TCP/IP Response Time Monitoring on z/OS

Abstract

- Response time measurement is critical to service level agreements. It is also critical in determining the impact of changes to your application and supporting systems (OS, database, network). Long or erratic response times can result in loss of productivity and even lost sales.
- This webcast discusses TCP/IP application end to end response time measurement, what it is, what it isn't, some of the pros and cons to various ways of gathering data, and the importance of breaking the data into network and host components.
- We cover a bit of 3270 historical perspective and then move to RFC2562, TN3270 and the present. From there we discuss measurement in another RFC-based application, HTTP, and finally cover more generalized non-RFC-based IP applications such as CICS sockets. We also talk about problems faced in measuring response time in a complex, multi-tiered environment.

Bio – or Who Is This Guy?

- Gary Bortner
- IT for 25 years and has worked over 16 years in mainframe networking covering roles in systems programming, software support, product installation, quality assurance, and training.
- Current: Senior Systems Engineer

  VIP (Vitalsigns visionnet IP monitor)

  SDS (Software Diversified Services)
- Prior – CA / Sterling / Interlink
- But enough about me…Let's talk about IP.

Agenda

- Describe the importance of monitoring response time on your computer network.
- What “response time” means starting with “simple” TN3270.

  (And what it is not)
- Ways to measure it, pros and cons.
- Evolution of measurement and application to increasingly complex applications.
- Present a few cases of response time monitoring.
- Note: focus here is on z/OS perspective.

Why It Matters (like I have to tell you this)

- Annoying unhappy users calling tech support.
- Lost opportunity — customers and clients go away.
- Poor return on investment — expensive new enterprise applications cannot deliver new efficiencies if the network can't support them.
- You need more people, more desks, more workstations to handle a given volume of transactions since each person can do fewer transactions in a given amount of time.
- It’s a good indicator of overall performance.
- Service-Level Agreements, internal and external, require documenting response time.

Poor response time costs you money...

What Is Response Time?

- For TN3270, it is the time from when a user hits enter until he sees a response back at his terminal.
- Round Trip Time (RTT): in TCP, this is the time it takes to send a datagram and get an acknowledgement.
- Is RTT Response Time?
- NO!!! - TCP Round Trip Time is only one component of response time for a TCP based application

What About Ping?

- What is a Ping?
  - **P**acket **I**nter**N**et **G**roper (bet you didn't know that – I didn't)
  - That's a "backronym" (bet you didn't know that term either – I didn't)
  - Per Wikipedia: "ping is a computer network tool used to test whether a particular host is reachable across an IP network. Ping works by sending ICMP "echo request" packets to the target host and listening for ICMP "echo response" replies. Using interval timing and response rate, ping **estimates** the round-trip time and packet loss rate between hosts. "
- This does not measure response time, though it is sometimes used to approximate it.
- It is also not RTT.
- It's not even TCP!
  - Ping times and RTT for the same IP pair often differ greatly

What Response Time Is (TN3270)

- User hits enter:
  - Request goes through client app (could be encrypted – SSL)
  - Goes through client stack (could be encrypted – IPsec/VPN)
  - Goes across network – routers, bridges, perhaps various topologies could be fragmented, could be decrypted by external IPsec/VPN device
  - TCPIP stack (could be decrypted – IPsec)
  - Telnet server (internal/external) (could be decrypted – SSL)
  - Session manager? – TPX, Multisess, etc.
  - SNA routing (could go back to IP (UDP) via EE)
  - Application – TSO / CICS / ?
  - Database calls?
  - This is getting long, and then the response has to reverse all this and follow the popcorn trail all the way back again.

Where Does RTT Fit in?

- It is just the TCP component of all this – TCP layer at the client stack to the TCP layer at the server stack.
- It does not include anything above the TCP layer
- It does not include application time to process requests, generate replies.
- It does not include SNA time to move messages through the SNA network.
- It does not include time to encrypt/decrypt via SSL

So How Do You Measure It?

- First let's decide WHERE to measure it
- 1) Client end
- 2) Host / server end

Client Side

Pros

- Monitor sits where user sits, so it can "feel his or her pain"
- Most accurate measurement.

Cons

- Only sees total response time, can't separate response time into its network and server components
- Need to deploy many monitors at many clients.

Server Side

Pros

- Able to isolate network from server and break measurement into components.
- Accurate measurement of server delays.
- Simpler deployment, lower cost.

Cons

- Doesn't measure **exactly** what the client sees (but the error is negligible).
- Misses some client activity, for instance, name resolution lookup time, which can matter for something like HTTP.

Active vs. Passive Monitoring

- An in-line device works like a router, monitoring inbound traffic and forwarding it.
- Pros
  - Can be moved around
  - Can have multiple devices and measure at different points.
- Cons
  - Costs for multiple devices
  - May not be where you want it to be when you want it there
  - An in-line device can itself become a point of failure.
    - Drop packets, causing TCP retransmits, slow start, even potentially dropping a connection.
    - Munge packets, corrupt data

Active vs. Passive Monitoring (cont)

- A pass-by monitor passively watches traffic go by; it is not responsible for an active action, I.E. forwarding traffic.
- Can be done in software
- Can be done on the server, or even in the actual server software
- Sitting closer to or in the server, it can provide most accurate measurement of server time

And the Winner Is:

- From a z/OS perspective, the best monitor is (IMHO):

Server-side

Pass-by (passive)

Building a Server Side Passive TN3270 Monitor

- There is a standard: RFC2562
- This is a standard for TN3270E

RFC2562

- Here's what it looks like:

```
 -------------------------------------------------
 |                                               |
 | Client            TN3270E          Target     |
 |                    Server          SNA Host   |
 |                   Timestamps                  |
 |                                               |
 | <---IP Network------><---SNA Network--->      |
 |                                               |
 |      request          D                       |
 | ------------------------------------------->  |
 |      reply(DR)        E                    |  |
 | <------------------------------------------<  |
 | |    +/-RSP           F                       |
 |  >------------------ - - - - - - - - - >      |
 |                                               |
 -------------------------------------------------
```

- Timestamp E minus timestamp D = "SNA" time
- Timestamp F minus timestamp E = IP time
- +/- response is thrown away if dynamically set by TN3270 server

RFC2562 Continued

- This breaks out the IP and the “SNA” components.
- It does “fudge” a bit on the IP side in that the IP component for a given request/response pair is not actually this:

```
 -----------------------------------------------
|                                               |
| Client            TN3270E          Target     |
|                    Server         SNA Host    |
|                   Timestamps                  |
|                                               |
| <---IP Network-------><---SNA Network--->     |
|                                               |
|      request         D                        |
| ------------------------------------------>   |
|      reply(DR)       E                    |   |
| <-----------------------------------------<   |
|                                               |
 -----------------------------------------------
```

- But over a large number of connections it should average out.

RFC2562 – a Problem

- This solution requires TN3270E and that Responses is successfully negotiated during session initiation.
- But not all TN clients support TN3270E.
- If the point of measurement is not in the TN3270 server, SSL will hide the TN3270 negotiation.
- Increased traffic with DR responses.

RFC2562 – a Solution

- Solution: Request a timing mark

```
 -----------------------------------------------
|                                               |
| Client            TN3270            Target    |
|                    Server          SNA Host   |
|                   Timestamps                  |
|                                               |
| <---IP Network------->&lt;---SNA Network--->      |
|                                               |
|      request         D                        |
| ------------------------------------------->  |
|      reply           E                    |   |
| <-----------------------------------------<   |
|      TIMING MARK req E                        |
| <-----------------------------------------<   |
| |    TIMING MARK rsp F                        |
|  >-------------------- - - - - - - - - - >    |
|                                               |
 -----------------------------------------------
```

RFC2562 – a Problem

- Client cannot have NOP option enabled
- Have to coordinate with “normal” timing mark processing
- Must insure two marks aren’t sent before a timing mark response
- Increased traffic with timing marks.

RFC2562 – Another Solution

- Solution: TCP ack

```
 --------------------------------------------------
|                                                  |
| Client             TN3270           Target       |
|                     Server          SNA Host     |
|                    Timestamps                    |
|                                                  |
| <---IP Network------><---SNA Network--->         |
|                                                  |
|      request          D                          |
| ------------------------------------------->     |
|      reply            E                    |     |
| <-----------------------------------------<      |
| |    TCP ack          F                          |
|  >------------------ - - - - - - - - - >         |
|                                                  |
 --------------------------------------------------
```

- This is not a method defined as part of the RFC since RFC was written for TN3270E server and server doesn’t see a TCP ack. Most be implemented at a point where the ack can be seen.

All You Do Is ack ack

- Benefits of using TCP ack
  - Passive
  - Doesn’t matter if it’s TN3270E or TN3270
  - Doesn’t care about RESPONSES
  - Doesn’t care about Client NOP
  - No network overhead as with DR and Timing Mark
  - SSL is not a problem
  - By moving measurement out of the TN server and not relying on protocol specific actions like DR, method can be expanded to other similar applications such as rlogin.

All You Do Is ack ack

- Downside
  - Delay Ack
  - Only goes to TCP layer, doesn't include client processing
- But
  - In practice have not seen delay ack prevent ack on TN connections
  - Measurement of client processing in DR or Timing Mark methods is an inexact approximation of time to process user input and response.

Response Time Components

- IP time – time in the IP network.
- Specifically as measured in the RFC, the time it takes for a response to go from the server to the client and for the client to respond with a +/- acknowledgement (or timing mark response).
  - (Server IP stack + network + client IP stack + client + network + server IP stack).
- In modified (ack) version, the time it takes for a response to go to from the server to the client and for TCP to send an acknowledgment over the network.
  - (Server IP stack + network + client IP to TCP layer + network + server IP stack).

Response Time Components

- SNA time – time in the SNA network plus time in application
- Specifically as measured in the RFC, the time it takes for a response to go from the server through the SNA network through application and back to server.
- This could include SNA routing through EE
- This could include session managers
- This could include “end” application (CICS, TSO, IMS).
- This could include time spent in a database that an application has to access
- “SNA” time is typically application (host transit) time.
- “SNA” time and IP time are two very different components.

What IP Problems Cause Slow Response Times?

- IP fragmentation.
- Collisions at the datalink level – full duplex/half duplex misconfiguration
- Congestion, not enough bandwidth.
- Latency and small transmission windows.
- The wrong route.

What IP Problems Cause Slow Response Times?

- TOS – IP Type of Service
  - Low Delay - Telnet
- Retransmissions.
  - Lack of flow control at the network level can cause a "slow in the middle" problem. Check this out: 1000Mb may run slower than 100Mb!
- Tuning:
  - Increasing the transmission window too aggressively.
  - Decreasing retransmission timeout too aggressively, thus causing more retransmits.

What Can Cause Long "SNA" Time?

- SNA Routing
  - Wrong route,
    i.e. IP to one machine and then cross-domain to another.
- Inadequate buffers.
- RU size
- EE
  - Anything that could affect IP routing

What Can Cause Long "SNA" Time?

- Server/host issues:
  - Inadequate memory.
  - Too-slow disk speed, and contention.
  - Too-slow channel speed, and contention.
  - Excessive Workload.
  - Inadequate processors and slow processor speed.
  - Inefficient performance groups and dispatch priorities.
  - Application problem (could include database issues, inefficient code, abends and abend recovery)
  - Other applications on same system sucking up resources
  - Other LPARS robbing yours.

Fast Response Time is Most Productive (Duh)

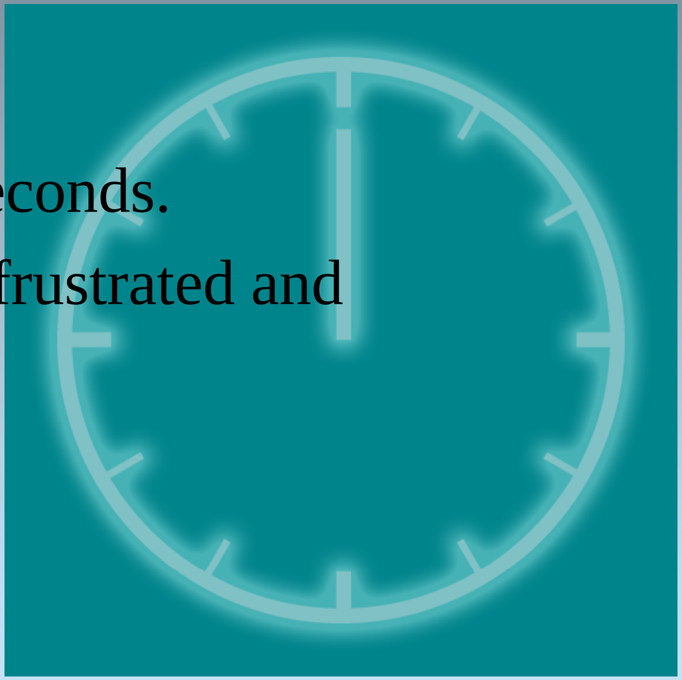
- You know this intuitively, but here's hard data:
- Early in the invention of the computer network IBM correlated response time with user productivity:
  - When they improved average response time from 2.2 secs to .8 secs per transaction, the productivity of users doubled.
  - And not just because of processing and data transfer. Long response time means users lose their train of thought. They have to "reboot" for the next transaction.
    — Thadhani 1984, Lambert 1984.

Consistent Response Times Are Just As Crucial

- This one may not be as obvious:
- “Studies have shown that user productivity increases significantly when system response time is consistent, even if on average the response time is low [meaning slow]...
- “A single large disturbance followed by high responsiveness is less disruptive to a user's productivity than many small unpredictable disturbances...”
  — Petrou, Ghormley, Anderson.
- It makes sense when you think about it

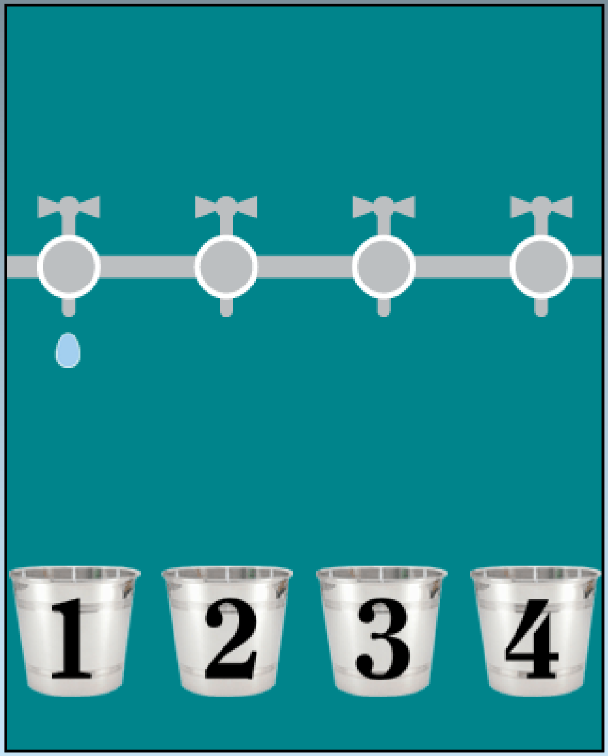
For Example:

- Target = 1 second/transaction.
- Average = 0.8 seconds/transaction.
- But 15% of the transactions take 2 seconds.
- Every 7th time I press "Enter," I get frustrated and distracted by the wait.



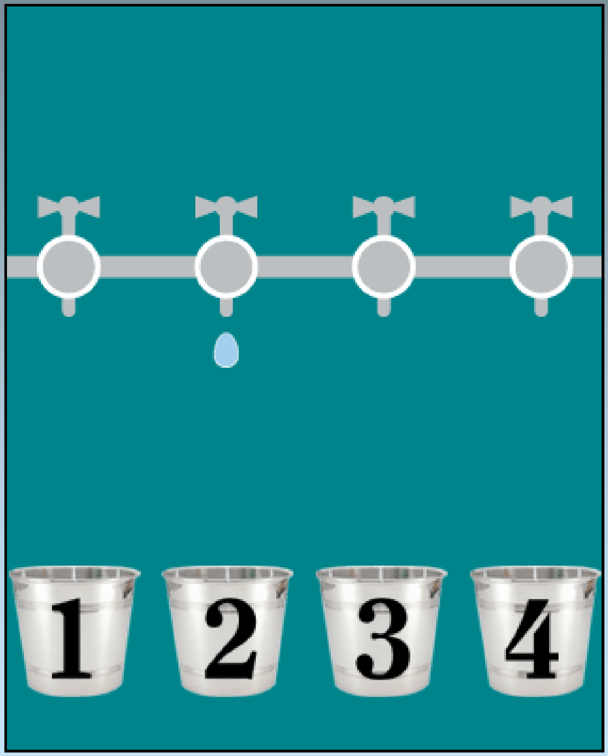
To Measure Consistency, Sort Transactions Into Response Time “Buckets”

- The TN3270E MIB, RFC 2562, tells us to sort response times for transactions into several time classes or "buckets."
- The buckets provide a simple, visual measure of consistency.
- Bucket #1 is the target, the acceptable response time.
- You typically want 90%-plus of transactions in bucket #1, regardless of your overall average response time.



To Measure Consistency, Sort Transactions Into Response Time “Buckets”

- When 15% of your transactions fall into the 2-second bucket, you have a problem, no matter how low your system's average response time.

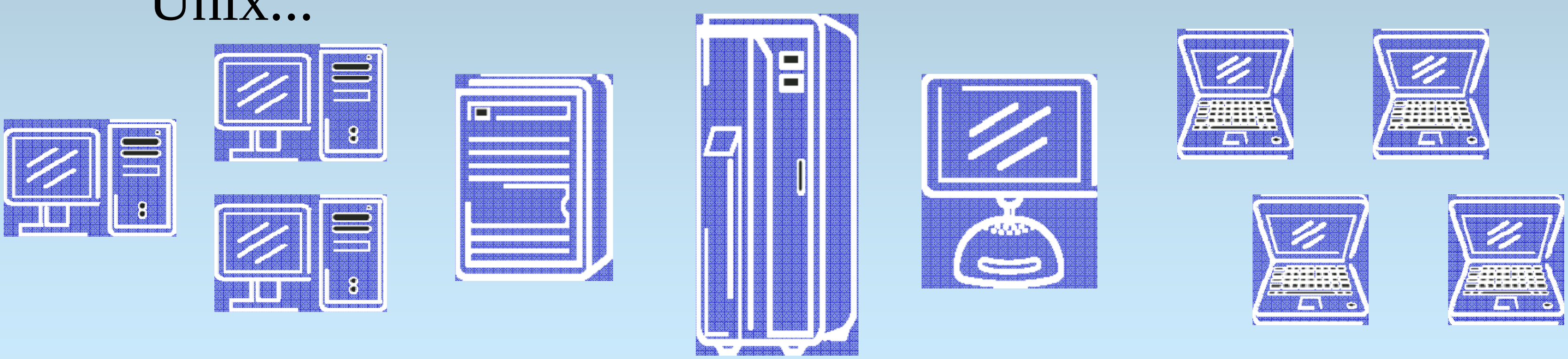


One Last Thing on RFC2562

- To Avoid “Crying Wolf,” Calculate Smoothed Averages from “Sliding Windows”
- The TN3270E MIB, RFC 2562, tells us to calculate average response times with an algorithm that variously weights data from multiple sample intervals — the “sliding windows.”
- The result is the “smoothed sliding average response time.”
- This technique emphasizes recent trends yet smoothes ephemeral spikes.
- It can recognize trouble arising in recent transactions, yet avoid generating nuisance alerts — “crying wolf.”

The HTTP (r)Evolution

- More and more, core business applications include browser interfaces.
- Critical web servers are running on z/OS or z/OS-USS platforms.
  - For instance on z/OS: Websphere, Neon Systems
  - USS: Apache Tomcat, pretty much anything that runs on Unix...



The Brave New World

- HTTP Monitoring Poses New Complications: More Connections
- One TN3270 connection can run all day. But HTTP completes only a few transactions, then closes the connection.
- In fact, HTTP v.1.0 does only one transaction per connection, never more.
- One mouse-click usually means creating several HTTP connections.
- These connections can be short lived.

Let’s Shake on It

- So the “handshake,” the time it takes to establish each connection, can be a significant part of transaction time.

File Transfer, Another HTTP Complication

- HTTP Monitoring Poses New Complications: Data Transfer
- HTTP servers combine transaction service with file transfer.
- A TN3270 transaction is generally a small matter--one request packet, one reply packet.
- For file transfers, the big jobs of sending multiple packets, we have FTP.
- HTTP works in both roles: Clients request. Servers reply. Then they transfer files.
- An HTTP server therefore also has to be monitored like a file transfer.
- And comparing times to complete file transfers isn't particularly meaningful.
- A 500 Kb message may travel a fast network and still look slow compared to a 5 Kb message on a slow network.

And There's the Old req/rsp

- As noted on the previous slide, in addition to the new component of data transfer rate, we still need to see how fast requests are turned around
- Solution: measure First Byte Response Time
- This is the time to send a request and get the first byte of response back.
- Don't care if response is completely carried in one message or runs for megs.
- We're measuring the responsiveness of the server here.

Look Familiar?

- Works similar to TN3270

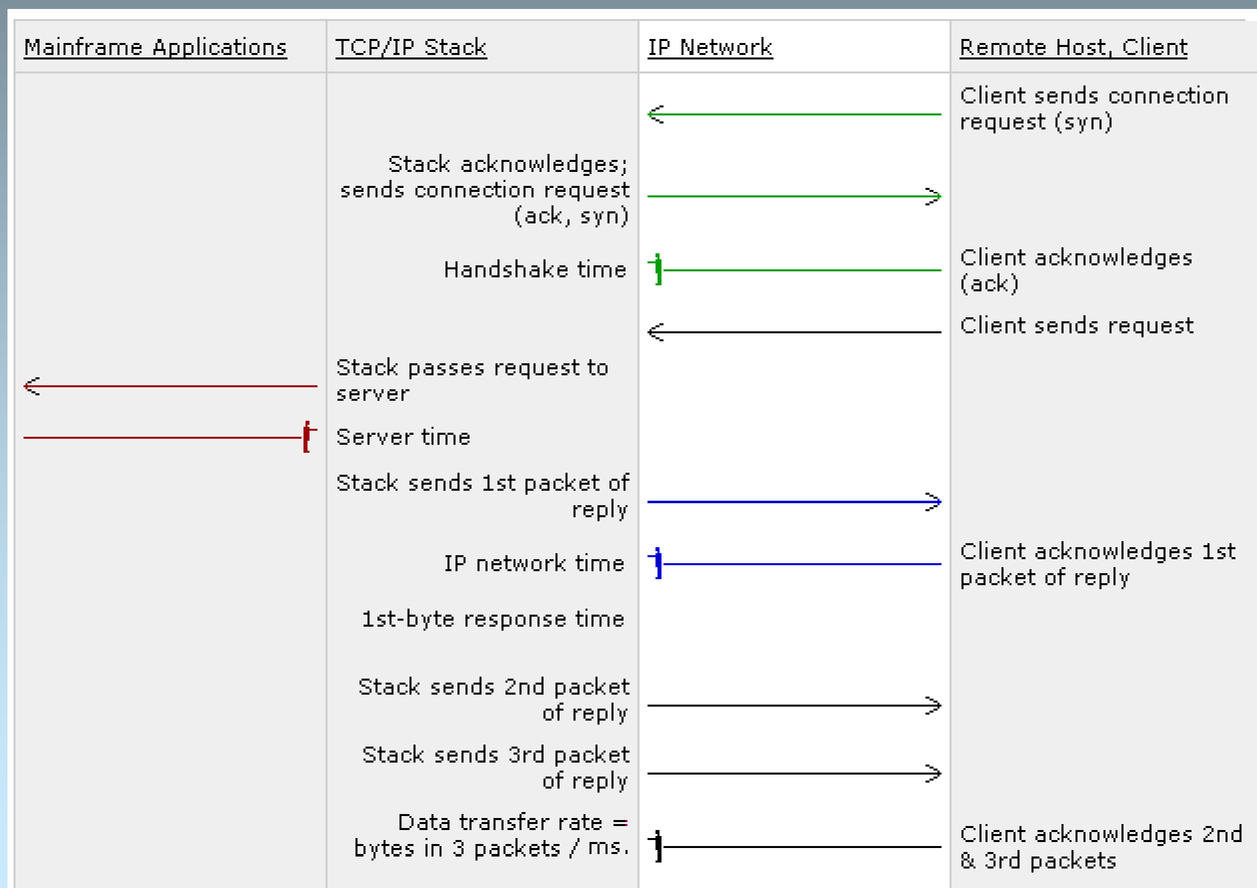
```
 ------------------------------------------------
|                                                |
| Client            HTTP            Target       |
|                   Server       Application     |
|                  Timestamps                    |
|                                                |
| <---IP Network-------><---Application--->      |
|                                                |
|      request         D                         |
| ------------------------------------------->   |
|      reply           E                    |    |
| <------------------------------------------<   |
| |    ack             F                         |
|  >---------------------- - - - - - - - - - >   |
|                                                |
  ---------------------- ------------------------------------------------
```

Then There Were Three:

- For HTTP There are three events to measure

1. Handshake time, the time it takes to establish each connection.
   - Because the ratio of transactions to handshakes is nearly 1:1 (it is 1:1 for HTTP 1.0).
2. 1st-byte response time, the HTTP equivalent of TN3270 transaction time.
   - That is, the time from the client sending its request, to receiving the first packet of the server's reply.
   - That's how we can separate server time from IP network time.
3. The data-transfer rate.
   - The number of bytes/second the server can send--same as for FTP transfers.

Join Together in the Band



Breaking It Down (Part 1)

- Handshake time delays
  - Problem in network – see earlier slide on TN3270
    - Latency
    - Wrong route
    - **Not** a direct result of fragmentation (could be indirect result)
  - Problem in stack (up to TCP layer)
    - Busy?
    - Maybe the stack isn’t getting cycles?
  - Application
    - Backlog can cause silent drops

Breaking It Down (Part 2)

- 1st Byte Response Delays
  - IP time
    - Network issues. See notes on TN3270
      - Congestion
      - Latency
      - Wrong route

Breaking It Down (Part 2)

- 1st Byte Response Delays
  - Server time
    - Application or host issues. See notes on TN3270
    - Inadequate memory.
    - Too-slow disk speed, and contention.
    - Too-slow channel speed, and contention.
    - Excessive Workload.
    - Inadequate processors and slow processor speed.
    - Inefficient performance groups and dispatch priorities.
    - Application problem (could include database issues, inefficient code, abends and abend recovery)
    - Other applications on same system sucking up resources
    - Other LPARS robbing yours.

Breaking It Down (Part 3)

- Data transfer rate
  - Network
    - Latency
    - Collisions
    - Wrong route
    - TOS
      - Low Delay, high throughput
    - Small MTU/MSS

Breaking It Down (Part 3)

- Data transfer rate
  - Network
    - Fragmentation
    - Retransmissions
    - Window size
    - Window size scaling issues
      - Proxy
      - Implementation differences
    - Increasing the transmission window too aggressively.
    - Decreasing retransmission timeout too aggressively, thus causing more retransmits.

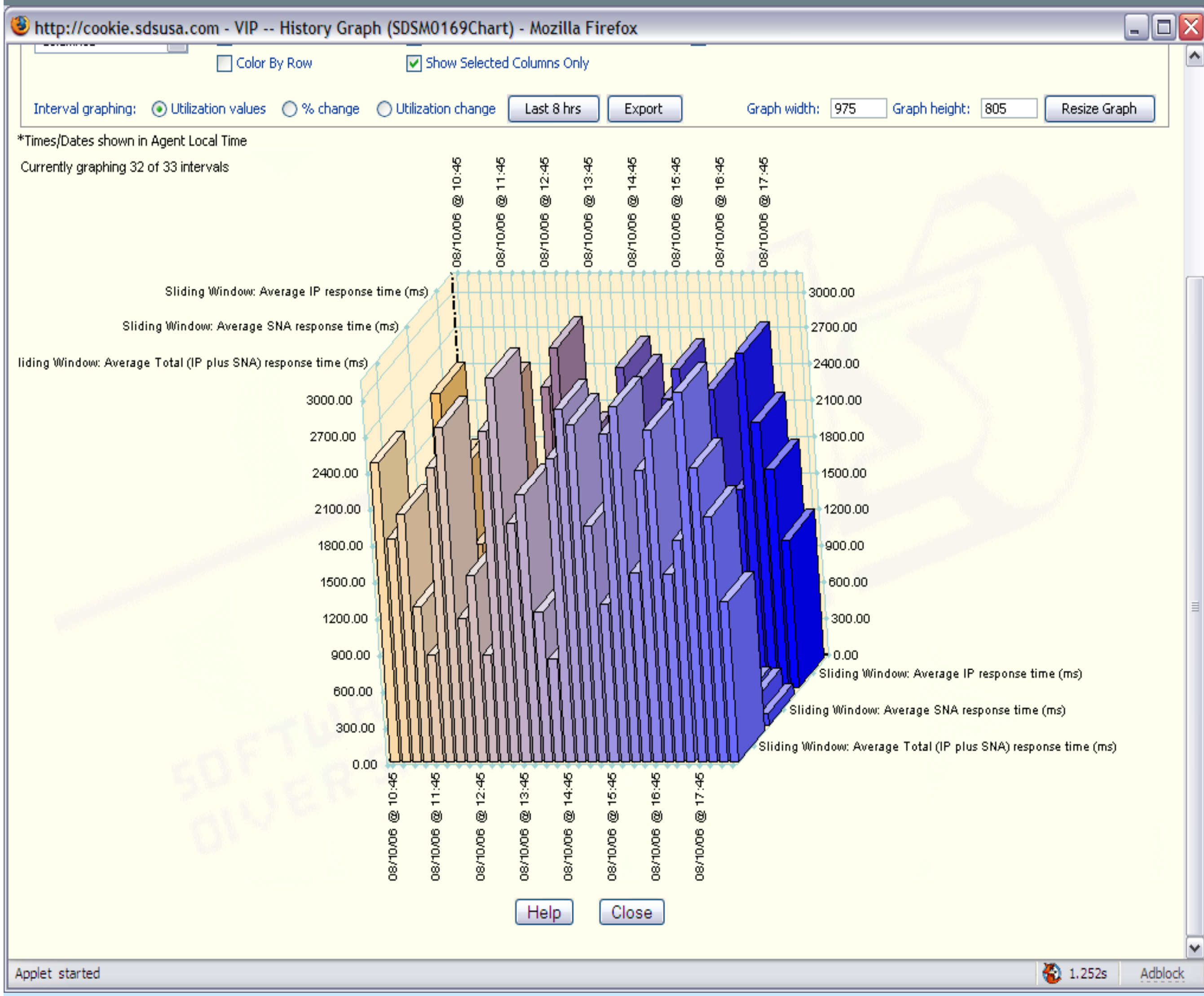
Breaking It Down (Part 3)

- Data transfer rate.
  - Application
    - Buffers (not enough, too small)
    - Blocksize
    - Database issues
    - All the stuff mentioned before on server issues

Beyond HTTP

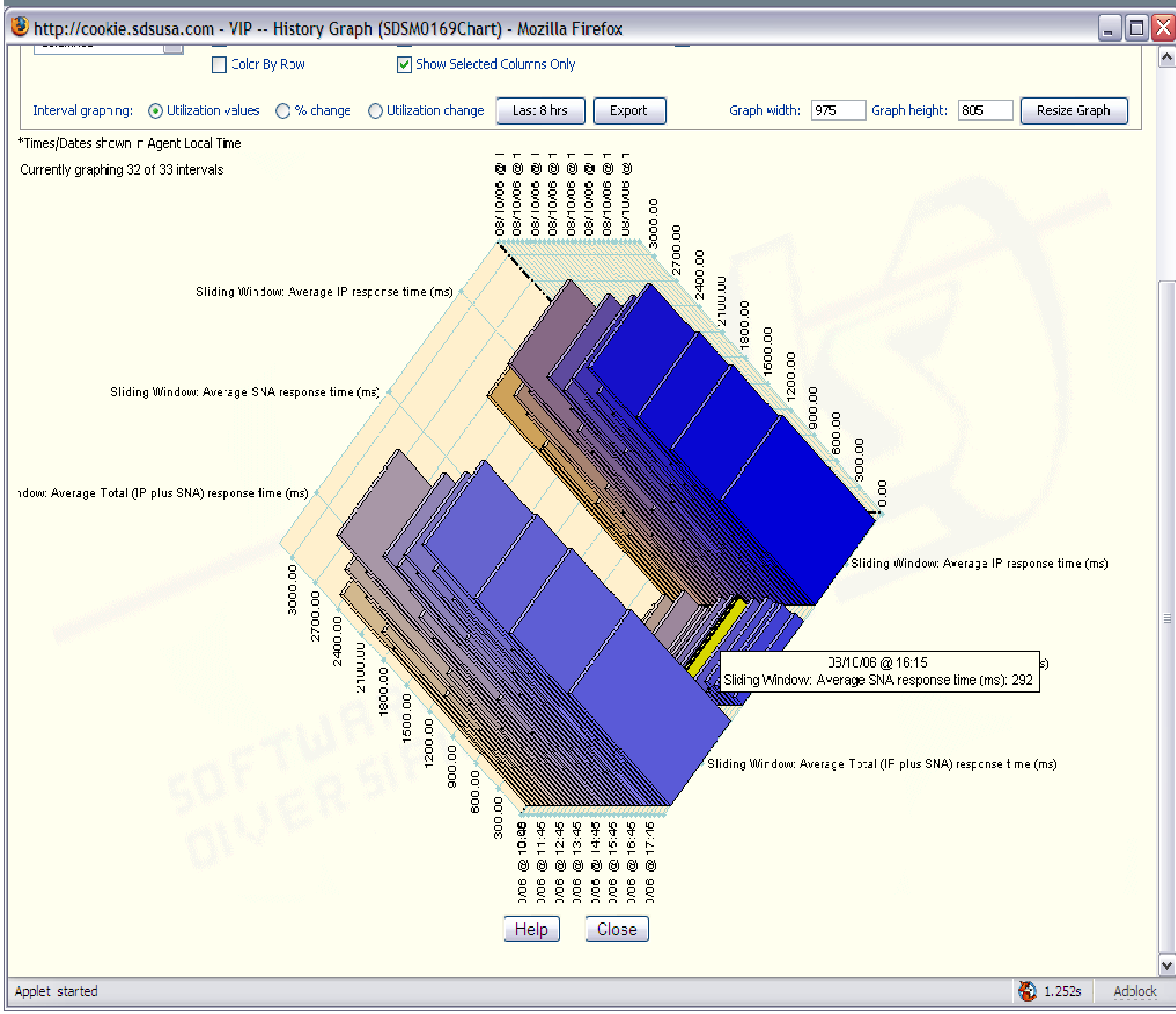
- With these three measures, any application following a req / resp format could be measured
- CICS sockets
- DB2 socket enabled backend
- Limitations
  - Detail of where delays are along the path
  - Doesn’t understand totality of user experience which could involve multiple apps on multiple servers

Example 1 – TN3270



- Total time closely follows IP time.
- IP time is spiky at fairly regular intervals.

Example 1 – TN3270 Another Viewpoint

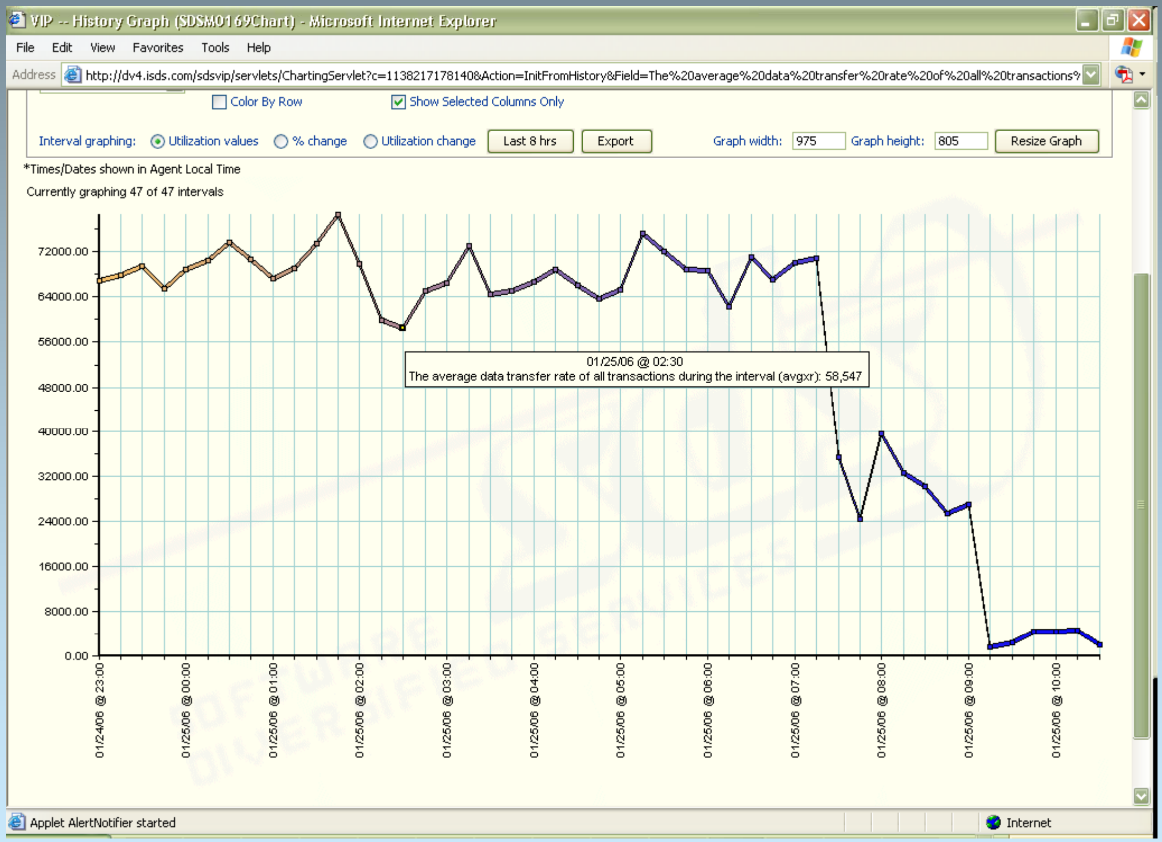


- Rotated a bit you can see that indeed SNA time is not a factor here.
- Could an external (to TN3270) factor be at play here?
- What sort of external factor might we look for?

Example 1 – TN3270 What Was the problem?

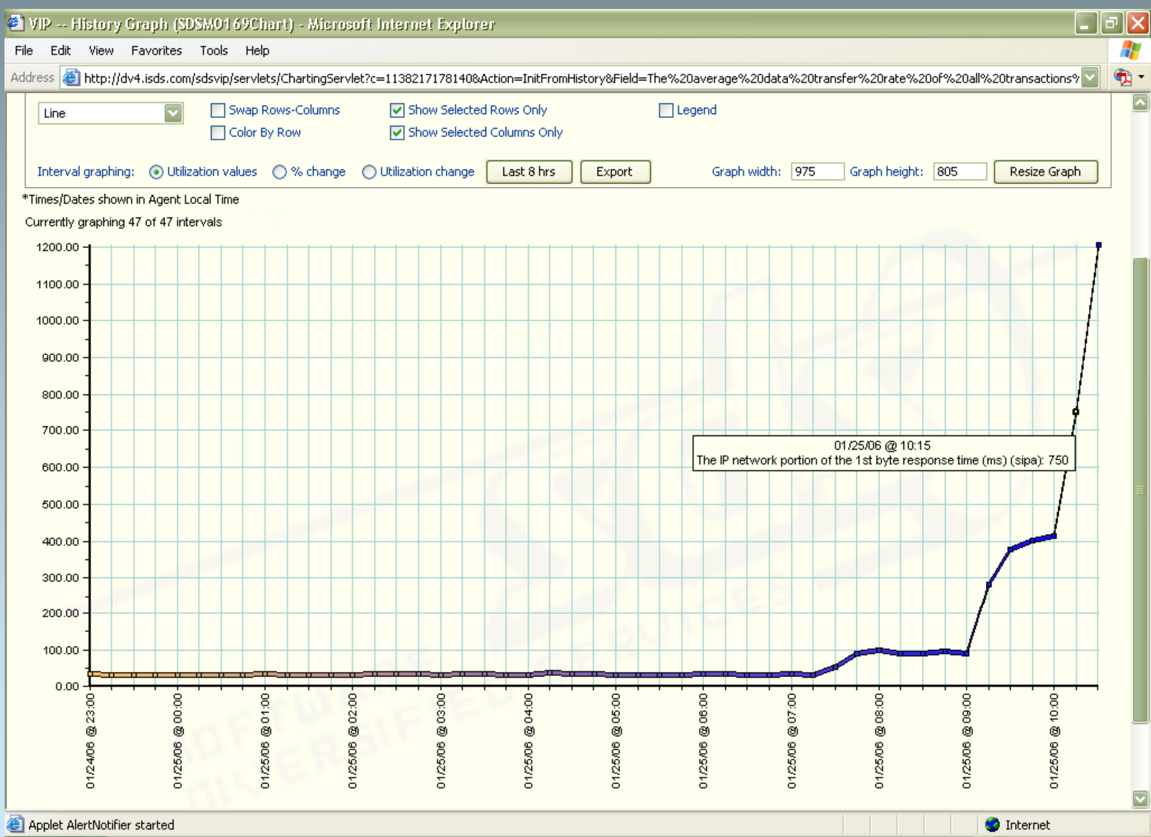
- It was indeed an external factor
- We were running TN3270 sessions across a slow segment
- TN3270 doesn’t move much data and wasn’t terribly impacted by lack of bandwidth
- However, when I ran FTPs across the same segment, TN3270 response times spiked.
- FTPs were run at regular intervals creating nice up and down in TN3270 response time.

Example 2 - HTTP



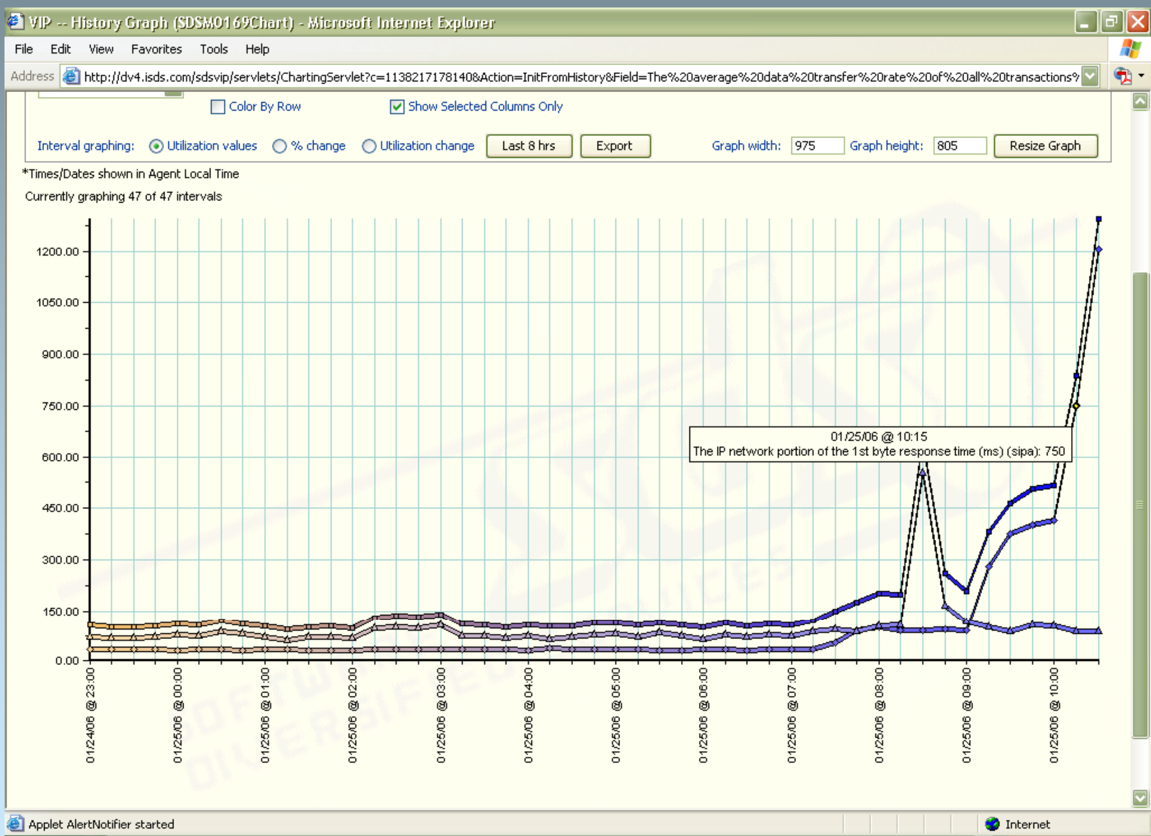
- Data transfer rate falls. That is, transfers take longer.
- Drops at 7:30 and again at 9:00.

Example 2 - HTTP



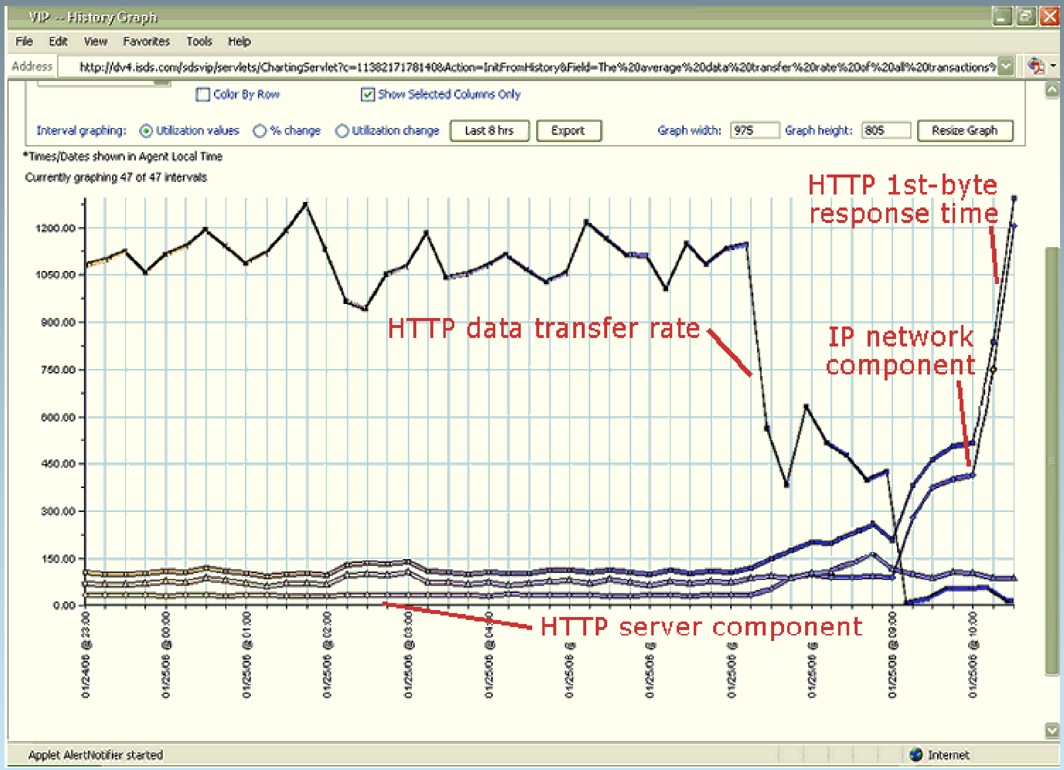
- Here is the IP portion of first byte response
- A little rise about 7:30/7:45. Levels to 9:00 and then starts up at 10 and continues up until the end of the measurement period.

Example 2 - HTTP



- Here is a chart of 1st byte response: total and the server and IP components.
- IP component is chart we just saw,
- Server component is mostly flat, though there is a spike about 8:30.
- Total pretty much follows IP component (except for spike).

Example 2 - HTTP



- Here it is altogether (with server spike removed so we can focus on main phenomenon here)
- Server time is fast and consistent.
- Data transfer rates fall.
- IP network time increases.
- Look at IP network related issues.

Example 2 – HTTP The Problem

- At 7:30 one of our guys accessed the app via VPN.
- At 9:00, another accessed via a slow connection.
- This affected the IP component of 1st byte response, and data transfer rate.
- What about that server spike at 8:30?
- That was uncontrolled (it happens), given the timeframe – morning, about the time people get it, I’m going to speculate that it was due to load on the system as people got in and started firing things up.

Sources

- BT Infonet. 2005. "ROI Constraint: The Impact of Network Underperformance on Return On Investment for Enterprise-Class Applications."
- www.bt.infonet.com/images/pdf/roi_white_paper.pdf Keohane, Jim. June 2003.
- "Response Time is in the Eye of the Mouse-Clicker." TCP/IP Manager eNews. Software Diversified Services. www.sdsusa.com/tcpipnews/june03response.htm Lambert, G. N. 1984.
- "A comparative study of system response time on program developer productivity." IBM Systems Journal, v23 #1. http://www.research.ibm.com/journal/sj/231/ibmsj2301E.pdf Petrou, David, Douglas P. Ghormley, Thomas E. Anderson. No date
- [late 1990s?].
- "Predictive State Restoration in Desktop Workstation Clusters." University of California, Berkeley. www.eecs.berkeley.edu/Pubs/TechRpts/1996/CSD-96-921.pdf Sevcik, Peter. July 1999.
- "Customers Need Performance-Based Network ROI Analysis." Business Communications Review. www.bcr.com/bcrmag/1999/07/p12.php Sevcik, Peter. July 2002.
- "Understanding How Users View Application Performance." Business Communications Review. www.bcr.com/bcrmag/2002/07/p08.php Thadhani, A. J. 1984. "Factors affecting programmer productivity during application development." IBM Systems Journal, v23 #1 http://www.research.ibm.com/journal/sj/231/ibmsj2301D.pdf White, K., R. Moore, IBM Corp. April 1999.
- "RFC 2562, Definitions of Protocol and Managed Objects for TN3270E Response Time Collection Using SMIv2 (TN3270E-RT-MIB)." ftp://ftp.rfc-editor.org/in-notes/rfc2562.txt
- Wikipedia

The End