we'll start the discussion of how much time it takes to maintain all those connecting pieces. We'll also talk about business risk tolerance: Those scripts weren't, in this case, written by professional programmers. Reviewing some of them, I was able to immediately point out significant programming flaws that could make the scripts perform improperly—and the script's current owner admitted that some of those situations had, in fact, happened.

I want to clearly point out that all of the *components*—the ordering system, the data transformation, the file encryption, the FTP server—functioned flawlessly. The problem inherent in a cobbled-together system isn't in the cobbles themselves but in the mortar (the scripts) used to integrate them.

Operational and Maintenance Considerations

What really astounds me is how some IT teams can argue both sides of any given issue with great skill. For example, I'll ask how many client operating systems (OSs) they're currently supporting in production. "One" is a very common answer, and "mostly one, but sometimes two for some situations," is also common. When I ask why, the invariable answer is that they don't want to spend more money supporting multiple client OSs. Folks in shops that have Windows, Macs, Linux, and Unix invariably point out the higher expense of supporting all those systems.

When I then ask why they're supporting a half-dozen (or more) discrete elements to form a "managed file transfer" system, they'll tell me *it's cheaper.* It often takes some extended discussion to point out that *by their own logic* a single system costs less to support. You get one set of patches from one place. You learn one admin interface.

The difference is that, in a multi-component hodgepodge, there usually isn't any one person in charge of the whole thing—so each person tends to see themselves as supporting only one solution, or maybe two. The problem is that you've got a bunch of people supporting the actual "solution."

A fix in most of those cases is to change the way the company tracks their IT team's time. Rather than logging tickets against "the FTP server" or "the mail server," I have them log tickets against "the file transfer system," with sub-categories for the individual components. That permits all the here-and-there effort to be consolidated into a more realistic picture of time expended—and it usually doesn't paint a pretty picture.

Pricing Considerations

Finally, I ask how much they think all this really costs. Accounting for the licensing costs of the individual solutions, accounting for the time spent patching and managing those systems (once they learn how to keep track of that), and accounting for downtime when some piece doesn't work properly, they realized they spend a great deal more than they would have spent on a fully-integrated MFT system.

To be clear, it's nobody's fault. These scenarios evolve over time and in response to business needs, and the need of the moment never seems big enough to justify moving to an "expensive" integrated MFT solution. Over time, though, all those "cheap" point solutions add up to a bigger cost than just going with the MFT system. This is why it's worth looking at your exact environment—it might be costing you more than you thought.

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.