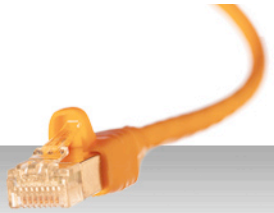
How to Boost File Transfer Speeds 100x Without Increasing Your Bandwidth

Date



Contents

Introduction 2

Network conditions and file transfer performance 2

Latency 3

Packet Loss 4

Bandwidth 4

Congestion 5

The problem with TCP 6

1. Reliable transmission 6

2. Flow Control 7

Where the problems lie 8

The problem with UDP 11

Achieving fast and reliable file transfers with AFTP 12

Choosing a file transfer solution 15

Does it support multiple protocols? 15

Is it platform independent? 15

Does it automate business processes? 16

Does it simplify auditing tasks? 16

Does it provide ample security? 16

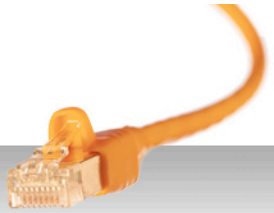
Is it able to meet compliance mandates? 17

About JSCAPE MFT Server 17

Example Uses 17

Feature Summary 18

Download 18



Introduction

It's hard to imagine that in an age where mere household broadband connections are already in the Megabits (or even hundred Megabits) per second, some large organizations still have to transfer files at unbelievably slow speeds.

Slow file transfers may not be a big issue if you're just sharing small, personal files. But if you're working in an organization that has to process gigabytes of mission-critical data under a tight schedule, a transfer speed of just a few hundred kbps may not be acceptable, unless of course, nothing can be done about it.

If you want to understand where speed limitations are coming from and how to overcome them, this paper is for you.

Well, for a long time, that has been the case for file transfers between offices of multinational corporations, universities, governments, research facilities, and other organizations separated by large distances. The reason is that networks spanning long distances are subject to high latency and other factors that result in poor network conditions.

These conditions can lead to failed file transfers, significant delays in processing of digital data, underutilization of network resources, and even lost business opportunities.

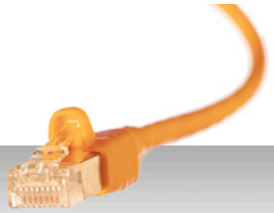
Because file transfers typically rely on TCP-based technologies, which unfortunately are very sensitive to such network conditions, many of these organizations are left with very limited choices. They put up with the slow file transfers, increase their bandwidth subscription, or revert to mailing files inside DVDs.

If you want to understand where these speed limitations are coming from and what you can now do to overcome them, then this paper is for you.

This time, you have a better choice.

Network conditions and file transfer performance

There are a number of factors that can affect file transfer performance. These include: latency, packet loss, bandwidth, and congestion. Before we take a closer look at these factors, let's briefly define the basic concept of a “packet” as we'll be using this term quite often soon.



When you send information during a file transfer session, it is first chopped into smaller pieces and then encapsulated into what are known as packets.

Aside from the pieces of information you send, which is known as the payload, each packet comes with a bunch of additional information comprising what is known as the TCP Header.

It is these packets, made up of the payload and TCP Header that are sent over the network to the intended receiver or destination. It is also these packets that are directly affected by the network conditions we are about to discuss.

Latency

When packets are transmitted over a local area network (LAN), they appear to arrive instantaneously. In reality, however, they do not.

> The greater the value of latency, the longer the delay. So, if you have a high-latency network, you can expect significant delays.

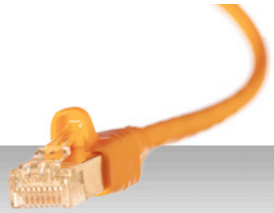
Because packets travel at speeds that are dependent on the properties of the physical medium (e.g. fiber optic or, as in the case of wireless networks, air) through which they pass and because they have to cover a certain distance, there is actually a very small amount of delay.

This delay, which is measured as the time it takes for a packet to get from one point in the network to another (although it can be the same point if you're measuring a round-trip), is more formally known as latency and is usually expressed in milliseconds (ms).

The longer the distance the packet has to travel from those two points, the greater latency will be. That's why latency is more evident in wide area networks that span very large distances. For example, latency in a LAN may just be 5 ms, but latency between a computer in LA and a server in Tokyo can be 200 ms.

In addition to the latency caused by distance and the properties of the physical medium, more latency can be introduced when the packet passes through gateways or proxy servers because of the various processes it has to undergo there.

Again, the greater the value of latency, the longer the delay. **So, if you have a high-latency network, you can expect significant delays.**



Packet Loss

Not all packets you send arrive at their destination. Some of them are dropped/lost along the way. This can happen due to a variety of reasons like signal degradation, faulty networking devices or drivers, and congestion. Naturally, if a packet is lost, the information that will be received on the other end would not be complete.

> Even a single lost packet can be a big problem because some text and other portions may end up missing.

In some cases, that's not a big issue. For instance, a few lost packets won't have a critical impact when you're having a VoIP session, playing a real-time online multiplayer game, or viewing streaming media. Audio and video quality may suffer a little but that's all.

However, if you're sending a document, **even just a single lost packet can be a big problem** because some text and other portions may end up missing.

Packet loss is expressed in percent (%), wherein a 0% packet loss means all packets reached their destination, a 100% packet loss means all packets failed to reach their destination, a 50% packet loss means only half of them arrived, and so on.

Clearly, high packet loss is not a good thing either.

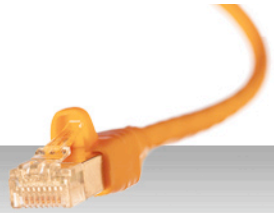
Bandwidth

If you have ever taken part in a buying decision involving network devices, Internet connections, and other stuff related to networking, chances are bandwidth was one of your major considerations. That's because sales people often put emphasis on bandwidth when they sell you networking products.

Bandwidth is usually understood to mean the **theoretical maximum rate of data transfer** that a particular network or network device is capable of handling. It is normally expressed in multiples of bits per second (bps) like: Kbps (Kilobits per second), Mbps (Megabits per second), and Gbps (Gigabits per second).

All things being equal, when you compare two networks or devices with different bandwidths, the one that has greater bandwidth is normally understood to be faster.

The table below features bandwidth specifications of some common Internet access technologies.



Technologies	Bandwidths
T1 / DS1	1.5 Mbps
Wireless 802.11g	11 Mbps
T3 / DS3	45 Mbps
10 Gigabit Ethernet	10 Gbps

Again, those are just theoretical values. In reality, due to factors like latency and packet loss, it's almost impossible to achieve them. In fact, in wide area networks, the actual rate of data transfer, which is known as throughput, is usually just a small fraction of advertised bandwidth.

For example, while the bandwidth between a host in LA and a host in Tokyo is 45 Mbps, the actual throughput might be only 5 Mbps. That's just a little over 10% of what you'd have expected in a world without latency and packet loss. Alas, that world is not the world we live in.

Congestion

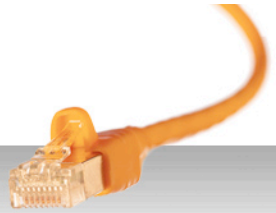
What happens when the total volume of data simultaneously directed to a T1 connection exceeds its bandwidth of 1.5 Megabits per second? The same thing that happens on most main roads during peak hours. You get traffic congestion, and in the case of networks, network congestion.

Just like roads, a congested network experiences reduced throughput.

Just like roads, a congested network experiences reduced throughput. All types of networks have bandwidth limitations. The moment the volume of incoming data exceeds a network's maximum capacity, congestion occurs.

But bandwidth is not the only network resource that has limitations. Network devices such as routers and switches have portions in their memory known as buffers in which received data is temporarily stored before they are processed. Just like any kind of memory, these buffers cannot accommodate an infinite amount of data. If the buffers start filling up because incoming data is more than the buffer can handle, this too can cause congestion.

Once network congestion gets worse, it can lead to what is known as a congestive collapse. At this stage, throughput is reduced to levels where useful communication is no longer possible. If you go back to our road traffic analogy, this condition is what we normally call a traffic jam.



Network congestion is a very undesirable condition because it can also increase latency and packet loss, which can aggravate the situation. This is how it happens.

When the rate of output on a device gets significantly lower than the rate of input, as what happens during congestion, packets in queue start backing up and processing gets delayed. Thus, latency eventually increases.

At the same time, when the receiving device's buffer starts filling up, the device starts dropping packets. But that's not the end of the story. The sending device, upon realizing that packets have been lost, starts resending packets. These new packets add to the congestion and the situation worsens.

The problem with TCP

Most file transfers are carried out over a technology known as the Transmission Control Protocol or TCP. In existence since 1974, TCP is interwoven into the core fabric of the Internet. Every time you use email, view web sites, send files, or administer IT systems remotely, you're most likely doing it over TCP.

> Being an old technology, it is understandable for TCP to have limitations when thrust into today's communications environment.

Being an old technology, it is understandable for TCP to have limitations when thrust into today's communications environment.

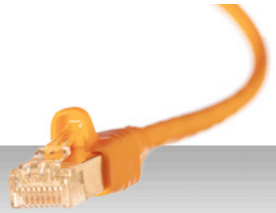
In the case of file transfers, majority of the protocols in use today, including FTP, FTPS, SFTP, SCP, HTTP, and HTTPS, still rely on TCP. Therefore, if you often do file transfers, then you're sure to be affected by the problems inherent in this protocol.

To gain a better understanding of the problems associated with TCP, we must first talk about two of its strong qualities. Ironically, the mechanisms that ensure those qualities are also where the problems are coming from.

1. Reliable transmission

TCP is designed to provide reliable delivery of information. That means, if you send a document using TCP, your recipient can expect to get it in exactly the same form as the original.

This is accomplished by a set of exchanges between the sender and the receiver. When a packet from a certain sender arrives at the receiving



end, the receiver responds with an acknowledgement to inform the sender that the data arrived successfully.

If no acknowledgement is received by the sender after a certain period of time, it retransmits the unacknowledged data.

In addition, each byte of data on a packet is marked with a sequence number. The number sequence, which is monitored by both the sender and receiver, is used to determine:

- The correct ordering of the data;
- Which packet got lost and require retransmission; and
- Whether there is already a duplication of received data.

2. Flow Control

TCP is also equipped with a flow control mechanism aimed at preventing the sending device from transmitting an amount of data greater than what the receiving device or the network can handle. This is known as the **TCP sliding window** flow control algorithm. How does it work?

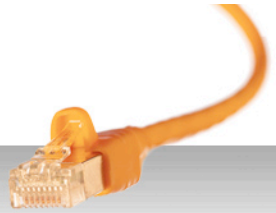
When transmission begins, the sender starts sending just a few packets. Although the receiver has to acknowledge each packet it gets from the sender, the sender does not always have to wait for the acknowledgement of the last packet it sent before sending the next packet.

Rather, the sender only has to wait for an acknowledgement once a certain number of packets are reached. This limit is known as the **TCP window**. It is the maximum number of packets that a sender can send out before it has to stop and wait for an acknowledgement.

Let's say the TCP window is 4. In this case, the sending device can send out a maximum of 4 packets before stopping to wait for an acknowledgement.

This is important because if no acknowledgement is received after a certain period of time, then that means a packet might have been lost and that something might have gone wrong at the receiving end.

On the other hand, if the acknowledgement it has been waiting for does arrive, the sender then increases the TCP window. In other words, the TCP window increases (or "**slides**") every time it receives an acknowledgement.



This also means that the number of packets that the receiver is allowed to send before it has to stop to wait for an acknowledgement slowly increases. When you have a larger window, the number of packets on their way to the receiver at a given time will be greater.

The increase in window size gets faster as more acknowledgements are received. Because the rate of increase in window size is slow at the start, it is known as a **slow start state**.

The increase of the TCP window slows down again once a certain **threshold limit** is reached. This threshold is initially set by the receiver at the start of the connection and is the maximum amount of data that can be accommodated by the receiver's buffer.

Once the threshold is reached, the TCP session enters what is known as the **congestion avoidance state**. At this point, the system would have reached **what it perceives as the highest throughput possible**.

So where do the problems lie?

Where the problems lie

If you recall, for TCP to ensure reliable transmission, each sent packet has to have an acknowledgement. The role of these acknowledgements is very critical.

“

For TCP to ensure reliable transmission, each sent packet has to have an acknowledgement.

During the **slow start state**, the TCP window will only grow upon the receipt of an acknowledgement. The longer it takes for an acknowledgement to arrive, the longer it will take for a TCP window to grow.

Remember, if you have a small TCP window, the sender can only send a few packets at a given time. So really, it would be in your best interests if the TCP window would grow quickly.

There wouldn't be an issue if both the sender and receiver are located in the same building or compound. The acknowledgement would arrive instantly and the sender's “waiting time” would be negligible.

But if you have a **high latency** network wherein the distance between sender and receiver is significantly large, e.g. if the sender is in LA and the receiver is in Tokyo, then the sending machine would have to wait much longer.

Things can get worse when packets or acknowledgements get lost. As soon as the sender detects **packet loss**, like when an acknowledgement fails to arrive, it reduces the TCP window considerably and the session goes back to **slow start** state.

Not only that. The system would also change its perception as with regards to the highest throughput possible. As a result, it will also significantly reduce the threshold limit we talked about earlier. This now has serious consequences.

Even if the TCP window goes back up after slow start, its rise will be limited by this greatly reduced threshold limit. Thus, even a small degree of packet loss can be catastrophic to your file transfer.

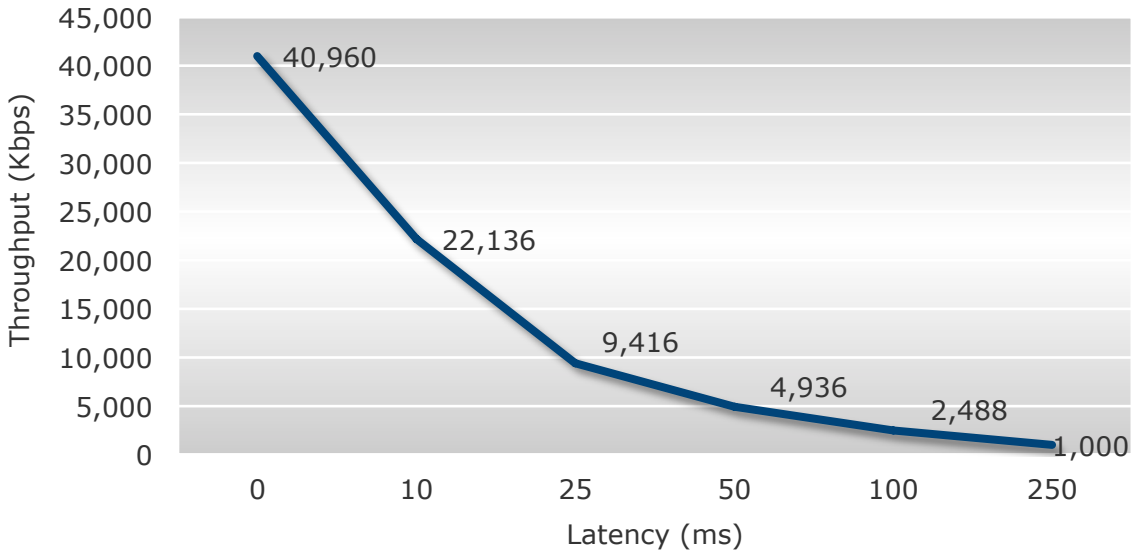
Networks spanning long distances and involving multiple network devices are always subject to high latency and high packet loss.

The result? **A tremendous drop in throughput**.

We conducted a series of FTP-based file transfer experiments in a lab setting to find out just how significant the effects of latency and packet loss are.

Here's how throughput fared at zero packet loss but increasing latency:

Figure 1: Effects of Latency – FTP

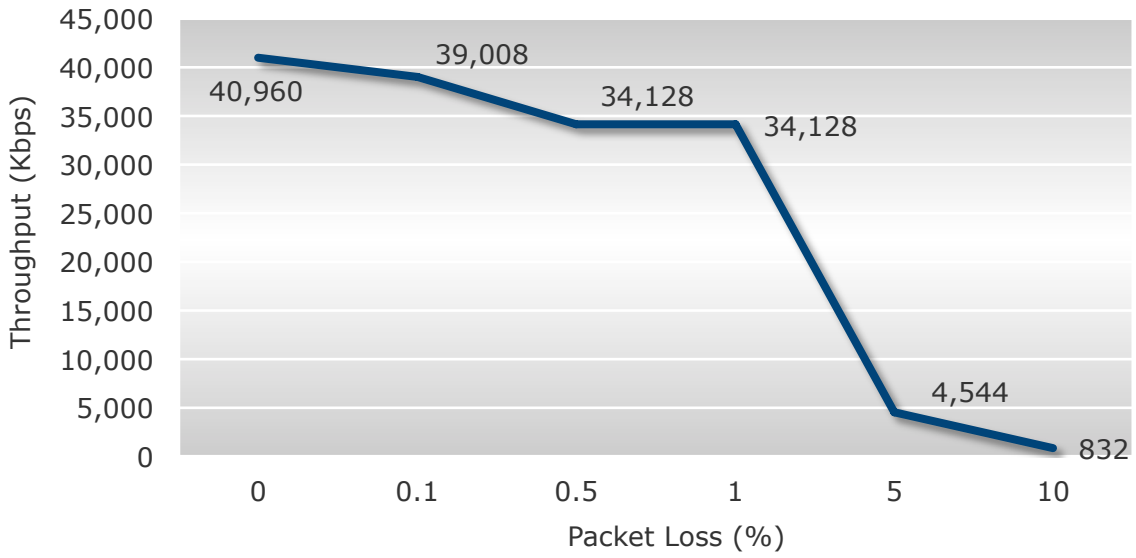


The throughput at the bottom-right of that graph is what you'd expect in a FTP-based file transfer from LA to Tokyo over a 45 Mbps network even

with zero packet loss. As you can see, while a latency of 200ms may seem negligible, the resulting throughput will tell you it isn't.

Now, here's how throughput fared when we fixed latency at 0 ms and steadily increased packet loss:

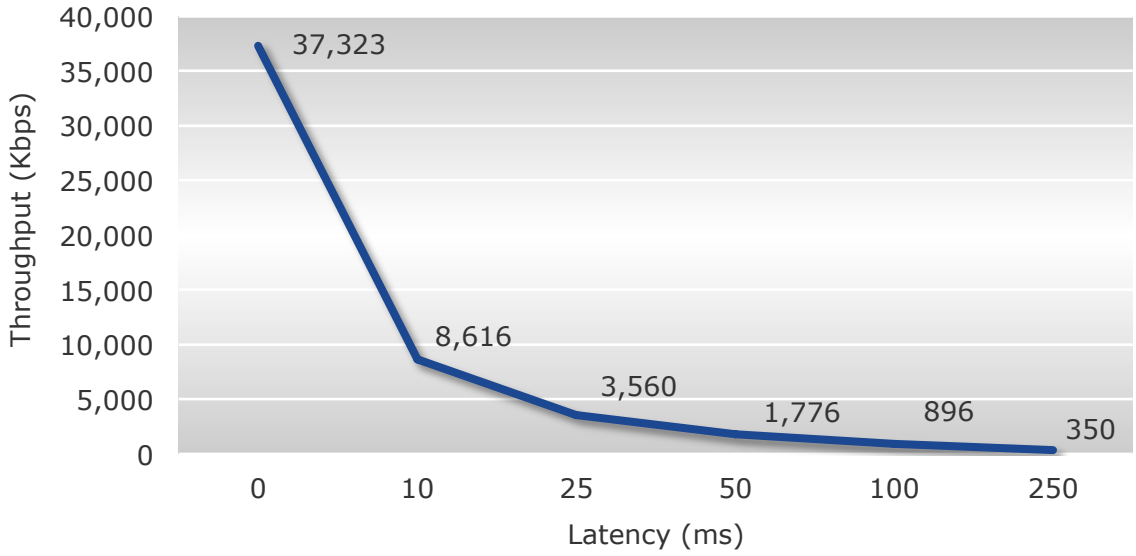
Figure 2: Effects of Packet Loss – FTP



Notice how steeply throughput drops beyond 1% packet loss. That's the packet loss you'd normally experience when transmitting from LA to Tokyo.

When we combined the two factors, specifically, when we increased latency while keeping packet loss at 0.5%, this is what we got:

Figure 3: Effects of Latency & Packet Loss (0.5%) – FTP



When you know that your throughput's only going to be like this, i.e., **no more than 1% of bandwidth**, it's not going to be easy to approve your staff's request for larger bandwidth.

Not all transmissions are carried out using TCP. The relatively steady flow of data in real-time online multiplayer games (or at least the real-time aspects of these games), VoIP, and video streaming certainly could not be achieved if you rely purely on TCP.

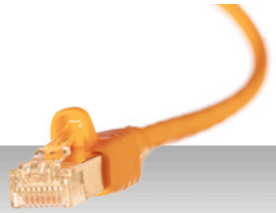
If TCP were used on an online multiplayer first-person-shooter game, for example, there'd be a lot of instances when players or even an entire scene would appear to be standing still.

Obviously, a much faster protocol is being brought into play here. That protocol is the User Datagram Protocol or UDP.

The problem with UDP

Like TCP, UDP is one of the core protocols of the Internet. But unlike TCP, UDP is not designed to provide reliability, ordering and data integrity.

Thus, if your sending device transmits packets using simply UDP, the packets that arrive at the receiving end may not be in order and may suffer duplication. Furthermore, if a packet gets lost, your sender and receiver may not know about it.



But if you recall our discussion on packet loss, that's not always a problem. A couple of lost packets certainly won't have a critical effect on, say, a VoIP session.

There may be a slight degradation in voice quality, but you'll still be able to understand what the other person is saying. In fact, the number of lost packets may be so small that you may not be able to notice anything wrong with the voice quality at all.

Because UDP is not burdened with acknowledgements and other processes for error checking and correction, transmissions are much faster.

Still, UDP by itself is not suitable for all types of transmissions.

Surely, if you're sending a document, you would need the information to be all in, in order, and devoid of duplicates. So now you understand why popular file transfer protocols like FTP, FTPS, SFTP, HTTP, and HTTPS are forced to employ TCP.

So how is it possible to achieve fast, reliable file transfers?

Achieving fast and reliable file transfers with AFTP

Fast and reliable file transfers can actually be achieved if you combine the strengths of TCP and UDP. JSCAPE has developed a protocol known as Accelerated File Transfer Protocol or AFTP and is included in its JSCAPE MFT Server and AnyClient software.

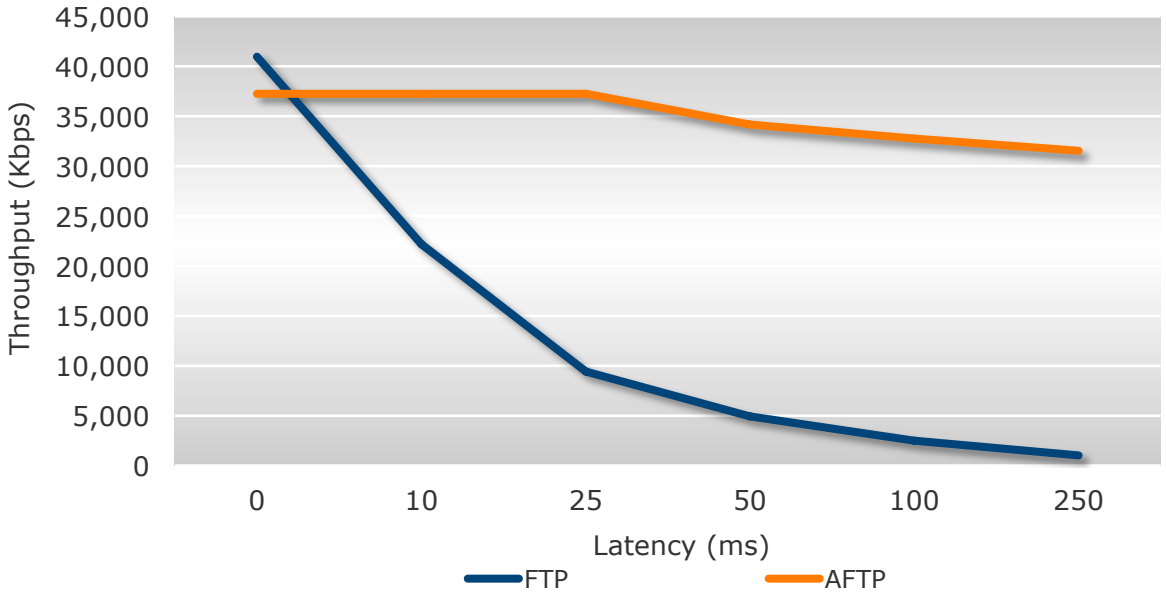
Fast and reliable file transfers can actually be achieved if you combine the strengths of TCP and UDP.

In a typical AFTP file transfer, the bulk of the data transfers are done on a UDP channel while other tasks such as user authentication, file management, and the coordination of the file transfers are done on a TCP channel.

Because UDP does not have an acknowledgement process like TCP, it is not as affected by latency and packet loss. Hence, a file transfer using AFTP performs much better than one relying purely on FTP.

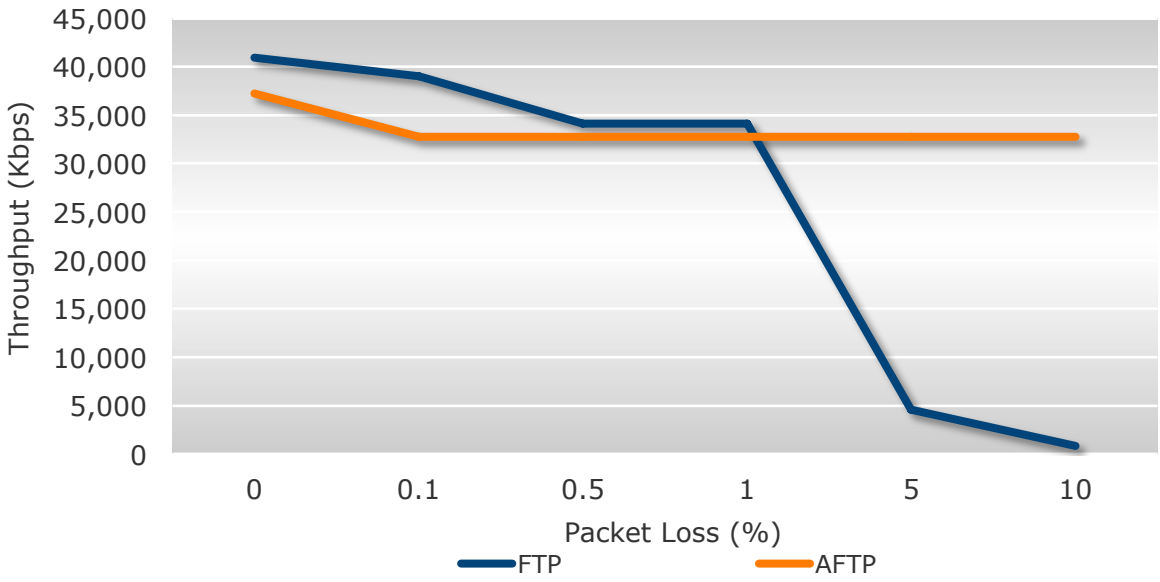
Here are a couple of graphs showing how noticeable differences between the throughputs of **AFTP** and **FTP** file transfers can be.

Figure 4: Effects of Latency – FTP vs. AFTP



	0	10	25	50	100	250
FTP	40,960	22,136	9,416	4,936	2,488	1,000
AFTP	37,232	37,232	37,232	34,128	32,768	31,504

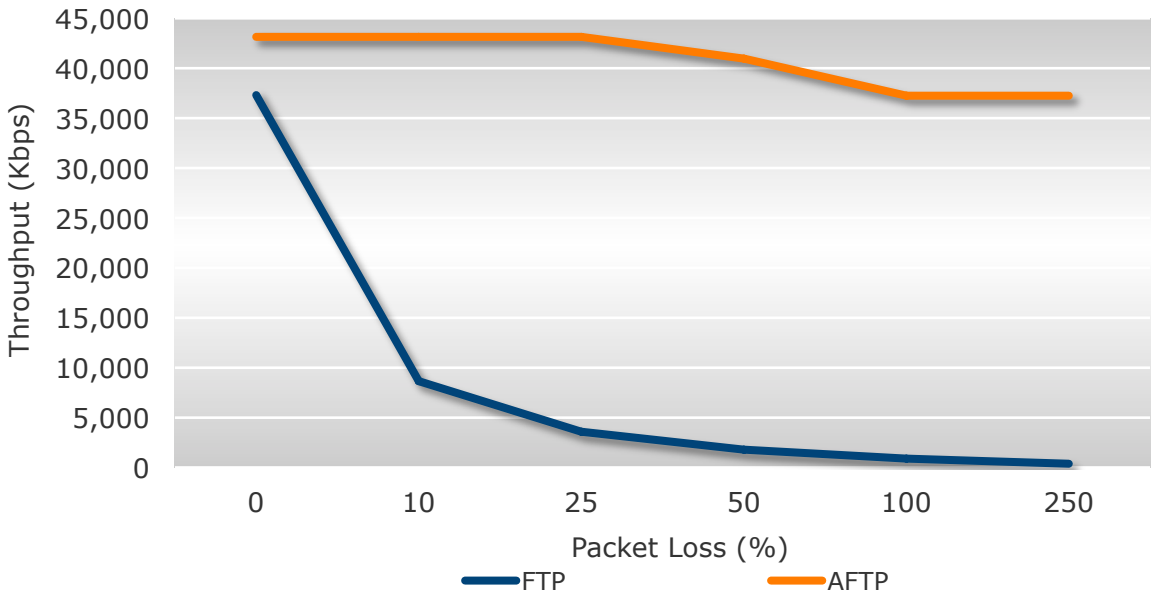
Figure 5: Effects of Packet Loss – FTP vs. AFTP



	0	0.1	0.5	1.0	5	10
FTP	40,960	39,008	34,128	34,128	4,544	832
AFTP	37,232	32,768	32,768	32,768	32,768	32,768

Notice how AFTP throughput remains almost unchanged. Here's one more graph showing what happens if you introduce both latency and packet loss:

Figure 6: Effects of Latency & Packet Loss (0.5%) – FTP vs. AFTP

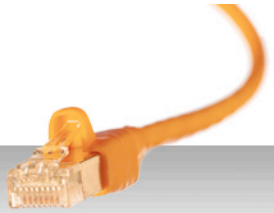


	0	10	25	50	100	250
FTP	37,323	8,616	3,560	1,776	896	350
AFTP	43,112	43,112	43,112	40,960	37,232	37,232

Clearly, AFTP outperforms pure FTP by a wide margin and the advantage gets even more pronounced as network performance deteriorates, i.e., when there is greater latency and packet loss.

What you're actually looking at in the lower-right corner of that last graph is an **FTP throughput of 350 Kbps** and an **AFTP throughput of 37,232 Kbps**. That's actually a boost in throughput of **more than 100x**.

So what particular product can boost your file transfers at least a 100x?



Choosing a file transfer solution

You might be surprised, but there are already a few file transfer programs that support some form of accelerated file transfer. And in most cases, these products don't differ much in terms of improved throughput.

But then that doesn't mean you can just go on and pick a product randomly. Throughput shouldn't be the only factor you need to consider when choosing a file transfer solution.

Some of the critical questions you should also be asking are:

Does it support multiple protocols?

Some companies don't just transfer files within their organization. If you also share information with other people, like your customers, suppliers, business partners, and so on, there will always be that possibility that their file transfer application won't be compatible with yours.

“

If you also share information with other people, there will always be that possibility that their file transfer application won't be compatible with yours.

That is, it's always possible that their application might be using a different protocol. If that is the case, sharing files with them can be a problem.

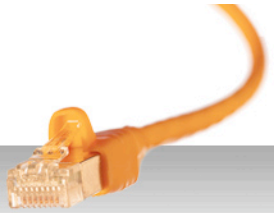
The most widely used protocols include FTP, FTPS, SFTP, HTTP, and HTTPS. So if your AFTP-enabled file transfer solution also supports other protocols, including those mentioned, incompatibility may no longer be an issue.

Is it platform independent?

It's not unusual for a large organization to maintain computers having different operating systems. Some of your employees may be using Windows laptops and desktops. Others may be using Macs. Some of your servers may be running on Linux, while others on Solaris, or AIX.

If your computers run on different platforms, it would be easier for both your IT staff and your end users if your file transfer solution can run on all the major platforms.

By having a platform-independent solution, you can simplify a lot of relevant activities like training, installation, maintenance, upgrades, troubleshooting, and so on, thus saving you time and money.



Does it automate business processes?

> Having the ability to automate some business processes can help you get more things accomplished much faster.

With so many tasks vying for attention everyday, having the ability to automate some business processes can help you get more things accomplished much faster.

It would therefore be a big boost to your team's productivity if your file transfer solution were already equipped with features that automate certain processes. Some tasks that should be automated whenever possible include the following:

- File transfers to trading partners especially during non-office hours when bandwidth utilization is low;
- Virus scans on every file upload;
- PGP encryptions on every file upload;
- Compression/decompression of transferred files;
- Email notifications regarding certain events such as those mentioned in this list;

Does it simplify auditing tasks?

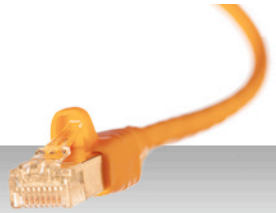
If you expect a lot of people to use your file transfer system, you will want to have a way to keep track of each individual's file transfer sessions. This will allow you to trace back their actions if ever any issue arises in the future.

For example, if a data breach hits you, you can conduct a more thorough investigation if you can identify who accessed what during a particular period of time. A file transfer solution with logging capabilities can be a big help in these situations.

Does it provide ample security?

Do the files you transfer include sensitive data like personal information, trade secrets, or financial data? If so, you wouldn't want unauthorized personnel to get a hold of them.

These days, threats against confidential data can come from both outside and within your organization. It is therefore important for your file transfer solution to be equipped with basic and advanced security features like DLP (data loss prevention), access controls, encryption, two-factor authentication, and others.



This will enable you with enough flexibility to implement appropriate security measures when addressing identified risks.

Is it able to meet compliance mandates?

In line with efforts to curb fraud, identity theft, corporate identity theft, and other attacks on electronic data, many developed countries now have laws and regulations that affect file transfers.

> Failure to comply with laws and regulations may result in large penalties and damage to reputation in the event of a data breach.

Failure to comply may result in large penalties and damage to reputation in the event of a data breach.

The Sarbanes-Oxley (SOX) Act, Gramm-Leach-Bliley (GLB) Act, Health Insurance Portability and Accountability Act (HIPAA), the Payment Card Industry Data Security Standard (PCI DSS), and the European Union Data Protection Directive, are just some of the laws and regulations you may need to comply with.

Thus, if the scope of your business operations include the US or any EU member state, having a file transfer system already designed to meet regulatory requirements would be a big boost to your overall compliance initiatives.

About JSCAPE MFT Server

JSCAPE MFT Server is a platform independent managed file transfer solution that centralizes all of your file transfer processes into a single easy to use application. JSCAPE MFT Server supports all major file transfer protocols including AFTP, FTP/S, SFTP, SCP and HTTP/S.

Example Uses

- Provide a highly secure method for exchanging data both internally and between trading partners that meets PCI DSS, SOX, HIPAA and GLBA compliance requirements
- Simplify file transfers by unifying all services and processes into a single platform independent solution
- Gain control of users and system resources
- Automate file transfers between trading partners and respond to server side events
- Provide a secure web-based file transfer gateway to clients without installing any software

Feature Summary

- Platform independent solution with installers for Windows, Linux, Solaris, UNIX, AIX and Mac OS X operating systems
- Support for all popular file transfer protocols including FTP/S, SFTP, SCP and HTTP/S
- Accelerated file transfer over high speed / high latency networks moves files up to 100 times faster than FTP
- Brandable and easy to use web based client interface for transferring and managing files
- Support for multiple domains each with their own unique file transfer services and configurations
- Multi-lingual web user interface with built in language packs for English, Spanish, French, German and Russian
- Authentication modules for integrating with existing LDAP, Active Directory, NTLM, PAM and database user repositories
- Triggers module for use in automating business processes such as file transfers with trading partners and responding to server events
- Data Loss Prevention module for use in preventing leakage of sensitive data such as credit card, SSN and bank account numbers.
- Jailed user accounts and virtual file system keep your users data separate
- Ad-hoc file transfer module for granting non-account holders temporary access to data
- Several security features designed to meet PCI DSS, HIPAA, SOX and GLBA compliance requirements
- Support for high availability and load balancing environments
- Integrated key management tools for OpenPGP, SSH and SSL standards
- Directory monitors with ability to detect when directories have been modified
- Java based API for managing the server, users and configuration settings
- Integrity checksum and resume file transfer capabilities
- ... and much more

Download

Start accelerating your file transfers today by downloading a free fully functional evaluation of JSCAPE MFT Server.

http://www.jscape.com/products/file-transfer-servers/jscape-mft-server/