TCP/IP Optimization over Wide Area Networks: Implications for Teleradiology

Steve G. Langer,¹ Todd French,¹ and Colin Segovis¹

Radiology examinations are large. The advent of fast volume imaging is making that statement truer every year. PACS are based on the assumption of fast local networking and just-in-time image pull to the desktop. On the other hand, teleradiology has been developed on a push model to accommodate the challenges of moderate bandwidth, high-latency wide area networks (WANs). Our group faced the challenging task of creating a PACS environment that felt local, while pulling images across a 3,000-mile roundtrip WAN link. Initial tests showed WAN performance lagging local area network (LAN) performance by a factor of 30 times. A 16-month journey of explorations pulled the WAN value down to only 1.5 times slower than the LAN.

KEY WORDS: Enterprise PACS, wide area network (WAN), teleradiology

BACKGROUND

PACS and teleradiology are two very different radiology workflow models, as befits their origins. PACS design assumes high-speed local area networking (LAN), and that users can pull images from a central core to any location on a just-in-time basis.¹ Teleradiology assumes high-latency wide area networks (WAN) that preclude large data moves in a just-in-time manner, also because the transfer is across the internet, transfer-slowness encryption must be used.² Rather than pull, teleradiology models assume images will be pushed to one or more fixed locations, and that interpreting physicians can be notified when the images are at the target location. In both cases, a key design goal is to not make the radiologist wait; in the PACS case, this is accomplished with high-speed image delivery and on teleradiology systems

by only notifying the radiologist the exam is available when it is at his location.

We have alluded to different performance between WAN and LAN links, but we have not described physically why that is the case. For instance, it is entirely possible and indeed likely that a 155-Mbps (million bits per second) link from New York to California would take longer to move a 10 MB file than a 45-Mbps link across New York City. Consider the following real-world observation; our institution reads exams from sister sites in the surrounding three states. Many of the WAN links to those sites are 155 Mbps, but the measured transfer speed averages about four computed tomography (CT) images per second (even after hours). Four CT images equate to 2 MB or 8 Mbits—so the observed transfer speed is about 8/155 or 1/20 of the stated network “speed.” How is this possible? Because the network performance numbers quoted here refer to the bandwidth of a given link. This should not be confused with the speed of a specific network transfer across a link. An analogy is useful.

Consider a two lane road, one lane is eastbound and one, west. Further, assume the cars follow the rules of transmission control protocol/internet protocol (TCP/IP). Consider a “conversation” to be the movement of cars up and down the road.

¹From the Mayo Clinic, Rochester, MN, USA.

Correspondence to: Steve G. Langer, Mayo Clinic, Rochester, MN, USA; e-mail: langer.steve@mayo.edu

Copyright © 2010 by Society for Imaging Informatics in Medicine

Online publication 11 June 2010

doi: 10.1007/s10278-010-9309-2

The road can carry multiple cars at once that belong to different companies. Now, consider that Company X has one four-passenger car and needs to move eight people from the West to the East office. The first eastbound car has three passengers and the driver. Once the driver reaches the eastbound destination, she has to return to the West to get three more passengers. It will take her three trips to transport all eight people to the East, and then she still has to return the car to the West office for further uses. If the speed limit on the road is strictly enforced, then the only ways to carry more passengers faster are to

- (a) Use a bigger vehicle or
- (b) Use multiple cars in parallel (i.e., a multi-association network application), but this requires more lanes on the road (i.e., higher bandwidth).

In this TCP/IP analogy, the car is a data packet; passenger capacity is the packet frame size. The eastbound packet is the data payload. The westbound packet is the "Acknowledgement" that the data packet arrived and that it is safe to send the next packet. The speed limit is the speed of an electron in a wire, or a photon in a fiber optic. One can add more lanes to carry more traffic, but that does not improve the speed for a given network association unless the application can send multiple packets in parallel. For the vast majority of PACS applications, the reality is a single network association between workstation and server, which is limited by the round trip physics of electrons or photons.

Herein lies the key point to optimize TCP/IP traffic across high-latency (long distance) WANs when one cannot alter the application's behavior: reduce the number of packet round trips. This can be done by

- (a) Lossless data compression to remove redundant packets
- (b) Expanding the default packet frame size
- (c) Caching any redundant data at the requesting site

The challenge assigned to our group was to permit a Radiologist to work and live at a site 1,500 miles from where the PACS is, and yet use a workstation connected to that PACS with performance expectations as if it were just across the hallway. To assess the feasibility and possible approaches to this problem requires a bit more information.

THEORY

One often sees authors approximate network performance with the following equation:

$$\text{Transfer time} = \text{File size} / \text{Network bandwidth} \quad (1)$$

where network bandwidth is usually described in megabits per second.³ This equation is only accurate, however, in the limit of an unloaded network with low latency. If the network is heavily loaded with other traffic, the bandwidth is not all available for a single network association, and if the latency is long, the packet round trip time (RTT) can reduce the network utilization for a single association far below the stated bandwidth. For instance, in tests between sites in Minnesota and Arizona near midnight when other traffic is minimal, our tests were only able to achieve 3–5% of the rated bandwidth utilization on a 155-Mbps link. This is in contrast to the same computers on a 1-Gbps LAN link with sub-millisecond latency where bandwidth utilization over 50% is observed. The governing equation for network performance over a single association (as alluded to by the freeway example above) is given by

$$\text{Transfer time} = \text{Number of packets} \times \text{RTT} \quad (2)$$

It is obvious that for fixed network latency or RTT, the only method available to reduce the transmission time for a single network association is to reduce the number of packets. This can be done by lossless data compression, enlarging each packet's payload, caching previously moved data to avoid resends, using a less chatty protocol, or a combination.

To understand the possible avenues for modifying the packet count, it is useful to understand some basics of TCP/IP. A review of networking trade journals will show articles on "Layer 3 switches" and similar expressions. The layers referred to are a result of the original formulations by the Internet Engineering Task Force (IETF) of what has come to be known as TCP/IP. Basically, if one starts at the physical layer (the network interface card (NIC)), the naming convention is physical or link layer (layer one), internet layer

(layer two), transport layer (layer three), and the application layer (layer four).^{4,5} Several years later, the International Standards Organization created the seven-layer Open Systems Interconnect model, which can lead to confusion if one does not know which system is being referenced.⁶ Table 1 shows a mapping of the two reference models.

The anatomy of a TCP/IP packet is a result of the challenge the IETF designers faced: guarantee that the bytes comprising a file(s) were all moved from site A to site B without loss, corruption, or reordering. The answer uses several methods. To prevent loss, the designers implement the “send and acknowledge” scheme. That is, send a packet with an ID number and wait for the receiver to send an acknowledgement of that packet ID. To prevent corruption, compute a number that is unique for the payload (called a “checksum”) and await the Acknowledgement packet to come back with the same checksum for that packet ID. Finally, ordering is guaranteed because packet 2 is not sent until receipt of packet 1 with a proper checksum is acknowledged, and so on. Given this description, the following diagram of a TCP/IP packet should make sense (Table 2).

Of immediate interest is the variable size of the payload section. The default size of this maximum transfer unit (MTU) is 1,500 bytes, but it can be increased to a maximum of 9,000 bytes under the rules of IP4.⁷ Naively applying Eq. 2 would predict a sixfold decrease in packet count with a concomitant reduction in send times. Furthermore, since the definition of the packet is at layer two, improvements benefit all networking applications. This is a decidedly elegant solution, because if it works, it automatically accelerates all networking

Table 1. Approximate Mapping of Internet Engineering Task Force (IETF) Networking Model Layers to Open Systems Interconnect (OSI) Model Layers

IETF model	OSI model
	Application
Application	Presentation
	Session
Transport (TCP)	Transport
Internet (IP)	Network
	Data link
Physical	Physical

The lowest layer is the physical and progresses up through the application layer, four layers in the IETF model and seven in the OSI

Table 2. The Anatomy of a Transmission Control Protocol/Internet Protocol Packet Showing Header and Payload Details

Byte offset	Elements
0	Source address Destination address
32	Sequence number (ID)
64	Acknowledgement number
96	Flags Window size
128	Check sum Urgent flag
160	Options
192 to 1,500 (up to 9,000)	Payload

applications without the complication of altering the behavior of every networked application (e.g., mail, web, and DICOM).

As simple as it appears, however, setting the MTU to 9,000 is not supported by all networking devices, and to be effective, it has to be applied to all devices that comprise a given route. In fact, the feature is referred to as a Jumbo Frames, and often not all devices along a route support it.⁸ Because of this often encountered difficulty, WAN acceleration is a broad and complex discipline that has evolved many strategies and vendors that specialize in it. This will become clearer in the next sections.

METHODS

Typically, teleradiology applications push images among sites with DICOM file transfers. However, the mission assigned to our group was a PACS workstation querying images from a server 1,500 miles away. Image moves within a PACS are not normally done via DICOM services because of the overhead of that protocol. Rather, vendors typically use LAN protocols for image pulls to the PACS workstation. The protocols of interest for our team are common internet file system (CIFS), file transfer protocol (FTP), hyper text transfer protocol (HTTP), and network file system (NFS). To emulate the baseline conditions of the real-world problem, we first constructed the following test network in our lab (see Fig. 1).

In Figure 1, the real-world conditions were closely mimicked to include a file server, a Cisco Catalyst 3000 switch (Cisco Systems, San Jose,

Fig 1. The test system configured to accurately replicate the current production system. The Simena appliance permits injecting controlled latency and measures network variables.

CA, USA), a Simena network simulator to provide the observed WAN latency (Simena, Sterling, VA, USA), another Cisco switch, and finally, a client computer. Network statistics were acquired with WireShark version 1.2.4 (<http://www.wireshark.org/>). The real-world latency between the production PACS client (in Minnesota) and server (in Arizona) was measured with the “ping” and “trace-route” commands and observed to be 53 ms. This value was halved and programmed into the Simena to simulate the RTT of the real-world WAN link for some of our tests; additional latencies were also used. The server and client computers were both based on Dell Precision 690 workstations with a 1 Gbit network interface, 8 GB of RAM, 15,000 RPM SCSI disks, and a 2-GHz dual-core processor (Dell Corporation, Round Rock, TX, USA).

The benefits of this simplified experimental setup were

- Accurate replication of all the principal components in the real-world system
- Validation that the test system accurately replicated real-world performance when all the configurations were identical to those in production
- Isolating the effects of “simple” network settings and measuring the effect with the Simena statistics tools. In particular, this allowed investigating what could be accomplished for no cost with the production equipment currently in place.

To observe the effects of using various WAN accelerator products, the test system shown in Figure 1 was augmented to include those appliances in the following manner (Fig. 2).

Fig 2. The test system augmented with wide area network (WAN) acceleration appliances at the WAN edge.

To eliminate the effect of client caching giving artificially high performance figures, all disk and RAM caches were cleared on the client side between tests and new files used for each test. To eliminate accidentally measuring normal network chatter, the NIC on the client computer was under programmatic control: a script turned on the NIC, started a timer, performed the file transfer, stopped the timer, and then shutdown the NIC. For all “standard networking hardware” tests, the Ciscos and Simena were set to an MTU value of 9,000 bytes, while the MTU size on the server and clients were adjusted from the default 1,500 up to 9,000 bytes.

RESULTS

Using Standard Networking Hardware

The goal in this test series was to discover the optimal performance that could be gained with standard networking hardware and protocols that were already in use and would require no further money to develop. For the file server, we chose FreeNAS version 7 (<http://freenas.org/>), which is an optimized open source file server appliance based on 64-bit FreeBSD (<http://freebsd.org/>). Similarly, the FTP, HTTP, CIFS, and NFS clients used were based on FreeBSD version 8 default tools. The system geometry followed that of Figure 1.

While FTP and NFS protocols are relatively simple, there are a myriad of tunable variables in CIFS and HTTP. Therefore, to standardize methodology for this round of tests, we adopted the following conventions:

- FTP, NFS, and HTTP: default configuration on server and client
- CIFS 16: default configuration on server and client (buffers equal 16 KB)
- CIFS 64: increased send/receive buffers on server to 64 KB.

We initially attempted HTTP tuning but returned to default values when we could not improve

performance. The results are summarized in Table 3 for various protocol types and network latencies. The latency values are chosen to relate to real-world scenarios: 0.1 ms mimics LAN latency, 25 ms would be sites within 300–500 miles, 50 ms 1,500 miles, and up to 800 ms, which simulates geosynchronous satellite latency. Note that for the non-CIFS protocols, the only parameter adjusted in this round was the MTU setting, a layer 2 option set at the networking card. The effects of all changes are seen in the final two columns of Table 3: “No. of packets” and “Kbps.” The bits per second results are the averages of three trials.

Given the equivalent payload for each test, it is immediately apparent that all protocols have a different inherent efficiency (i.e., more efficient protocols can transfer the same payload with fewer packets). It is also true that packet counts vary for the same protocol as latency varies, showing that transmission is imperfect and often requires resends. Note also that because the MTU is at the transport layer, increasing it reduced the packet counts for all networking protocols. However, effective bits per second was not universally improved. Both FTP and NFS were notably accelerated with the larger MTU, but CIFS and HTTP showed more complex behavior. In the case of CIFS, speed did not improve much until the send buffer size was increased from 16 to 64 KB; this tripled performance at the longer latencies over the default setting (16 KB buffer and 1,500 MTU). For HTTP, packet count was reduced by almost 500% across the board with the larger MTU, but kilobits per second is more complex. When total latency is small (<50 ms), the MTU 9,000 results are better than the 1,500 case; as latency increases to satellite times, the advantage disappears. Clearly, this points to the need to look for other HTTP configuration optimizations.

Using an Open Source Transmission Tool: CTP

As part of its mission of advancing radiology, the Radiological Society of North America releases several open source software tools. One of these, the clinical trial processor (CTP) was originally created to reside at a clinical research site, take in patient identified scans on the DICOM protocol, replace the real patient information with a study alias, and then forward the images over HTTP protocols to the research center. However,

Table 3. The Effect of Maximum Transfer Unit Changes on Packet Count and Bits Per Second for Common Internet File System, File Transfer Protocol, Hyper Text Transfer Protocol, and Network File System Protocols

Latency ms	FTP-1,500		FTP-9,000		NFS-1,500		NFS-9,000		HTTP-1,500		HTTP-9,000		CIFS16-1,500		CIFS16-9,000		CIFS64-1,500		CIFS64-9,000	
	No. of packet	Kbps	No. of packet	Kbps	No. of packet	Kbps	No. of packet	Kbps	No. of packet	Kbps	No. of packet	Kbps	No. of packet	Kbps	No. of packet	Kbps	No. of packet	Kbps	No. of packet	Kbps
0.1	305	41,046	138	40,947	902	257,595	260	249,992	1,196	236,701	220	348,657	370	263,207	215	14,819	345	36,745	89	118,713
25	555	7,508	135	7,977	902	18,442	262	17,399	1,334	9,727	224	17,448	393	4,041	188	3,474	363	4,234	132	4,277
50	606	3,949	143	4,314	902	10,093	273	9,259	1,263	7,472	222	8,003	421	1,900	206	1,574	359	2,134	137	2,867
100	542	2,024	146	2,174	906	4,251	262	4,454	1,251	4,295	229	4,165	409	893	191	976	328	1,069	120	2,308
200	566	1,051	133	1,090	911	2,082	267	2,355	1,263	2,261	232	2,217	419	477	192	490	345	550	107	1,311
400	555	539	140	507	918	1,228	269	1,129	1,259	1,117	225	1,097	446	227	203	232	362	269	122	678
800	527	279	146	286	926	506	293	520	1,244	589	233	602	439	122	197	124	389	135	138	328

The CIFS results also vary based on the CIFS send/receive buffer size on the server; 16 (CIFS 16) and 64 KB (CIFS 64)

by simply modifying its configuration, CTP can also be used to forward fully identified DICOM information over a secure HTTP link.

For this series of tests, both server and client computers were reloaded with the Windows XP32 operating system. The setup mimicked the design shown in Figure 1. The Sun Java 1.6 runtime engine was used to run CTP on both ends of the connection (Sun Microsystems, Santa Clara, CA, USA). Trials were conducted with the transmission over the “WAN” leg being either unencrypted DICOM or encrypted HTTPS. Table 4 contains the results for transmission of 20 CT images. For all these tests, MTU settings remained at 1,500, and the CTP configurations are shown in Figures 3 and 4.

Using Commercial WAN Accelerators

For this series, we applied the commercial products to the production PACS network, the topology resembled that in Figure 2, except that in place of the Simena was the real 53 ms latency WAN between Minnesota and Arizona. The nature of the installation precluded the measurement procedures followed heretofore (using WireShark).

In general, the commercial products tested attempt to apply a pipeline of optimizations: compression, caching previously moved data, and minimizing the overhead of chatty protocols such as CIFS. Unfortunately, the details of our production PACS implementation defeated much of this strategy. We have investigated WAN optimizers from two vendors (Cisco WAAS, San Jose, CA, USA, and Silver Peak NX5600, Santa Clara, CA, USA). The PACS in question already applies lossless compression as the exams are acquired

and stored to the archive. Hence, further attempts to compress by the WAN optimizers proved ineffective. Further, since the second stage in the pipeline relies on caching redundant data, it was also largely defeated for two reasons:

- (a) Since the data arrived already compressed, much of the redundancy was already reduced and
- (b) Since it is rare that the same exam was pulled more than once in a given 48-h period, the caches rarely saw the same bit pattern twice.

The final step, optimizing the chattiness of the specific protocol was only able to garner a few percent throughput enhancement. It should be noted, however, that sites whose PACS do not pre-compress the image data would likely benefit greatly from these devices. For example, moving standard files (e.g., operating system update files 10–35 MB in size), the effect of both products was a six- to eightfold boost in transfer speed and a tenfold reduction in packets and bytes sent.

Before giving up, each vendor invested the effort to return to their labs to develop work-arounds; in the end, each solved the problem differently. Their final efforts resulted in both vendors gaining a significant acceleration of six to eight times faster than baseline WAN performance (as measured by CT images per second via stopwatch) and approximately a tenfold reduction in packets (as measured by the vendor software).

DISCUSSION

Our group undertook this work to support our radiologists reading cases from a sister site over 1,500 miles away. The practice desire was to use PACS as usual, in a just-in-time pull based model. Original benchmarks showed performance, as measured by images per second pulled across the network was about 30 times slower than LAN performance. We successively applied the lessons learned in this work to the production system. First, we enabled compression on the PACS, which was a win for both local storage and transmission times across the WAN. Next, we altered the CIFS send buffer from 16 to 64 KB. From Table 3 for 50 ms latency, we see this results in a 1.3 (320/242) improvement. Unfortunately, we

Table 4. Performance of the Radiological Society of North America Clinical Trials Processor (CTP) for DICOM and HTTPS Protocols

Latency (ms)	DICOM		HTTPS	
	No. of packets	Kbps	No. of packets	Kbps
0.1	8,731	85,801	5,150	18,895
25	8,720	9,385	5,218	4,045
50	8,814	4,952	5,129	1,789
100	8,776	2,551	5,550	1,344
200	8,825	1,053	5,587	733
400	8,863	522	6,004	358
600	8,853	429	5,543	236
800	8,909	301	5,458	173

Listing: HTTP

a) Sender

```
<ExportService
name="HTTP Export"
id="HTTP export"
class="org.rsna.ctp.stdstages.HttpExportService"
root="roots/http-export"
url="https://9.9.9.9:1443"
zip="yes"
acceptFileObjects="yes"
acceptDicomObjects="yes"
acceptZipObjects="yes"
acceptXmlObjects="yes"
dicomScript="scripts/df.script"
zipScript="scripts/zf.script"
xmlScript="scripts/xf.script"
interval="5000"/>
```

b) Receiver

```
<ImportService
name="HTTP Import"
class="org.rsna.ctp.stdstages.HttpImportService"
root="roots/http-import"
ssl="yes"
port="1443"
zip="no"
acceptFileObjects="yes"
acceptDicomObjects="yes"
acceptZipObjects="yes"
acceptXmlObjects="yes"
quarantine="quarantines/StorageServiceQuarantine" />
```

Fig 3. The clinical trial processor application configuration for (a) sending and (b) receiving files using the HTTPS protocol.

could not enjoy the benefits of a 9,000 MTU because we do not “own” all the intervening networking switches on the route between our two sites. Because of this, the first unit to only support MTU 1500 essentially forces all the other network switches to also drop back to that MTU size.

Applying the combination of methods listed above (2.5:1 lossless compression and 1.3:1 CIFS buffer optimization), we achieved a system performance improvement of about 3.25 on the WAN. Put another way, rather than being 30 times

slower than the LAN, the WAN was now “only” 9.2 times slower. While not yet equal, this improvement combined with some workflow modifications allowed the practice to move forward.

Finally, to achieve further acceleration, we trialed several WAN optimization appliances to accelerate the application—this resulted in an additional sixfold acceleration from the 3.25 WAN performance gain mentioned above. Hence, 9.2/6 yields 1.5—the WAN is now only 50% slower than the LAN.

Listing: DICOM

a) Sender

```
<ExportService
name="DICOM Export"
class="org.rsna.ctp.stdstages.DicomExportService"
root="roots/pacs-export"
url="dicom://ctp-wrk:ctp-srv@9.9.9.9:1104"
zip="yes"
acceptZipObjects="yes"
acceptDicomObjects="yes"
acceptXmlObjects="yes"
dicomScript="scripts/df.script"
zipScript="scripts/zf.script" />
```

b) Receiver

```
<ImportService
name="DICOM Import"
class="org.rsna.ctp.stdstages.DicomImportService"
root="roots/dicom-import"
port="1104"
acceptFileObjects="yes"
acceptDicomObjects="yes"
acceptZipObjects="yes"
zip="yes"
quarantine="quarantines/StorageServiceQuarantine" />
```

Fig 4. The clinical trial processor application DICOM configuration for (a) sending and (b) receiving files.

CONCLUSIONS

Attempting to mimic “just-in-time” LAN performance on a high-latency WAN network is a daunting challenge. We have analyzed many approaches to the problem and believe we have struck on a mixture that maximizes what can be done with common technology and protocols, but to achieve performance that will scale as exam sizes grow, it would seem that fundamentally different protocols and infrastructure may need to be employed as health care providers attempt to do more—faster—at a distance. Internet2, a next generation internet implementation that avoids many of the limits of current TCP/IP models, will likely be needed (<http://www.internetnews.com/infra/article.php/3403161> and <http://www.internet2.edu/about/>).

REFERENCES

1. Langer SG, Wang J: An evaluation of ten digital image review workstations. *J Digit Imaging* 10(2):65–78, 1997
2. Langer SG, Stewart BK: Computer security: a primer. *J Digit Imaging* 12(3):114–131, 1999
3. Kimura H, Akatsuka T: Modeling and performance analysis of image transfer in PACS. *Stud Health Technol Inform* 52(Pt 2):1080–1084, 1998
4. RFC 1122: <http://tools.ietf.org/html/rfc1122>. Last accessed May 2009
5. RFC 1123: <http://tools.ietf.org/html/rfc1123>. Last accessed May 2009
6. Zimmermann H: OSI reference model. *IEEE Trans Commun* 28(4):425, 1980
7. MTU: http://en.wikipedia.org/wiki/Maximum_transmission_unit. Last accessed May 2009
8. RFC 1323: <http://tools.ietf.org/html/rfc1323>. Last accessed May 2009