Hitchhiker's Guide to Network and WAN Optimization Technologies

NETEQUALIZER

Manufacturers make all sorts of claims about speeding up your network with special technologies. In the following pages, we'll take a look at the different types of technologies, explaining them in such a way that you can make an informed decision on what is right for you.

Compression

At first glance, the term compression seems intuitively obvious. Most people have at one time or another extracted a compressed Zip windows file. Examining the file sizes pre and post extraction reveals there is more data on the hard drive after the extraction. WAN compression products use some of the same principles, only they compress the data on the WAN link and decompress it automatically once delivered, thus saving space on the link, making the network more efficient. Even though you likely understand compression on a Windows file conceptually, it would be wise to understand what is really going on "under the hood" during compression, before making an investment to reduce network costs. Some questions to consider: How does compression really work? Are there situations where it may not work at all?

How Compression Works

A good, easy-to-visualize analogy to data compression is the use of shorthand when taking dictation. By using a single symbol for common words, a scribe can take written dictation much faster than if he were to spell out each entire word. Thus, the basic principle behind compression techniques is to use shortcuts to represent common data. Commercial compression algorithms, although similar in principle, vary widely in practice. Each company offering a solution typically has its own trade secrets, which they closely guard for a competitive advantage.

There are a few general techniques common to all strategies. One technique is to encode a repeated character within a data file. For a simple example, let's suppose we were compressing this very document, and as a format separator we had a row with a solid dash (i.e." ____________").

The data for this solid dashed line is comprised of approximately 160 times the ASCII character "-". When transporting the document across a WAN link without compression, this line of document would require 80bytes of data. However, with clever compression, we can encode this using a special notation "- X 160". The compression device at the front-end would read the 160 character line and realize: "Duh, this is stupid. Why send the same character 160 times in a row?" It would then incorporate the special code to depict the data more efficiently. Perhaps that was obvious, but it is important know a little bit about compression techniques to understand the limits of their effectiveness.

There are many types of data that cannot be efficiently compressed. For example, many image and voice recordings are already optimized. There is very little improvement in data size that can be accomplished with compression techniques on image and voice recordings.

Summary of Key Concepts

Compression - *Relies on data patterns that can be represented more efficiently. Best-suited for point-to-point leased lines.*

Caching - *Relies on human behavior, accessing the same data over and over. Best-suited for point-to-point leased lines, also viable for Internet Connections and clogged VPN tunnels.*

Protocol Spoofing – *Best-suited for point-to-point WAN links.*

Application Shaping - *Controls data usage based on spotting specific patterns in the data. Best-suited for both point-to-point leased lines and Internet connections. Very expensive to maintain in both initial and ongoing costs, and also in labor spent.*

Equalizing - *Makes assumptions on what needs immediate priority based on the data usage. Excellent choice for Internet Connections and clogged VPN tunnels.*

Connection Limits - *Prevents access gridlock in routers and access points. Best suited for Internet access where peer-to-peer (P2P) usage is clogging your network.*

Simple Rate Limits - *Prevents one user from getting more than a fixed amount of data. Best suited as stop gap first effort for a remedying a congested Internet Connection with a limited budget.*

When considering a compression technology, *we recommend asking the companies that sell compression-based solutions to provide you with profiles on what to expect, based on the type of data sent on your WAN link.*

Caching

Suppose you are the Administrator for a network, and you have a group of a 1000 users that wake up promptly at 7:00 am each morning and immediately go to MSNBC.com to retrieve the latest news from Wall Street. This synchronized behavior would create 1000 simultaneous requests for the same remote page on the Internet.

Or, in the corporate world, suppose the CEO of a multinational 10,000 employee business, right before the holidays, put out an all-points 20 page PDF file on the corporate site describing the new bonus plan? As you can imagine, all the remote WAN links might get bogged down for hours while each and every employee tried to download this file.

Well, it does not take a rocket scientist to figure out that if somehow the MSNBC home page could be stored locally on an internal server, it would alleviate quite a bit of pressure on your WAN link.

And in the case of the CEO memo, if a single copy of the PDF file was placed locally at each remote office, it would alleviate the rush of data.

Caching does just that. *Caching is offered by various vendors, and can be very effective in many situations*. Vendors can legitimately make claims of tremendous WAN speed improvement in some situations. *Caching servers have built-in intelligence to store the most recently and most frequently requested information, thus preventing future requests from traversing the WAN link unnecessarily*.

You may know that most desktop browsers do their own form of caching already. Many web servers keep a timestamp of their last update to data, and browsers, such as the popular Internet Explorer, will use a cached copy of a remote page after checking the timestamp.

So what is the Downside of Caching?

There are two main issues that can arise with caching:

1) Keeping the cache current. If you access a cached page that is not current, then you are at risk of getting old and incorrect information. Some things you may never want to be cached, for example, the results of a transactional database query. It's not that these problems are insurmountable, but there is always the risk that the data in cache will not be synchronized with changes.

2) Volume. There are 234 million websites out on the Internet alone (as of December, 2009 per internet-2009-in-numbers) . Each site contains upwards of several megabytes of public information. The amount of data is staggering. Even the smartest caching scheme cannot account for the variation in usage patterns among users, and the likelihood that they will hit an un-cached page.

Protocol Spoofing

Historically, there are client/server applications that were developed for an internal LAN. Many of these applications are considered chatty. For example, to complete a transaction between a client and server, 10's of messages may be transmitted, when perhaps one or two would suffice. Everything was fine until companies (for logistical and other reasons) extended their LANs across the globe, using WAN links to tie different locations together.

To get a better visual on what goes on in a chatty application, perhaps an analogy will help with getting a picture in your mind. Suppose you were sending a letter to family members with your summer vacation pictures; and, for some insane reason, you decided to put each picture in a separate envelope and mail them individually on the same mail run. Obviously, this would be extremely inefficient.

What protocol spoofing accomplishes is to fake out the client or server side of the transaction and then send a more compact version of the transaction over the Internet, i.e. put all the pictures in one envelope and sends it on your behalf thus saving you postage...

You might ask why not improve the inefficiencies in these chatty applications rather than write software to deal with the problem?

Good question, but that would be the subject of a totally different white paper on how IT organizations must evolve their legacy technology. It's beyond the scope of our white paper to answer that.

Application Shaping

One of the most popular and intuitive forms of optimizing bandwidth is a method called "application shaping", with aliases of "traffic shaping", "bandwidth control", and perhaps a few others thrown in for good measure. For the IT Manager that is held accountable for everything that can and will go wrong on a network, or the CIO that needs to manage network usage policies, this is a dream come true. If you can divvy up portions of your WAN link to various applications, then you can take control of your network, and insure that important traffic has sufficient bandwidth.

At the center of application shaping is the ability to identify traffic by type. Is this Citrix traffic, streaming audio, Kazaa peer-to-peer or something else?

The Fallacy of Internet Ports and Application Shaping

Many applications are expected to use Internet ports when communicating across the Internet. An Internet port is part of an Internet address, and many firewall products can easily identify ports and block or limit them. For example, the "FTP" application, commonly used for downloading files, uses the well known "port 21". The fallacy with this scheme, as many operators soon find out, is that *there are many applications that do not consistently use a fixed or standard port for communication*.

Many application writers have no desire to be easily classified. In fact, they don't want IT personnel to block them at all; so they deliberately design applications to not conform to any formal port assignment scheme. For this reason, any product that purports to block or alter application flows by port should be avoided, if your primary mission is to control applications by type.

So, if standard firewalls are inadequate at blocking applications by port, what can help?

As you are likely aware, all traffic on the Internet travels around in what is called an IP packet. An IP packet can very simply be thought of as a string of characters moving from Computer A to Computer B. The string of characters is called the "payload," much like the freight inside of a railroad car. On the outside of this payload, or data, is the address where it is being sent. These two elements, the address and the payload, comprise the complete IP packet. In the case of different applications on the Internet, we would expect to see different kinds of payloads. For example, let's take the example of a skyscraper being transported from New York to Los Angeles. How could this be done using a freight train? Common sense suggests that one would disassemble the skyscraper, stuff it into as many freight cars as it takes to transport it, and then hope that when the train arrived in Los Angeles, the workers on the other end would have the instructions on how to reassemble the building.

Well, this analogy works with almost anything that is sent across the Internet; only the payload is some form of data, not a physical hunk of bricks, metal, and wires. If we were sending a Word document as an e-mail attachment, guess what? The contents of the document would be disassembled into a bunch of IP packets and sent to the receiving e-mail client, where it would be re-assembled. If I looked at the payload of each Internet packet in transit, I could actually see snippets of the document in each packet, and could quite easily read the words as they went by.

At the heart of all current application shaping products is special software that examines the content of IP packets, and *through various pattern-matching techniques, determines what type of application a particular flow is*. Once a flow is determined, then the application shaping tool can enforce the operator's policies on that flow.

Some examples are:

- Limit AIM messenger traffic to 100kbs
- Reserve 500kbs for Shoretell voice traffic

The list of rules you can apply to traffic types and flows is unlimited.

The Downside to Application Shaping

Application shaping does work, and is a very well-thought out logical way to set up a network. After all, complete control over all types of traffic should allow an operator to run a clean ship, right? But as with any euphoric ideal, there are drawbacks to the reality that you should be aware of:

1) The number of applications on the Internet is a moving target. The best application shaping tools do a very good job of identifying several thousand of them; and yet there will always be some traffic that is unknown (estimated at ten percent by experts from the leading manufacturers). The unknown traffic is lumped into the "unknown" classification, and an operator must make a blanket decision on how to shape this class. Is it important? Is it not? Suppose the important traffic was streaming audio for a webcast, and is not classified. Well, you get the picture. Although theory behind application shaping by type is a noble one, the cost for a company to keep current is large and there are cracks.

2) Even if the application spectrum could be completely classified, *the spectrum of applications constantly changes*. You must keep licenses current to insure you

have the latest in detection capabilities. And even then it can be quite a task to constantly analyze and change the mix of policies on your network. As bandwidth costs lessen, how much human time should be spent divvying up and creating ever more complex policies to optimize your WAN traffic?

Equalizing

Take a minute to think about what is really going on in your network to make you want to control it in the first place.

We can only think of a few legitimate reasons to do anything at all to your WAN: "The network is slow", or "My VoIP call got dropped".

If such words were never uttered than life would be grand.

So you really only have to solve these issues to be successful. Who cares about the actual speed of the WAN link, or the number and types of applications running on your network, or what port they are using, if you never hear these two complaints?

How Equalizing Works

Equalizing goes to the heart of congestion, using the basic principal of time. The reason why a network is slow or a VoIP call breaks up is that the network is stupid. The network grants immediate access to anybody who wants to use it, no matter what their need is. That works great much of the day, when networks have plenty of bandwidth to handle all traffic demands, but it is the peak usage demands that play havoc.

Take the above statement with some simple human behavior factors. People notice slowness when real-time activities break down, like accessing a web page, sending an e-mail, running a chat session, or placing a VoIP call. All of these activities will generate instant complaints if response times degrade from the "norm".

The other fact of human network behavior is that there are bandwidth-intensive applications, such as peer-to - peer, large e-mail attachments, and database backups. These bandwidth-intensive activities are attributed to a very small number of active users at any one time, which makes it all the more insidious as they can consume well over ninety percent of a network's resources at any time. Also, most of these bandwidth-intensive applications can be spread out over time without the user noticing.

That database backup, for example. Does it really need to be completed in three minutes at 5:30pm on a Friday, or can it be done over six minutes and complete at 5:33pm? That would give your network perhaps fifty percent more bandwidth at no additional cost, and nobody would notice. It is unlikely the user backing up their local disk drive is waiting for it to complete with stopwatch in hand.

It is these unchanging human factor interactions that allow equalizing to work today, tomorrow, and well into the future. *Equalizing looks at the behavior of the applications and usage patterns*. By adhering to some simple rules of behavior, the real-time applications can be differentiated from the heavy non-real-time activities, and thus be granted priority on the fly, *without needing any specific policies to be set by the IT Manager*.

How Equalizing Technology Balances Traffic

Each connection on your network constitutes a traffic flow. Flows vary widely from short dynamic bursts, such as when searching a small website, to large persistent flows, as when performing peer-to-peer file-sharing.

Equalizing is determined from the answers to these questions:
1) *How persistent is the flow?*
2) *How many active flows are there?*
3) *How long has the flow been active?*
4) *How congested is the overall network trunk?*
5) *How much bandwidth is the flow using, relative to the network trunk size?*

Once these answers are known, then *Equalizing makes adjustments to flows by adding latency to low-priority tasks, so high-priority tasks receive sufficient bandwidth*. Nothing more needs to be said and nothing more needs to be administered to make it happen, and once set up it need not be revisited.

Exempting Priority Traffic

Many people often point out that although equalizing technology sounds promising, it may be prone to mistakes with such a generic approach to traffic shaping. For example, what if a user has a high-priority bandwidth-intensive video stream that must get through; wouldn't this be the target of a misapplied rule to slow it down?

The answer is yes; but, what we have found is that high-bandwidth priority streams are usually few in number, and known by the administrator. They rarely if ever pop up spontaneously, so *it is quite easy to exempt such high-priority flows*, since they are the rare exception. This is much easier than trying to classify every flow on your network at all times.

Connection Limits

Often overlooked as a source of network congestion is the number of connections a user generates. *A connection can be defined as a single user communicating with a single Internet site.*

For example, take accessing the Yahoo home page. When you access the Yahoo home page, your browser goes out to Yahoo and starts following various links on the Yahoo page to retrieve all the data. This data is typically not all at the same Internet address, so your browser may access several different public Internet locations to load the Yahoo home page, perhaps as many as ten connections over a short period of time. Routers and access points on your local network must keep track of these "connections", to insure that the data gets routed back to the correct browser.

Although ten connections to the Yahoo home page is not excessive, there are very poorly behaved applications, otherwise known as peer-to-peer applications (most notably Gnutella, Bear Share, and Bittorrent), which are notorious for opening up 100's or even 1000's of connections in a short period of time. This type of activity is just as detrimental to your network as other bandwidth-eating applications, and can bring your network to a grinding halt.
The solution is to make sure any traffic management solution deployed incorporates some form of connection-limiting features.

Simple Rate Limits

The most common and widely used form of bandwidth control is the simple rate limit. This involves putting a fixed rate cap on a single IP address, as often is the case with rate plans promised by ISPs to their user community. "2 meg up and 1 meg down" is a common battle cry, but what happens in reality with such rate plans?

Although setting simple rate limits is far superior to running a network wide open, we often call this strategy "set, forget, and pray"!

Take for example six users sharing a T1. If each of these six users gets a rate of 256kbs up and 256kbs down, then these six users each using their full share of 256 kilo bits per second is the maximum amount a T1 can handle. Although it is unlikely that you will hit gridlock with just six users, when the number of users reaches thirty, gridlock becomes likely, and with forty or fifty users, it becomes a certainty, and will happen quite often.

It is not uncommon for Schools, wireless ISPs, and Executive Suites to have 60-200 users sharing a single T1, with simple fixed user rate limits as the only control mechanism.

Yes, *simple fixed user rate limiting does resolve the trivial case where one or two users, left unchecked, can use all available bandwidth*. However, unless your network is not oversold, there is never any guarantee that busy-hour conditions will not result in gridlock.

Conclusion

The common thread to all WAN optimization techniques is *they all must make intelligent assumptions about data patterns or human behavior to be effective*. After all, in the end, the speed of the link is just that, a fixed speed that cannot be exceeded. All of these techniques have their merits and drawbacks. The trick is finding a solution or combination of solutions best-suited for your network needs. Hopefully the background information contained in this document will help you to make an informed decision.

About APconnections, Inc.

APconnections is based in Lafayette, Colorado, USA.
We develop cost-effective, easy-to-install and manage, traffic shaping appliances. Our NetEqualizer product family optimizes critical network bandwidth resources for any organization that purchases bandwidth in bulk and then redistributes or resells that bandwidth to disparate users with competing needs.

We released our first commercial offering in July 2003, and since then customers around the world have put our products into service. Our flexible and scalable solutions can be found at ISPs, WISPs, major universities, Fortune 500 companies, SOHOs and small businesses on six continents.

About the NetEqualizer Product Family

NetEqualizer appliances are bandwidth shaping systems designed to optimize your Internet Connection, while giving priority to your important business and data applications. The flexible, scalable, and cost-effective bandwidth control products can be deployed in both corporate and service provider networks.

NetEqualizer is available in a range of configurations from 2Mbps up to 5 gigabits.