Realtime publishers

The Shortcut Guide™ To

Secure, Managed File Transfer

sponsored by

IPSWITCH FILE TRANSFER

Don Jones

Copyright Statement

Chapter 3: Mapping Business Requirements to Technical Capabilities—Creating Your File Transfer Shopping List

By this point, you're probably ready to start considering a secure, managed file transfer solution for your company, or even for a specific department, division, or project. Before you start doing Google searches on "managed file transfer," however, you need to have a solid list of your requirements in mind. Although a lot of managed file transfer solutions are remarkably similar in basic capabilities, each of them does offer unique features that, depending on your needs, may be advantageous or disadvantageous to your organization. In this chapter, I'll examine specific business requirements that you may have and translate those to the technical requirements of a file transfer solution.

It's important for me to acknowledge that I can't determine which of these business requirements are important for *your* business; that's up to you. What I can do, however, is cover the ones that are important to a *variety* of businesses, explain why each one might be important, and let you use that information to construct your own shopping list for file transfer capabilities.

> **Hint**
> I'll use the term "shopping list" a lot in this chapter, and I use it very purposefully. Although I don't see a lot of actual lists at the grocery store any more, you should definitely create a written list of business requirements *and* technical features that are important to *your* business. I'll offer examples as I go, and you're welcome to use those as a sort of template for building your own list. Trust me—when it comes time to evaluate solutions, you'll appreciate having a detailed list handy.

Security

Security is the number-one reason to invest in a managed file transfer solution. In fact, if security is of no concern whatsoever, then managed file transfer might not be in your future. However, as I outlined in the previous chapter, *everyone* should be concerned about security to some degree. With that in mind, let's look at some of the very *specific* security requirements that you might want for your managed file transfer solution.

Encryption

Do any of your data transfers need to be kept private? Again, this is something where I've never really met a company who could honestly and completely say, "No." Encryption of some kind or another always winds up on the requirements list for file transfer because every company has data they need to protect. Whether you're being driven by internal security policies or by external requirements (such as the legion of compliance acronyms—GLBA, SOX, HIPPA, PCI DSS, FISMA, and so on), privacy—and thus encryption—is a pretty common requirement.

This is a good time to start documenting the technical capabilities and business requirements you've identified for your company. Figure 3.1 is a sample "shopping list" that I use for almost all software evaluations and acquisitions; you can see that I document not only the technical capability but also the business requirement—and most important, the *reason* for the business requirement (in this example, because I need to comply with the Payment Card Industry Data Security Standard). That way, if I get into a detailed discussion with a vendor, I know exactly what's driving each decision. Vendors can help *me* make better choices when they know *why* I'm asking for a particular feature.

Managed File Transfer Features/Requirements List

	Importance	Supports	Notes
Encryption	5	Must comply with PCI DSS	Transferring cardholder data
Feature			
Feature			
Feature			
Feature			
Feature			
Feature			
Feature			
Feature			
Feature			

Figure 3.1: Starting a file transfer shopping list.

Note that I've also provided an *importance* indicator. I usually use a scale of 1 to 5, with higher numbers representing something of more importance. Again, this is a *huge* time saver for me when I start evaluating actual solutions because I'll be able to score them and rank them against one another based on how well they meet my *most important* business needs.

For now, I'm not necessarily going to worry about what *kind* of encryption. That's a detail I'll dive into in the next chapter, and in my example, the kind of encryption will be driven by my need to comply with PCI DSS. That's one reason to note that if I'm talking to vendors, I don't necessarily need to ask for "1024-bit key encryption;" I can just say, "I need encryption that will be compliant with PCI DSS requirements." Most vendors have already done their homework in that area and know whether they're compliant, and there's no reason not to take advantage of the effort they've put into it.

> **Note**
>
> Annoyingly, most compliance efforts—especially those driven by legislation—contain no technical specifics whatsoever, meaning that even weaker 40-bit encryption would *technically* satisfy the rules. The trick is that the rules only specify that you "keep the data private" or some other non-technical phrase, meaning that if you *do* choose weaker encryption, and it's broken, you've failed to comply. That leaves you a bit on your own—which is why it can be nice to work with a vendor who understands the difficulty and can help guide and educate you. Remember, they've worked with hundreds of customers—they've probably run into your same questions over and over.

Non-Repudiation and Delivery Tracking

As I described in the previous chapter, delivery tracking, guaranteed delivery, and non-repudiation are important requirements for many companies. From a business perspective, these work together to ensure:

- You can prove that a given file was in fact delivered to the intended recipient
- The recipient can verify that the file was not altered in-transit, and that what they have is exactly what you sent

Of course, if you're on the receiving end, the reverse is true: You want to confirm who the file came from, for example. Believe it or not, these capabilities are *often* requirements when you're dealing with industry and legislative compliance, although they may not always seem like an obvious requirement.

Here's why: Compliance rules such as HIPPA and PCI DSS require you to keep track of where your data goes and who accesses it. Once data is out of your hands, of course, you're no longer responsible—provided you played by the rules when transferring it to another party. Imagine an example where two banks are working together, and one needs to transfer credit card member data to the other. Both banks are aware of the PCI DSS rules, and the transfer takes place. A month later, it comes out that some of the cardholder data was leaked and Bank One, who owned the data in the first place, is under investigation. Well, maybe the problem wasn't at Bank One at all, but if they can't *prove* they transferred the data to Bank Two—encrypted and playing by all the other rules—then Bank One might still be on the hook for fines and penalties. Delivery tracking and non-repudiation let Bank One *prove* that another, authorized party was involved, and the investigation can expand to include Bank Two. Compliance is, in many ways, about *evidence*, and these particular capabilities let you provide the additional evidence you need for many situations.

Logging

Aside from encryption, logging is easily the next most-wanted security feature in a managed file transfer system. Businesses want to know *who sent what file to whom, and when*—and logging can provide that information.

This is another *very important* area where you need to be clear on *exactly why* you want logging:

- Because administrators need to be able to troubleshoot file transfers, and the log will contain valuable troubleshooting details
- Because compliance requirements mean you have to keep track of every movement of specific kinds of data
- Because you want to implement service charge-backs against departments that utilize the file transfer infrastructure
- Because you need to support eDiscovery activities, meaning you will need to not only archive the logs but also the data that was transferred

These are all very different requirements. The middle one probably requires a tamper-evident log so that any alterations to the log will be obvious and you'll know that the log has been compromised. The last one might require additional details like file sizes and transfer times so that you can have accurate charge-backs. The *reason* for your logging requirement will be hugely important when you start looking at solutions and talking to vendors.

Authentication and Authorization

Authentication is the process by which users identify themselves to a computer system, such as by typing in a username or password. *Authorization* is the means by which a computer determines what an authenticated user is allowed to do—their permissions, in other words. Any managed file transfer system will support both authentication and authorization, but *how* they do it will vastly impact their usefulness to you.

Some solutions, for example, may only maintain their own internal database of user accounts, passwords, and permissions. That might be exactly what you need: a self-contained solution. Other companies might prefer a solution that lets users authenticate with an existing identity, such as their Active Directory (AD) account. Some organizations may have internal policies that require strong, multi-factor authentication such as biometrics or smart cards. Some solutions permit the use of different authentication mechanisms for different scenarios.

I find that the most flexibility comes from file transfer solutions that can connect to an external directory, such as AD or some other LDAP-based enterprise directory. Why? Those enterprise directories *do* authentication. That's almost *all* they do. And so they do it *well,* meaning they can usually handle biometrics, smart cards, security tokens, or whatever other authentication mechanism you might need, either natively or via extensions. By letting the directory authenticate everyone, your file transfer solution can focus on file transfers.

Of course, if your favorite file transfer solution *only* supports internal usernames and passwords, all is not necessarily lost. Figure 3.2 illustrates an identity architecture that utilizes a *metadirectory.* This is designed to support environments that have multiple solutions—such as a file transfer solution and an Enterprise Resource Planning (ERP) solution—that maintain their own user directories. You manage identities in the metadirectory; it then replicates and synchronizes that information across all your subsidiary directories. Microsoft's Identity Lifecycle Manager (ILM) is one example of a metadirectory.

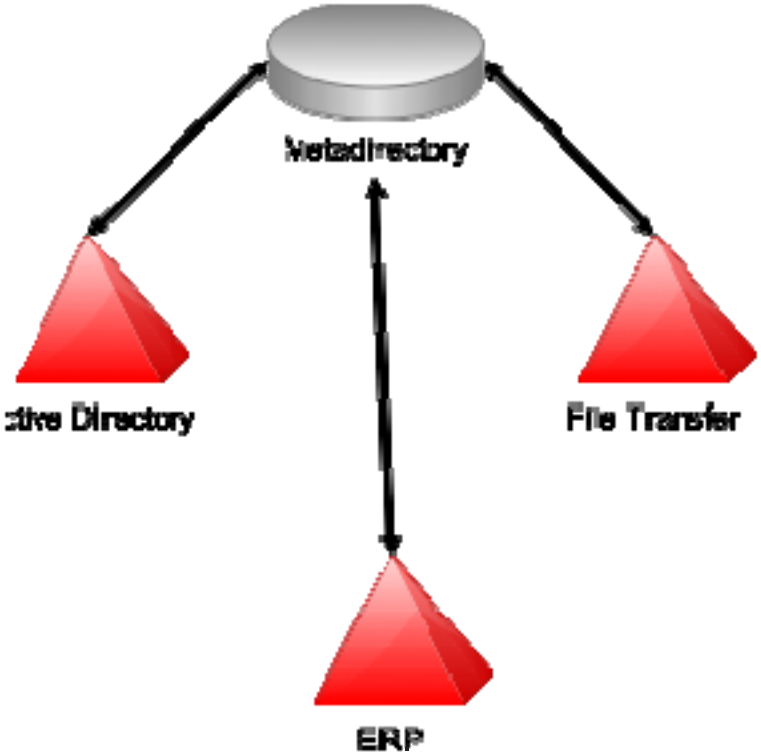

Figure 3.2: Using a metadirectory to synchronize identities.

If your organization already has a metadirectory, your "authentication" requirement might simply be for a file transfer solution that can have its user database managed by the metadirectory, if direct integration with something like AD isn't a possibility.

> **Remember**
>
> The real point is to make sure you're documenting the *why* behind each requirement. Don't just say, "Active Directory integration required;" also note that it's because "single sign-on is an internal policy goal." Vendors may have other ways of achieving single sign-on—the *real* requirement—that you haven't thought of.

Be very careful if you decide that single sign-on is exactly what you want. In some cases, you can manage a file transfer solution more securely when it does *not* integrate with your existing directory. Single sign-on is about convenience, but it can provide broad access to many users when that's not really a business requirement. For example, you might only want to grant users access to the file transfer system for a short period of time. With a non-integrated user database, that's easier: Just create a file transfer user account within the file transfer solution, and perhaps even configure it to auto-expire after a few hours or days. Because users aren't using the corporate-wide directory identity, you can configure different parameters—such as stronger or weaker passwords, or account expiration—than what might be appropriate for a companywide identity.

Protecting Against Attacks

Everybody thinks they won't get attacked…until they are. Having a file transfer solution doesn't necessarily make you more prone or vulnerable to attacks, but it is another thing that *can be* attacked. Some file transfer systems are designed to sit firmly *behind* your firewall, which should do *most* of the protecting that's needed. Firewalls that provide specific support for stopping or reducing Distributed Denial-of-Service (DDoS) attacks, in particular, are valuable because the firewall takes the brunt of any attack and helps shield the file transfer solution. In these instances, the firewall's external IP address is usually the one that business partners and others will use; the firewall then passes that traffic—often after checking it for validity—to the file transfer solution. Figure 3.3 illustrates how this works.

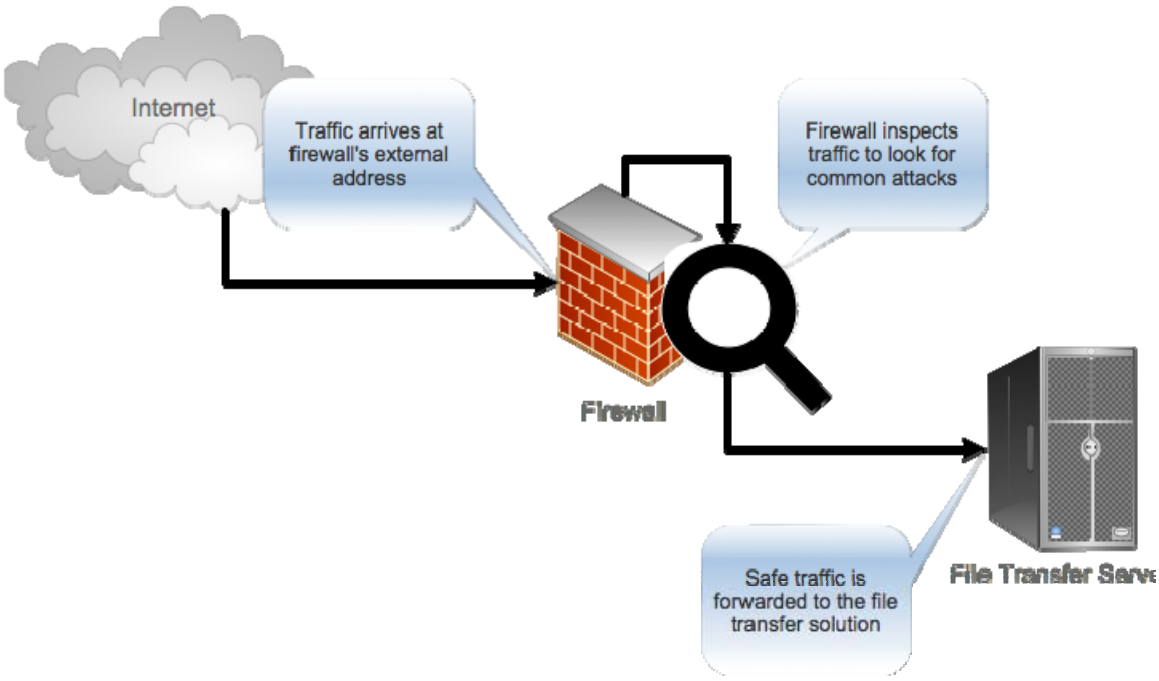

Figure 3.3: Protecting the file transfer solution.

Other file transfer solutions are specifically designed for use in the "demilitarized zone," or DMZ, outside your main firewall. These can sometimes be sold as a customized build of an operating system (OS) like Linux, providing a more locked-down and hardened server; these more-secure file transfer solutions are built with the understanding that they're more exposed than ordinary servers, and often contain their own features for surviving attacks such as a DDoS attack. Still, smart companies will usually install these behind a firewall, creating a second "intranet" as their DMZ rather than installing servers (other than firewalls) that connect directly to the Internet. Figure 3.4 illustrates this approach.

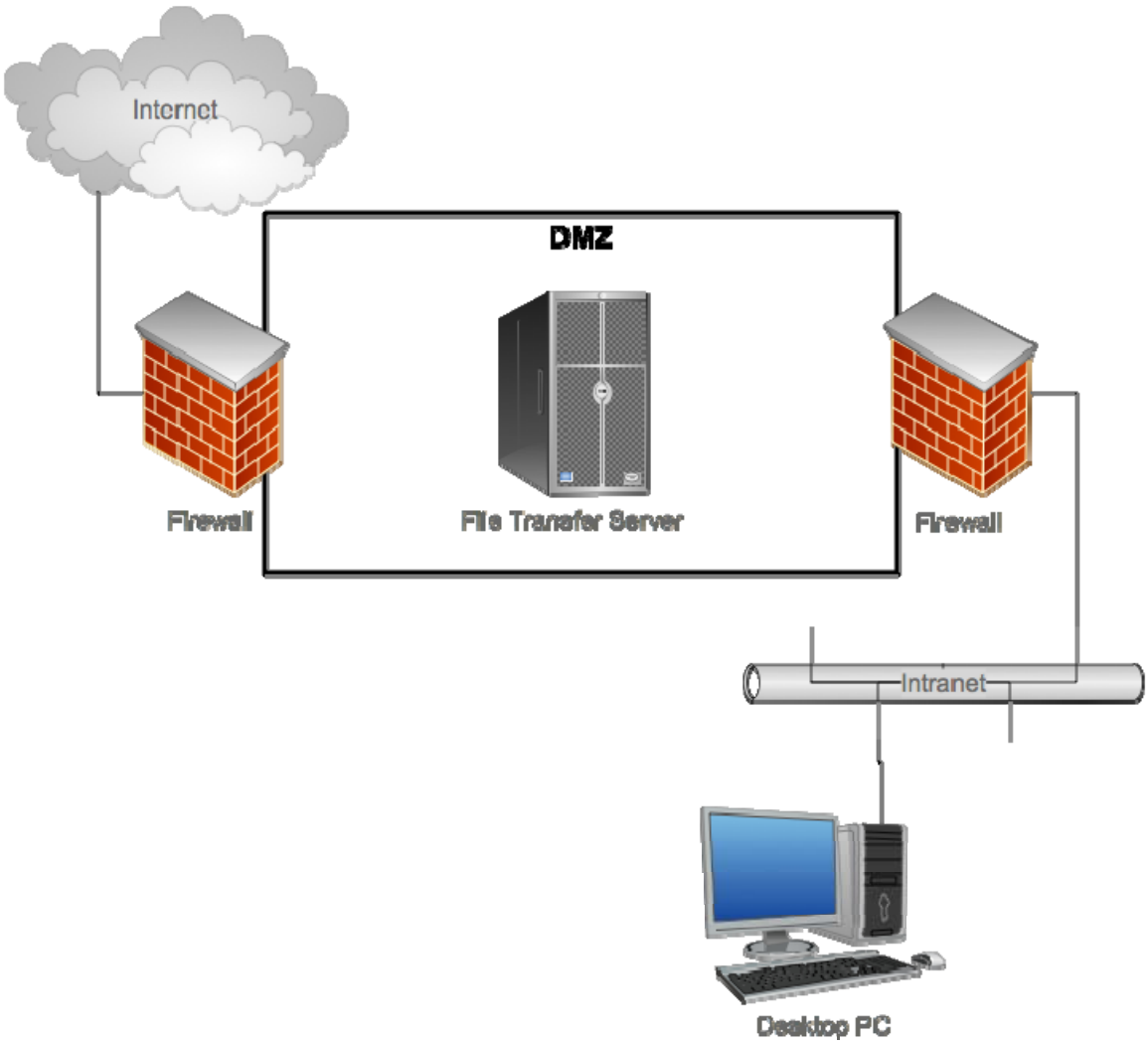

Figure 3.4: Installing a file transfer solution in a firewalled DMZ.

Frankly, "the more firewalls the merrier" is my usual approach.

> **How Do You Stop a DDoS Attack?**
>
> When a solution—whether it's a firewall or a file transfer server—is designed to understand the potential for a DDoS attack, the response is usually an *auto-ban.* Computers are able to look at a source IP address very easily without incurring a lot of overhead. When a server determines that it's receiving a bit too much traffic from an IP address—as can occur during a DDoS or "hammering" attack—the server can simply instruct its network stack to drop all packets originating from the offending address(es).
>
> This isn't always successful; some DDoS attacks "spoof" their source IP addresses to random ones, specifically to overcome the auto-ban technique. But servers can also use a variety of other techniques to help defend themselves.

Other Security Concerns

There are a few other "miscellaneous" concerns that fall under the category of security. See if any of these might be requirements for your organization:

- **Auto-Expiration of Data.** Many organizations don't want data sitting around on a file transfer server for any longer than it needs to, and some solutions offer rule sets that can automatically remove data when it's past a certain age. You might also want the information removed after a specified number of downloads, and depending upon your organization's security needs, might want to ensure that removal is done by means of a secure wipe rather than a simple deletion.
- **Secure Erasing.** In order to transfer a file, a copy of the file needs to be on the file transfer server for at least a short period of time. Solutions that support secure erasing can ensure that *no* traces of files remain on the server once the file is completed and removed.
- **User Account Management.** For file transfer solutions that maintain their own user databases, having the ability to expire user accounts at a certain date, auto-expire accounts a certain number of days after their creation, enforce strong passwords, and so on can make for a more secure environment.

Deployment

How will you get your file transfer solution up and running? This isn't something a lot of companies consider up front, but how—and how quickly—you deploy a solution can have a major impact on your IT staff, your users, and other elements of your company.

Hosted vs. On-Premises

There are two broad deployment scenarios for a managed file transfer solution, although not every vendor offers both. The first is *hosted,* meaning it's managed by a service provider or provided as "software as a service" (SaaS), and the second is *on-premises.* In both scenarios, the file transfer solution usually works exactly the same; the only difference is where it physically lives.

SaaS deployments are often fast; some vendors can have you up and running in a few hours or a few days. They require very little in the way of special skills or attention within your company, and might even be something you do with little or no IT staff involvement, which can be nice for a busy IT staff. You won't have to worry about patching or maintaining the solution, either; the hosting vendor will take care of all that (or should—be sure to check). You may pay more over the long run, but you're up and running quickly and your IT staff won't have much work to do. A SaaS deployment may not offer as much in the way of customization, and it may require some tweaks to your infrastructure so that users can communicate with, and use, the solution. Your data *will* be in someone else's hands, for however briefly, so some compliance-sensitive data may not be suitable for an SaaS deployment.

On-premises deployments mean you're installing the solution right in your own data center. You'll have full control over the process, and your data will be entirely under your control, which might be a must-have feature if you're dealing with strict compliance requirements. However, you're going to be spending some IT staff cycles deploying and maintaining the solution, and you're going to have to invest in some infrastructure—like a server machine—to support the solution.

Neither hosted nor on-premises is better than the other; they're different. Some companies start with a hosted deployment and then migrate the solution in-house; if that's a possibility for you, be sure to include that fact in your shopping list and look for a file transfer vendor that has some experience with that scenario.

Needed Skills or Services

You'll need to work with your eventual file transfer vendor to determine what skills or services you'll need to deploy their solution. Some solutions can be deployed with skills you might already have on your IT team; others might require a consulting services engagement with the vendor. Consulting means less work for your staff, but more money; neither the "do it yourself" nor the "outsource it" approaches are inherently better than the other, but they are different. If your company has a preference, indicate that as a requirement on your shopping list.

Deployment Timeframe

How long will a solution take to deploy? In this instance, while I'm a big fan of working with vendors and relying on their expertise, I'd caution against taking the salesperson's word for it. Most are probably perfectly honest, but some might be tempted to fudge the numbers a bit if they sense a deal in the works. Instead, simply ask the vendor for a couple of references: customers who've implemented the solution and can tell you how long it took them. Of course you can expect to get "best case" numbers, but you can also ask if there were any particular challenges that you need to watch out for.

> **Note**
>
> One advantage of having the vendor's consulting arm deploy your solution is that they can often guarantee a timeframe because the deployment is more under their control. If time is a real factor for your deployment, allowing experts with plenty of past experience is definitely a way to put the project in the express lane.

High Availability

Do you need high availability? That depends: If your file transfer server goes down—and you have to assume it will eventually, if for no other reason than a failed power supply in the server or something—will your business suffer? If it will, then high availability needs to make it onto your shopping list.

Vendors accomplish high availability through varying means. In some cases, as shown in Figure 3.5, an external load balancer may be used to direct traffic to one server or the other; if the "active" server becomes unavailable, traffic is instead routed to a standby. Both servers access a common file system, often on a Storage Area Network (SAN), so they can take over for each other.

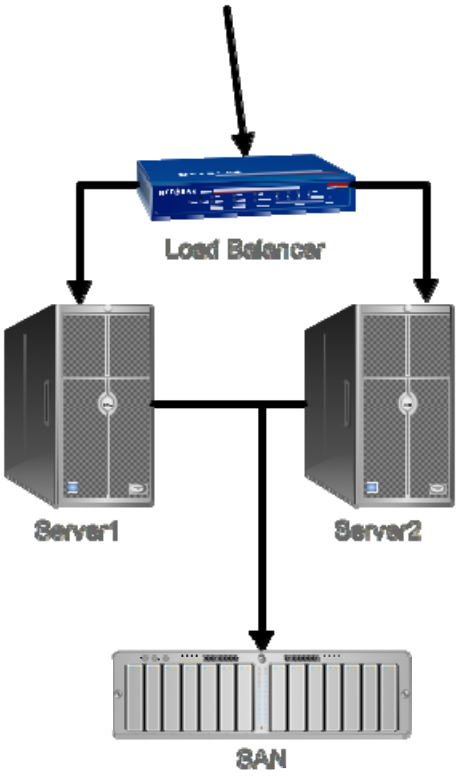

Figure 3.5: High-availability options.

In some cases, both servers can operate simultaneously, balancing workload between them; in others, one is active and the other is a passive standby. Some vendors implement this without the use of external devices or routers; others rely on OS features like Windows' Cluster Service to attain high availability. This is one of those areas that can contain a lot of subtle variations between vendors; don't just take "high availability" as a checkmark on your list. Find out how each vendor *does* high availability, and make sure you're happy with whatever answer they provide.

Integration

How well will a file transfer solution fit into your existing environment? If you're looking at a hosted deployment, this is less important; if you're going to be hosting the solution yourself, then making sure the solution works with what you already have in place is going to be pretty critical.

Supports Your Database?

Most file transfer solutions utilize a database of some kind to log activity, maintain their user database, store workflows and other customizations, and so on. Some solutions use an entirely-internal database, which is fine; just make sure you understand what kind of maintenance that database will require, such as backups, data purging, and so on.

Other solutions use a standardized database platform, such as Microsoft SQL Server or Oracle. There's no one right answer, and no database is better than another for this purpose, but if you're an all-Oracle shop, you might not be excited about having to support SQL Server. You might prefer to run the file transfer database on an existing database server, too, to conserve licensing costs; if that's the case, specify your preferred databases in your shopping list.

Be alert for solutions that use Microsoft SQL Server Express. Not that Express is a bad product, because it isn't, and using it as an "embedded" application database is exactly what it is intended for. It is not, however, natively free of maintenance: It needs to be backed up, it needs to have its performance tuned, and so forth. Many vendors who use Express will write their own code to handle these maintenance tasks, which is a great convenience. I'm not in any way suggesting that Express is a bad choice; if it is what a solution uses, however, just educate yourself about what maintenance—if any—it will require. I've run across a few lower-end application developers who choose Express because it's free and easy, but they don't educate their customers about the maintenance requirements and before long the application isn't performing well.

Also note that Express is the same thing as the "full" edition of SQL Server; if you'd rather not have a solution use Express, but already have a SQL Server that could be used instead, ask the vendor if that's possible. It should be; aside from changing a connection string, the vendor wouldn't have to make any changes to enable compatibility with "full" SQL Server editions.

Works with Virtualization?

Is your company on the virtualization bandwagon? Will you run your file transfer solution in a virtual machine rather than on a physical one? If so, just make sure there aren't any issues in doing so, and that the vendor will support you in that scenario. Most should; some might feel that the virtual environment introduces variables that they don't want their support staff to have to deal with. Again, there's no wrong answer, just what's important to your organization.

Works with Client Computers?

Will the file transfer's client components work with all your client computers? If you're an all-Windows shop, the odds are that the answer is "Yes" because everyone writes Windows-compatible client components. However, if you've got non-Windows stuff, you might want to make sure that there's a plan for it. Cross-platform clients written in Java are one example; Web-based clients are another.

Figure 3.6 shows my shopping list at this point. I want to point out a particular element—about halfway down, I have a requirement for a "Java-Based Client." This is a common type of requirement that often comes out of internal discussions or discovery; I've noticed that my organization has a few Unix and Mac computers, so a Windows-only solution might not be perfect for us. This isn't hugely important—only a 2 out of 5—but I want to make note of it anyway. Also notice that I've made some notes indicating that it's the multi-platform aspect of Java I'm after, not necessarily a love affair with Java itself that's driving the requirement. I've also made a note that other cross-platform alternatives might be acceptable, such as a Web-based client. That's important because as I begin evaluating products, I have something other than "Java client" to look for. The otherwise-perfect solution might not have a Java client, but it might well have a Web-based client that does a just as good—or even better—job. By making sure I note the flexibility in my requirements, I—or whoever is evaluating solutions—will be able to keep it in mind.

Managed File Transfer Features/Requirements List

	Importance	Supports	Notes
Encryption	5	Must comply with PCI DSS	Transferring cardholder data
Logging	5	Must comply with PCI DSS	
Hosted / SaaS	4	No staff time / resources	On-premises may be OK w/svcs
High Availability	2	Might help reduce overhead	Transfers are not time-sensitive
AD integrated	4	Single-Sign On Directive	Could also use an LDAP proxy
Java-Based Client	2	Few Mac/Unix users	Web-based client would be OK too
SFTP	4	Preferred protocols	Other similar protocols might be OK
Customization	5	Workflow is mandatory	We have minimal programming res.
Runs inside VM	4	Move to virtual datacenter	Not a concern if hosted/SaaS
Timeframe	4	Need for "Falcon" Project	Must be deployed within 2 months

Figure 3.6: Finishing up my shopping list.

Workflow and Automation

One big reason—aside from security—to move to managed file transfer is the ability to have review/approval workflows and powerful automation tools. If you'll need these, then consider some of the common details in the next three sections.

Programming vs. GUI-Based Workflow Building

How do you build workflow into a file transfer solution? Some provide graphical drag-and-drop tools that let you build flowcharts, which the solution then follows. Others provide dialog-based "wizards" that are, to my way of thinking, an even easier way to create custom workflows (I hate drawing flowcharts). Still others require you to master a scripting language of some kind, which might be right up your alley.

My general recommendation to my consulting clients is to aim for something graphical—either a workflow builder drag-and-drop system—or a dialog-based "wizard" style interface that "interviews" you and builds a workflow based on your choices and answers. Both of these might well spit out a script in the background, but I don't generally recommend that you purchase a solution with the *intent* of building workflows purely in script. Being able to do so *as an additional option* is nice, especially for extremely complicated workflows, but in general, you'll enable more people to build and maintain workflows if you have less programming or scripting.

Automation

What sort of automation capabilities will you need? This is probably one area where you'll have to do the most investigation into how the solution will be used. Will it need to move files from place to place internally? Will it need to access databases? Figure out exactly what you'd like it to do, then see what solutions are out there that meet, or come close, to your needs.

Support for Delegation

Who will manage the file transfer solution? Here's where it's easy to make a mistake, so let me try and prevent that. A *lot* of companies purchase file transfer solutions to solve a particular divisional, department, or project-based need. That means their delegation requirements are initially simple: Joey over in Sales will run this thing, and that's all we need. Very quickly, though, the file transfer solution starts to be used by other departments or projects, and Joey over in Sales gets too busy. At that point, you need more powerful and flexible delegation options, and you can't add that to a solution that doesn't have them in the first place.

Thus, with regard to "who will manage the file transfer solution," I advise you to imagine the solution being used by your entire company. Make sure it offers delegation that's flexible so that various people can be put in charge of various pieces or elements, and that "who is in charge" can easily be changed over time. You might not need that flexibility up front, but you likely will in the end.

Also—and this is critical if you're dealing with compliance requirements—make sure that the solution supports *separation of duties.* That is, pretty much *nobody* who manages the solution on a daily basis should be able to change auditing or logging settings, clear audit logs, and so on.

Programmability

You may have a need to integrate file transfer with other business applications or processes, and that's where *programmability* comes in.

APIs for Automation and Customization

If you need to—or think you may need to—integrate your file transfer solution with other applications, then make Application Programming Interfaces (APIs) an item on your shopping list. Typically, a file transfer solution API will allow *another* application to start, monitor, manage, or control file transfer operations. For example, you may have an application that generates a large amount of data; that application could use a file transfer API to automate the transfer of that generated data to its destination.

There are some obvious advantages to integrating a file transfer solution in this fashion; however, there are also some downsides. Once you've written code to a specific API, it becomes difficult and time-consuming to rewrite them. Writing to a solution's API pretty much makes that solution a permanent feature of your environment; make sure you're willing to take that dependency before making the decision to use APIs.

Specific Programmability Needs

It's difficult for me to address all the needs that can arise around programmability; you need to look at your environment's own capabilities and needs. Will you need APIs that are accessible via Microsoft's .NET Framework? Windows' Component Object Model (COM)? Java? Something else? Make a list of the programming technologies already in use in your environment, and try to find a solution that natively supports those technologies, or can do so through a vendor add-in or extension.

Protocols

This is probably one of the easiest requirements to map out, but do it in two parts. First, which protocols *do you know for a fact* you will need? Obviously, a solution is only useful if it supports those at a minimum. Then, look at *every other* protocol the solution supports, and try to get the solution that supports everything you *know* you need, and *as many* other protocols as possible. For example, you can use this as a checklist of what's possible:

- AS1, AS2, and AS3
- Local file system
- Network shared folders (SMB)
- FTP
- FTPS

- SFTP / SCP2
- HTTP
- HTTPS
- SMTP

Why worry about things you don't need? Because you might need them in the future, and you'll want to have as much flexibility as possible.

External Connectivity

What types of file transfers will you need? In Chapter 1, I discussed the differences between user-to-user, user-to-system, and system-to-system transfers; each has slightly different techniques and requirements that you might want to consider.

User-to-User

These are the ad-hoc transfers that users often accomplish through email attachments. Typically, file transfer solutions support this by providing an end-user interface—either a standalone client utility or a Web interface or something—that talks to the central file transfer server to accomplish file transfers. Commonly, *both* users—that is, the sender and recipient—will have to have a specialized file transfer client. Different vendors accomplish this in different ways, using cross-platform client software, Web applications, and so on. Some file transfer solutions may also support the use of email, allowing your internal users to specify a file to be transferred, then having the file transfer solution encrypt the file and attach it to an email. There's a lot of variety in how vendors approach this task, and it's an area where you'll want to ask a lot of questions and determine what's right for your company's needs.

Enforcing User-to-User Transfers

If user-to-user file transfers are a business need—and I haven't run into many businesses where they *aren't*—make sure that, in addition to acquiring a solution that supports them, you have the capability to enforce the use of that solution. In other words, you need to be able to turn off *other* ways of completing ad-hoc user-to-user transfers. This might include restricting the use of instant messaging clients that allow file transfers, restricting the size of message attachments in your email system, and so on. You might also use your firewall to block ports commonly used for protocols like FTP, although you'll obviously need to provide an exception for the file transfer system itself. You might also block outgoing traffic in ports used by torrent and other file-sharing clients—just to ensure that files aren't being transferred outside the managed system.

It's very, very difficult to block *all* forms of file transfer. Microsoft's Background Intelligent File Transfer (BITS) service, for example, can upload as well as download. BITS is used by Windows Update, so you don't usually want to disable the service outright; it uses HTTP, so it's difficult to separate BITS traffic from ordinary Web browsing. All things to keep in mind.

User-to-System

This isn't much different than a user-to-user transfer, although the "person" on the other end is usually a Web site or FTP server. That means less specialized software on the receiving end of the transfer. Again, vendors vary greatly in how they implement this, so ask questions of the vendor if you'll need to provide user-to-system transfers for your users.

System-to-System

These transfers are usually automated, between two servers—often via some form of FTP or via HTTP or HTTPS. This is the type of transfer that gets most companies looking at managed file transfer in the first place.

Coming Up Next…

By now, you're ready to construct your file transfer shopping list, if you haven't done so already. With that list in hand, you can start evaluating file transfer solutions. But evaluations like this aren't as simple as checking off features from your list because, in most cases, there are important subtleties that you need to be aware of. In the next chapter, I'm going to focus on the task of actually evaluating a file transfer solution against your capabilities shopping list. I'll show you a technique for scoring different solutions on your list, and cover some of the often-overlooked details that can save you a lot of time and money—after all, *nobody* likes to acquire a solution only to find out weeks or months later that the features you bought aren't *quite* what you were hoping for.

Download Additional Books 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 book to be informative, we encourage you to download more of our industry-leading technology books and video guides at Realtime Nexus. Please visit http://nexus.realtimepublishers.com.