Realtime publishers

Tips and Tricks Guide™ To

Managed File Transfer

sponsored by

IPSWITCH FILE TRANSFER

Don Jones

Copyright Statement

Volume 5

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 24: How Have Different Organizations Approached Managed File Transfer? How Have They Used It?

The answer to those questions varies with different levels of sophistication within different organizations. In other words, there are several levels to which I see Managed File Transfer (MFT) being used within organizations. Each successive level encompasses, or is a superset of, the one before. I've named these five levels operational, automated, integrated, service-oriented, and proactive. They're a great illustration of how your most immediate file transfer needs can lead to a more agile, responsive business.

See where your organization fits in. It should probably go without saying, but I'll say it anyway: The more sophisticated your company, the more flexible and complete an MFT solution you're going to need in order to really fulfill your company's expectations and requirements.

Operational

At the operational level (see Figure 24.1), the company is primarily concerned with operational issues. Can the organization securely connect to partners to transfer files? Can delivery of those files be guaranteed? These are typically the bare-minimum requirements in organizations that have file transfer needs. Note that the main emphasis is on *security* and *guarantee*—we need the data to remain private, and we need to make sure it gets there. In most cases, transfers are set up and managed manually by an administrator.

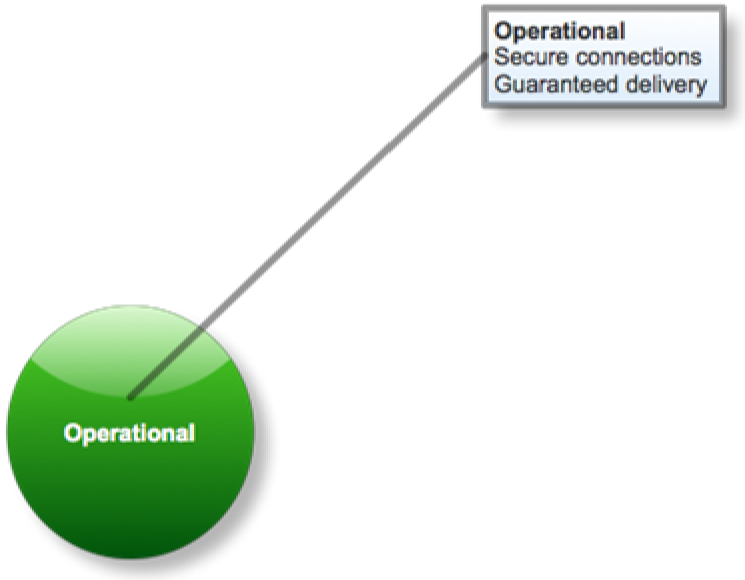

Figure 24.1: The operational level of file transfer sophistication.

Automated

At the next level, the company has set up their file transfers and is now looking to reduce the amount of manual overhead involved in managing them. As Figure 24.2 shows, the organization is now looking to automate, reducing the time administrators spend on file transfer as much as possible. This step also includes enabling user-to-user transfers, taking administrators out of the loop entirely when internal and external users need to exchange files. This also often includes a lot of reusability within the MFT solution so that a single connection profile (for example) can be used across multiple connections.

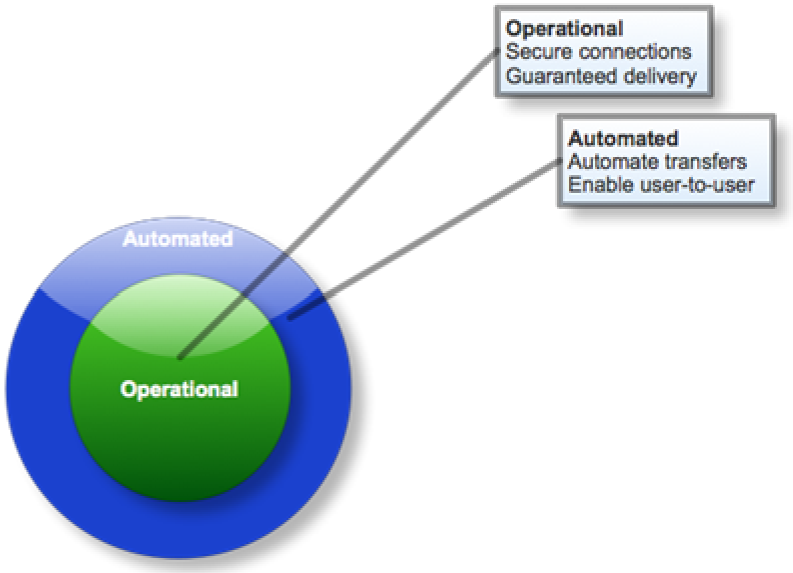

Figure 24.2: The automated level of file transfer sophistication.

Integrated

At the integrated level (shown in Figure 24.3), the organization is moving beyond even automation—which is where *many* folks see MFT itself ending. A more sophisticated organization can continue using MFT to *programmatically* trigger file transfers. That is, rather than responding to a set schedule for transfers, they happen automatically when an external process needs them to. Integrating with third-party business processes is also at this level as is a more mature approach to monitoring: Whereas lower levels of sophistication are often satisfied with basic server- and service-level monitoring, organizations at the integrated level often want to track individual file transfers, frequently for auditing and compliance purposes.

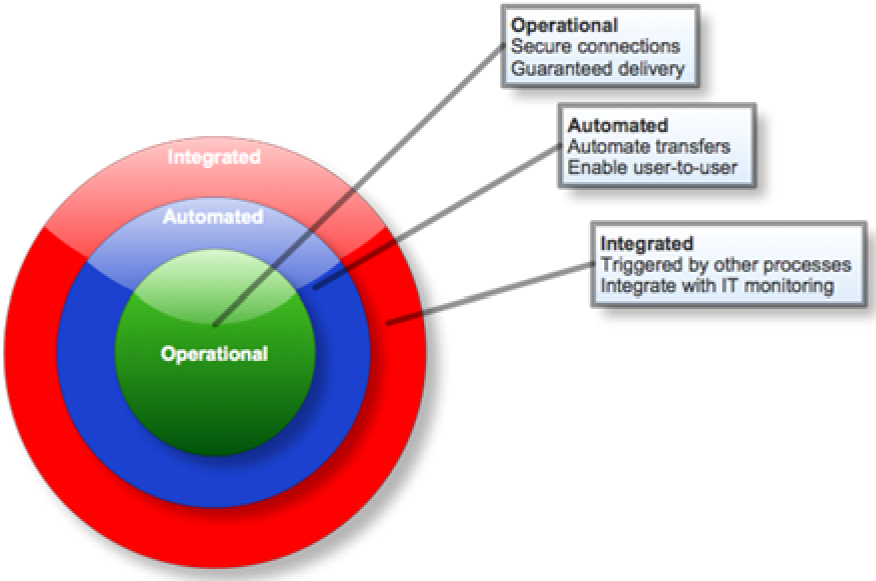

Figure 24.3: The integrated level of file transfer sophistication.

Service-Oriented

An organization at the service-oriented level of sophistication (see Figure 24.4) wants to make file transfer a service within their environment. They don't want to actively monitor the solution; instead, they're looking to be primarily hands-off. They rely on MFT solutions that can deliver alerts, provide monitoring dashboards, and so forth. MFT is a service provided to end users, as well, much like messaging, file storage, and so on. The organization will often be interested in allocating the cost of the MFT service across different partners, users, or departments. Finally, many provisioning activities—such as provisioning new users or partners—will be automated as part of the MFT service, further reducing the manual overhead commonly associated with file transfer management.

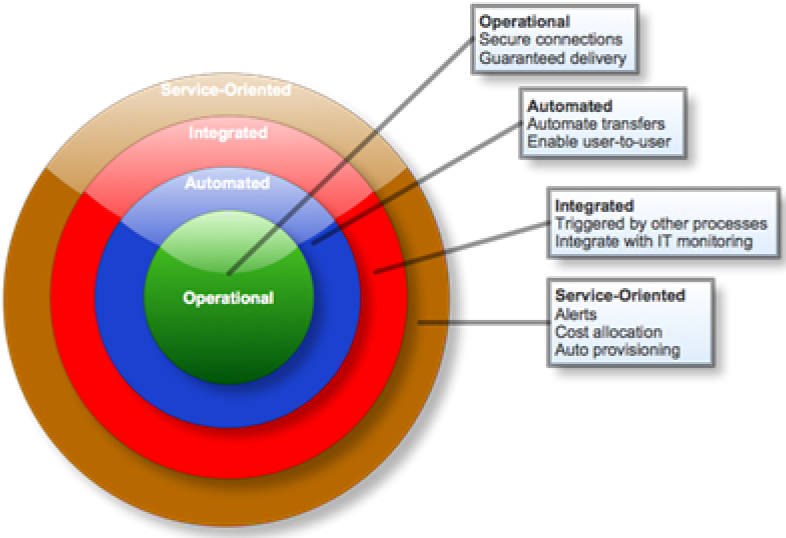

Figure 24.4: The service-oriented level of file transfer sophistication.

Proactive

At the top level of sophistication (see Figure 24.5), organizations want to provide metrics about file transfer not only to themselves but also to partners and customers. They want policy-based control over changes, improving their change-management posture. They also want a system that can help them recognize opportunities in their data so that they can proactively further their business. In many cases, these organizations will have business intelligence (BI) solutions in place, and will be looking to integrate many of their corporate data stores with those BI solutions.

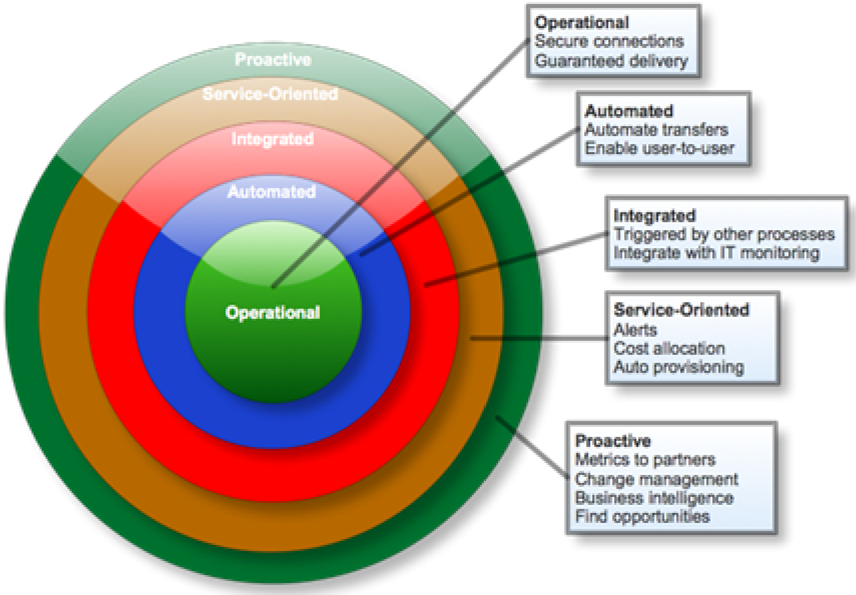

Figure 24.5: The proactive level of file transfer sophistication.

It's important to recognize where your organization falls within this spectrum—and to understand where it's likely to go in the near future. That will help you understand what types of MFT solutions are most appropriate not only for your immediate project needs but also for the foreseeable future.

Tip, Trick, Technique 25: What Kind of Threats Do I Have to Worry About When I Have an MFT Solution?

Anytime you connect a server to the Internet, you're going to have to concern yourself with how it might be attacked, or how it might become a conduit for attack. Managed File Transfer (MFT) solutions are no exception.

MFT as a Victim

The primary ways in which MFT solutions are attacked are *hammering* and *DoS/DDoS* attacks. *Hammering* is exactly what the idiom implies: Sending so much traffic to the server that it fails to respond to legitimate requests. Hammering is actually the way in which *Denial of Service (DoS)*, and its more effective cousin, *Distributed Denial of Service (DDoS)*, attacks are conducted.

As Figure 25.1 shows, the idea is to have a central controller for the attack. In a typical DDoS attack, that controller will have control over hundreds, thousands, or even tens of thousands of end nodes, or *zombies,* which act under its orders. The zombies are a simple model of distributed computing: By having more end nodes sending traffic to the victim (in this case, an MFT server), you stand a better chance of overwhelming the victim and denying service to legitimate users and processes.

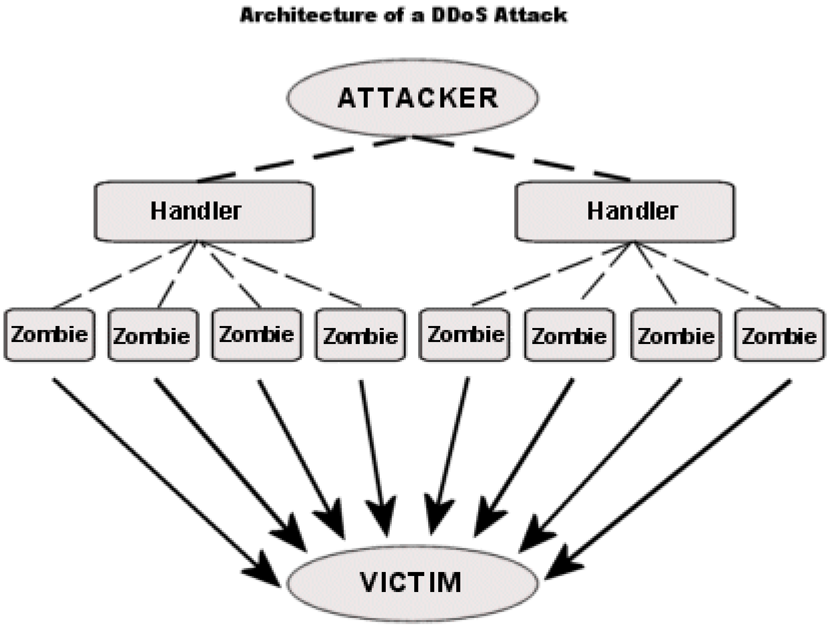

Figure 25.1: Architecture of a DDoS attack.

DDoS attacks often begin life as viruses, which work to distribute the zombie code to nodes across the Internet. On a given date and time, or at the transmitted order of the controller, the zombies spring into action and begin attacking their target.

One approach to combating this kind of attack is to use an intelligent firewall—or, more commonly, arrays of firewalls acting as one—to intercept incoming requests and discard those that are felt to be attacks. Firewalls can often do so much faster than a normal server, making it more difficult to overwhelm the firewalls. Only legitimate connections are passed on to the MFT server, as Figure 25.2 shows. Numerous techniques are often used to detect attacks, such as too many requests in a short period of time from a given source IP address.

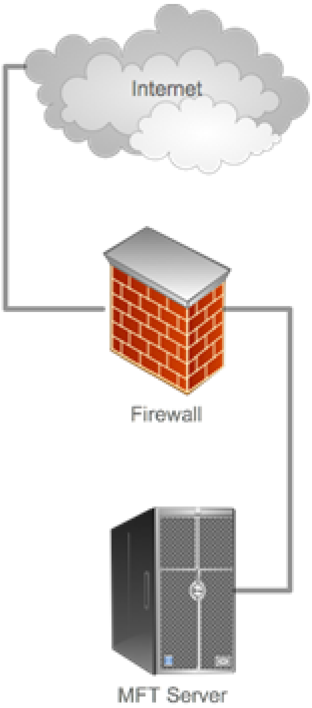

Figure 25.2: Protecting the MFT server with a firewall.

Smart attackers can sometimes bypass the generic protections provided by a firewall by carefully constructing their incoming traffic to *look* legitimate. In those cases, an intelligent MFT solution can provide its *own* anti-hammering protections. This can include techniques such as temporarily blocking connections from IP addresses whose traffic doesn't properly authenticate within a certain number of tries. An effective solution will send alerts to administrators when this happens, alerting them to the attack that is underway and permitting them to take additional protective measures if necessary.

MFT as a Conduit

Because an MFT server transfer data in and out of the organization, it—like messaging servers, in many ways—has the potential to be a conduit for viruses and other malicious code. This threat isn't immediately obvious to many people. "File transfer is used for data files, not executables," they'll often tell themselves, "and what's the harm in a data file?" That depends. Some data files *contain* executable code: Microsoft Excel spreadsheets, Access databases, and so forth are good examples. But MFT solutions are often used for user-to-user transfers that may indeed include executables. You have to move beyond thinking of MFT as solely a means to transfer data files between two servers, and realize that many organizations eventually use their MFT solutions for *all* file transfers, often even between internal users.

To help prevent that MFT system from becoming an unknowing conduit for viruses, it's important to select a solution that supports anti-malware scanning both on incoming *and* outgoing transfers. (After all, nobody wants to be the guy who *sent* a virus to a business partner or customer, right?). A few MFT solutions offer their own anti-malware scanning engines, which I dislike. I think most companies today have an anti-malware solution in place that they're comfortable with; to the greatest degree possible, an MFT solution should simply provide hooks so that the company's *existing* anti-malware solution can be triggered to scan incoming and outgoing files, and to quarantine infected files (and alert an administrator, of course, when that happens).

Figure 25.3 shows this arrangement. In some cases, it may simply mean installing anti-malware software right on the MFT server, and configuring the MFT solution to use the software. Other times, it may—as suggested in the illustration—involve moving the file to an external server, letting software on that server scan it, then moving the file along to its next destination.

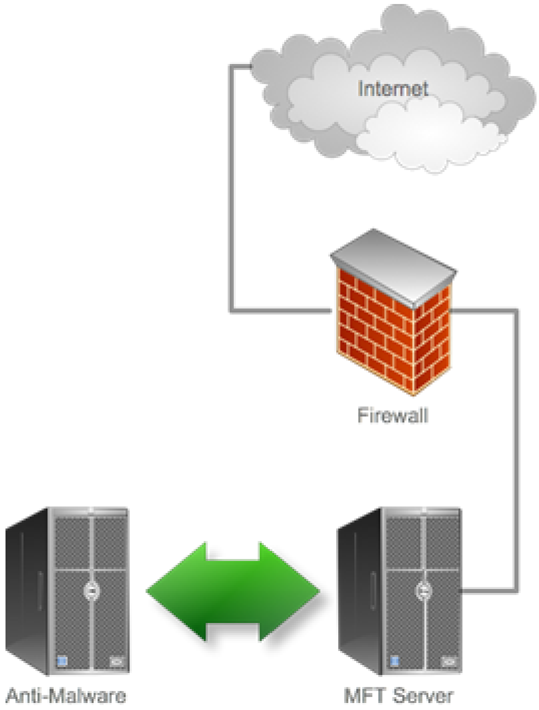

Figure 25.3: Protecting against anti-malware.

As you consider MFT solutions, it is *hugely* important that you understand what model they use, what kind of scanning throughput your anti-malware software can support, and so on.

Tip, Trick, Technique 26: We Have Numerous File Transfer Solutions in Place Because Different Departments Have Different Requirements. What Are the Downsides to This Setup and How Can We Avoid Them?

In my consulting practice, this is perhaps the single most-common scenario that I see. Whether I'm going in to consult on directory consolidation, access management, database performance, or whatever, I can almost always rely on the organization having multiple file transfer solutions in place.

It's easy to see how it happens: One project or department finds a need for file transfer, acquires a solution, and begins using it. Another project or department comes along with slightly different needs, acquires a new solution that meets *those* needs, and begins using it. Before long, you have a plethora of solutions, all of which work slightly differently and all of which require individual management, monitoring, maintenance, and so forth.

The *best* solution is to not wind up in this situation in the first place, which you can do by looking beyond the initial project's requirements when acquiring your *first* file transfer solution. That said, if you find yourself with a fragmented file transfer infrastructure—that is, multiple solutions operating on different levels—what are you risking? How can you fix it?

The Downsides of a Fragmented Approach

The single biggest risk of a fragmented file transfer infrastructure is security. With multiple points of entry, you'll have a more difficult time consistently maintaining security within your environment. You'll have to audit each solution, and each one will likely provide different capabilities and features for doing so. You'll have to secure multiple entry (and exit) points to (and from) the network—meaning more complex configurations that are more difficult to monitor and maintain over time. File transfer servers move data to other servers in your organization, further compounding the problem and the security risk. Figure 26.1 shows how this situation can quickly get out of hand.

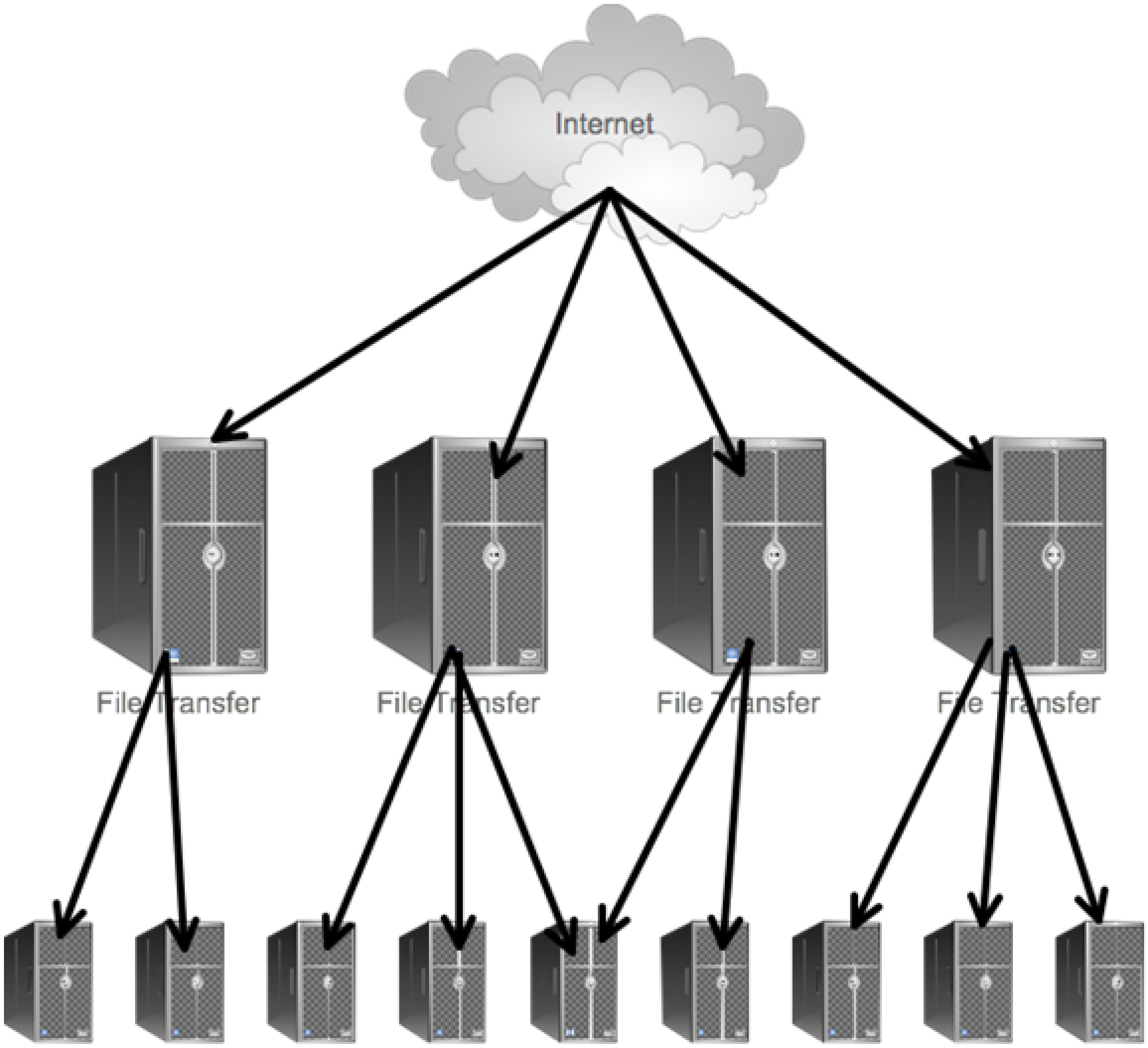

Figure 26.1: Multiple file transfer solutions means multiple security concerns.

And if you're thinking, "Well, the security isn't a concern because we've never been hit with a problem," don't worry. You will be, eventually. Everyone is. With any luck, you'll have moved on to a different company before that happens, because cleaning up after this kind of mess is tough. How do you even find out *which* conduit was used for a security attack or compromise when you have *so many* conduits in and out of the network? It's tough.

The other downside associated with a fragmented file transfer infrastructure is the one you probably *feel* more immediately: increased overhead. Every file transfer solution has its own configuration. Its own monitoring. Its own patches. Its own user database, its own file transfer rules, its own scripting language, its own, its own, its own. It amazes me: Administrators will vigorously defend a decision to use only a single client operating system (OS)—typically Windows—on the ground that maintaining one thing is less difficult and expensive than maintaining two or three different ones. Yet a company will happily implement a half-dozen file transfer solutions (I have one client with eight) from different vendors.

How to De-Fragment File Transfer

The answer is simple on paper: consolidate. Bring all file transfer into a single system that can meet *all* of the requirements for the *entire* organization. Yes, on paper, it seems like a waste of money. You're replacing a bunch of systems that are already working with one system that does the same thing.

You *will* save money, though. It's the "soft" money that companies rarely bother to measure: time, overhead, and security risk. Not to mention the easier-to-measure software maintenance fees you're probably paying to each file transfer solution vendor you've bought from.

A single solution, as Figure 26.2 shows, provides a single pipe between you and the rest of the world (a pipe which, with the right solution, can of course be made redundant and fault-tolerant). You get one place to configure. One place to customize. One to monitor. One to patch. One to secure. One to audit. Everything happens in one place, in one way.

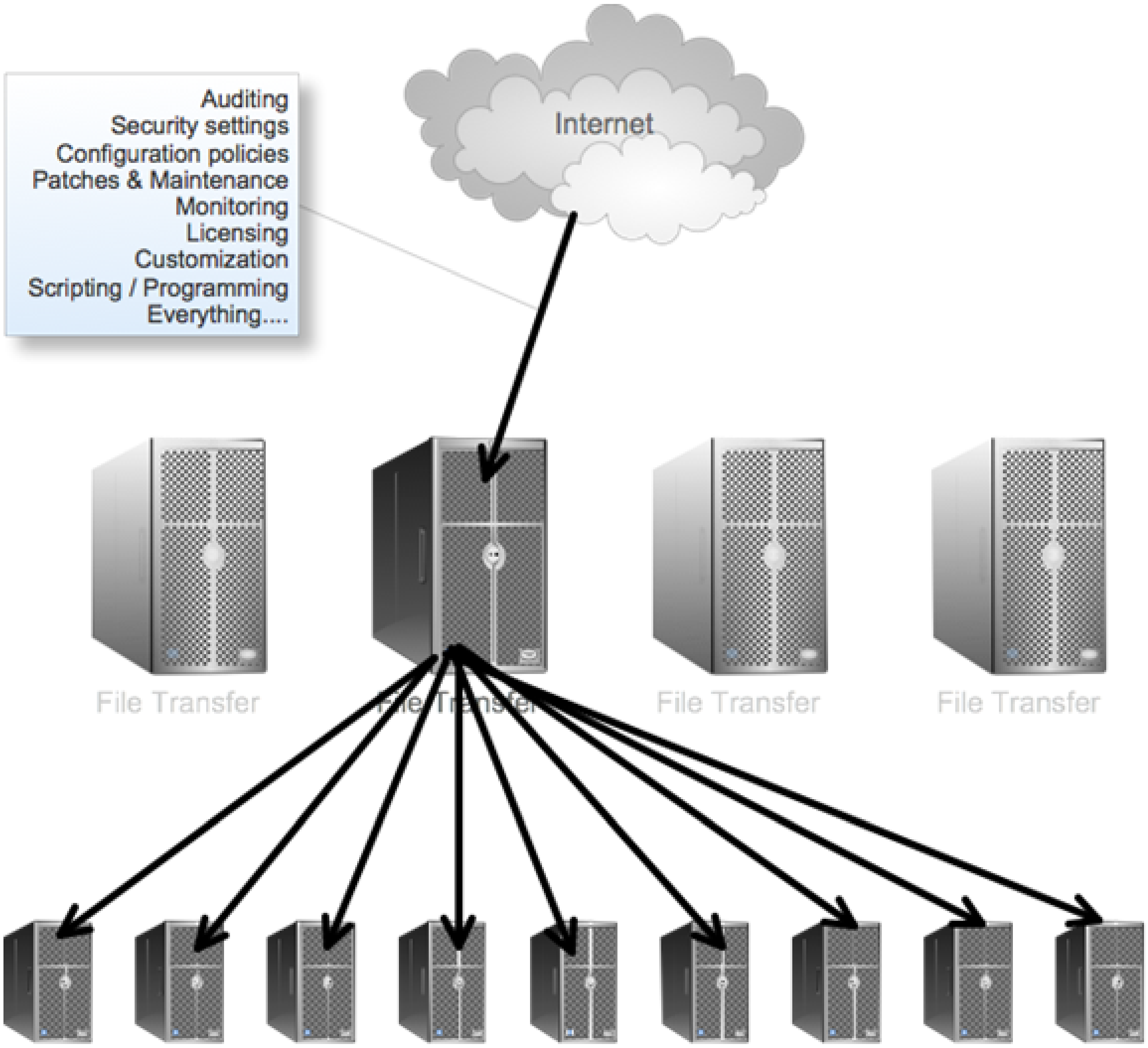

Figure 26.2: Consolidating file transfer solutions to a single platform.

When something happens, you don't have to go from solution to solution looking for answers. When a new need arises, you don't have to figure out which solution can support it.

In fact, some of my larger consulting clients have decided to offer file transfer as a service to the rest of the organization—just as they do with messaging, network infrastructure, directory services, and *nearly every other aspect of the network.* A single group—often a small "Managed File Transfer" team—is dedicated to managing fie transfer for the entire company. As important and ubiquitous as file transfer is—easily on par with email, in most organizations—this is the perfect approach.

Tip, Trick, Technique 27: Can a Managed File Transfer Solution Work with Electronic Data Interchange (EDI) and Other Forms of Data Exchange?

Usually, depending upon the Managed File Transfer (MFT) solution you're talking about, of course. There are numerous EDI standards out there, so you will have to be specific about which ones you want to use. Often, companies will think about EDI in application-level terms: A bank, for example, thinks about SWIFT codes and Automated Clearinghouse (ACH) transfers. The underlying *computer protocols* are often more generic, though. Secured FTP transfer of XML files is a common, modern means of EDI, for example. More commonly, protocols like AS1, AS2, or AS3 are used when "EDI," rather than plain-old "file transfer," is in play. As Web services continue to become more prevalent, HTTP and HTTPS are often mentioned in the same breath as EDI in many organizations.

If you're looking at an MFT solution to give you EDI capabilities, then it—as always—pays to buy the most flexible one possible. The following list highlights protocols commonly used in EDI exchanges:

- AS1—Built around the SMTP and S/MIME email protocols, this was the first AS protocol developed by the Internet Engineering Task Force (IETF)
- AS2—Built around HTTP and HTTPS as opposed to SMTP for transmission, AS2 includes increased verification through the use of digital signatures and receipts
- AS3—Built around secured FTP
- AS4—A newer protocol that is, as of this writing, still in draft form and is not yet widely used
- FTP and, more commonly, FTPS or FTPS—File transfer protocol typically secured through SSL or SSH channels
- HTTP and HTTPS—Hypertext transport protocol, which is the main transmission language of the Web

There's a bit more to EDI than simply file transfer, though. Typically, EDI standards are built around the concept of a document: an invoice, an electronic check, and so forth. Data must be formatted in a specific way so that each party involved in the data interchange can understand it. In-house systems are nearly never built to understand these intermediate standards, though, which means EDI systems often play as big a role in *translating* as they do *transferring* data. Figure 27.1 illustrates.

Data, often stored in a back-end database or mainframe system, is translated into the intermediate EDI standard form, then transferred using some EDI-standard protocol, which both parties have agreed upon.

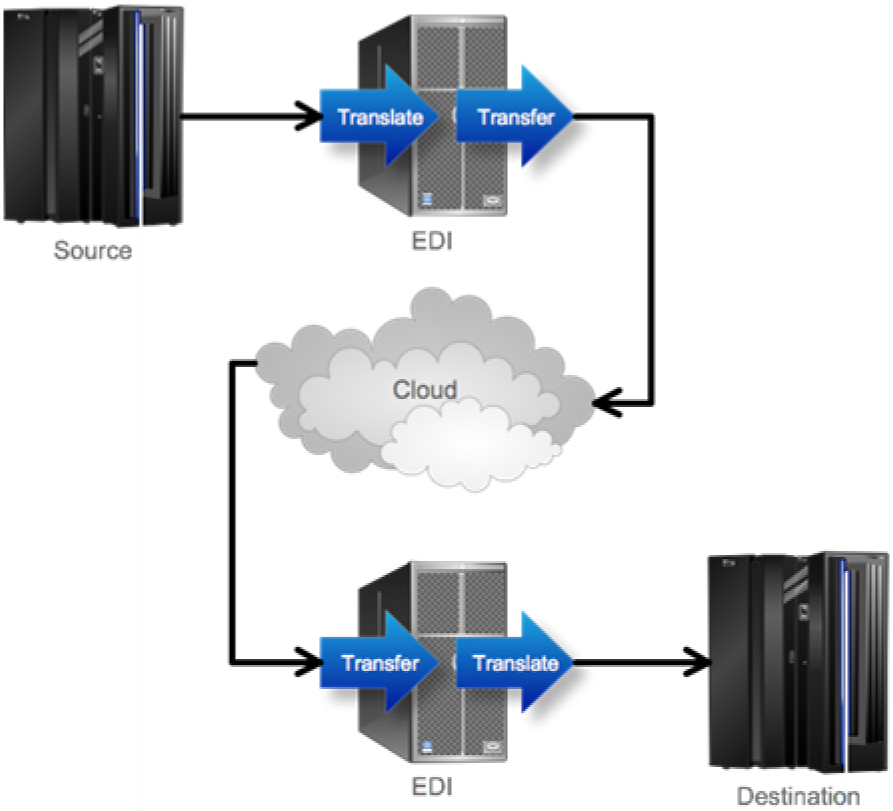

Figure 27.1: EDI often involves translation as well as transferal.

The “cloud” in this illustration might well be the Internet, but it also might be a private EDI hub, or even a private Wide Area Network (WAN) link that connects the two parties. EDI hubs were popular in the eighties and nineties before the rise of the Internet but have since played a less-important role as ubiquitous Internet connectivity has provided virtual-direct connectivity between nearly every organization on the planet.

Once received, the data is translated from the standard EDI format into whatever the destination—again, often a back-end database or mainframe system—requires. The EDI servers must handle the guaranteed delivery, security, and other aspects of the transfer as well as the customized translation from and to the company’s internal data formats.

Managed File Transfer (MFT) systems are increasingly taking on both roles. Translation has become such a commonly-used facility in EDI and other file transfers that it’s becoming an increasingly-included ability in MFT solutions themselves. In fact, data translation (or *transformation,* as it’s sometimes called, to differentiate it from the translation of human languages) is more and more seen as an inseparable component of full-featured, high-end MFT systems.

Tip, Trick, Technique 28: Can a Managed File Transfer System Help Ensure Clean Data Coming into My Systems?

It should be able to, and it should be able to do so in a couple of different ways. First, as I discussed in Tip 25, Managed File Transfer (MFT) systems are a potential conduit for malware traveling both into and out of your organization. For me, the word "clean" also implies "malware-free." Although it's true that many data files simply can't contain malware of the traditional forms, it's also true that many can. Even XML files, depending upon how they are used, can contain executable code (an example are the configuration, formatting, and extension files used by Microsoft Windows PowerShell). So one way in which an MFT system can help ensure "clean" data is by integrating with an anti-malware system to scan incoming and outgoing data files, identify any threats, and potentially even quarantine infected files.

Let's assume for a moment, though, that executable code isn't a concern, or that you've already dealt with it. What you're primarily concerned about is clean *data*. Again, there are a couple of meanings for that.

One thing to keep in mind is that data itself, with no executable code involved, can present a security risk. For example, if an attacker knows that you use XYZ Software version 3.45, he might know about specific security issues in that software, such as the potential for buffer overruns and other programming defects. By feeding you data that is deliberately not well formed, the attacker could trigger those security flaws and at the very least take your system offline. Such a thing could also be done unintentionally by a trading partner.

An MFT solution could help protect you from that by not only transferring your data but also *checking* it, perhaps doing range checks on specific fields. Figure 28.1 shows what that might look like.

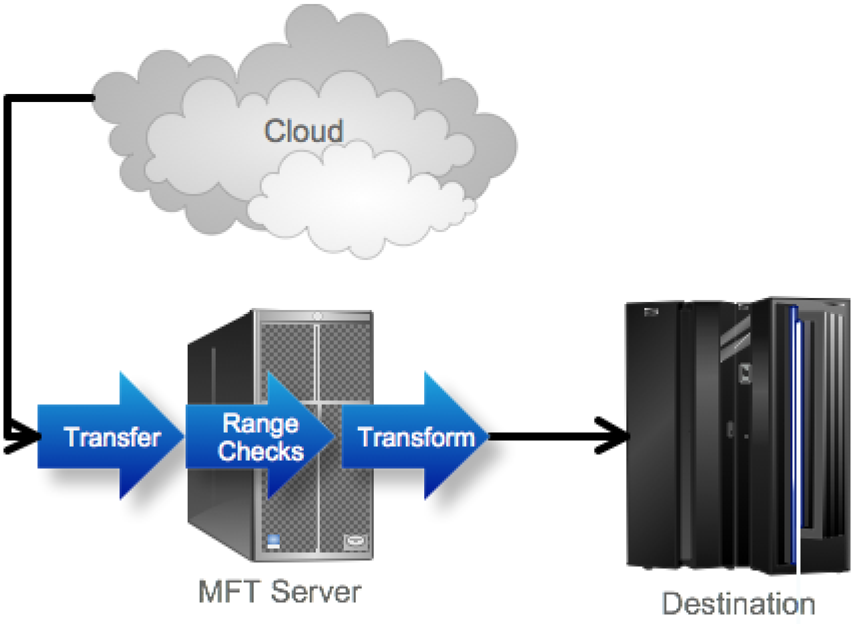

Figure 28.1: Using MFT to check incoming data.

Going a step further, the MFT system could perform more extensive validations on data, ensuring that it was not only well formed but also met your internal systems' criteria. For example, it might translate specific data—changing "US" into "United States"—to make the incoming data conform to what your internal systems require.

In other cases, a complete transformation of the data into different formats might be required. Incoming data received as an XML document, for example, might be transformed into a set of CSV files for loading into your internal systems.

The idea, essentially, is that MFT has become more than just a means of managing the *transfer* of data. In many organizations, it is increasingly becoming a *business communications platform,* capable of translating or transforming data between different formats, different layouts, and different data sets as well as physically moving the data from place to place. Many businesses start using these capabilities for basic transformations, like the XML-to-CSV example I just cited. More sophisticated businesses also use those capabilities to validate data, providing a layer of application-specific protection between the outside world and their internal systems.

I even see companies using MFT systems for purely-internal data transformations. As Figure 28.2 shows, it's becoming more common to see MFT systems sit between different line-of-business systems, moving and translating data between those systems so that the company can enjoy the integration between two un-integrated systems—typically at a much lower cost than implementing some kind of custom integration.

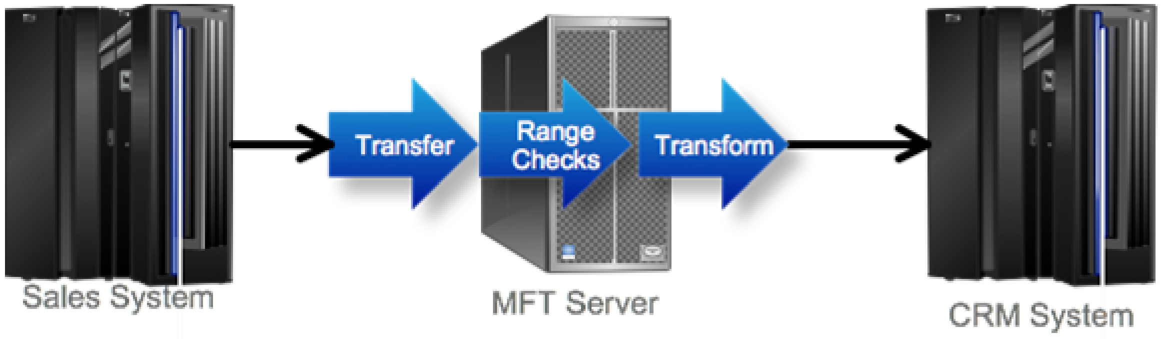

Figure 28.2: Using MFT to integrate internal business systems.

This *type* of technology has long existed in various forms. Beginning with Microsoft SQL Server 7.0, for example, that product's Data Transformation Services (DTS; later renamed SQL Server Integration Services) provided a similar set of capabilities for data transformation. Today, because *moving* the data is often occurring at the same time that it needs to be *transformed*, MFT systems are beginning to naturally fill that role.

Download Additional eBooks from Realtime Nexus!