GMSOCKS:

A Direct Socket Implementation for Myrinet

Markus Fischer
University of Mannheim, Germany
fischer@ti.uni-mannheim.de

IEEE Cluster 2001, Newport Beach, CA

Outline

Motivation - Overall Goal for GMSOCKS

GMSOCKS for Windows 2000/NT

Detours, LSP's, Winsock Direct

Implementation Details

Tuning TCP/IP for W2K

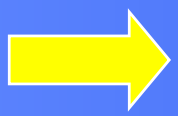
Performance Measurements

Outlook

Motivation

TCP/IP on GM = GigEth Performance = 85 MB/s, 80usec Latency

GM Raw = ~ (7usec, 247 MB/s)



How can existing, distributed applications using TCP/IP be improved ?

Goal

- Replace TCP/IP stack with thin, fast software layer, reduce overhead
  - (or: Redesign your application and use MPI/PVM or VIA to speed up your performance …)
  - Provide *ALL* TCP/IP semantics
  - Boost Performance
- Let existing applications run out of the box
  - No relinking
- Achieve Failover strategy
  - in case something goes wrong
  - External communication without SAN

Different Concepts for Implementation

Overwrite Existing Socket Functions (HPVM's Fast Messages)

- limits application's functionality (read, write to console fail, ...)

Windows World:

- Intercept Functions
- Layered Service Provider
- Winsock Direct

Ways of Implementation

Detour Package (*) allows to intercept functions

- DLL modification
- Source Function, Detour Function, Trampoline Function

Layered Service Providers (LSP)

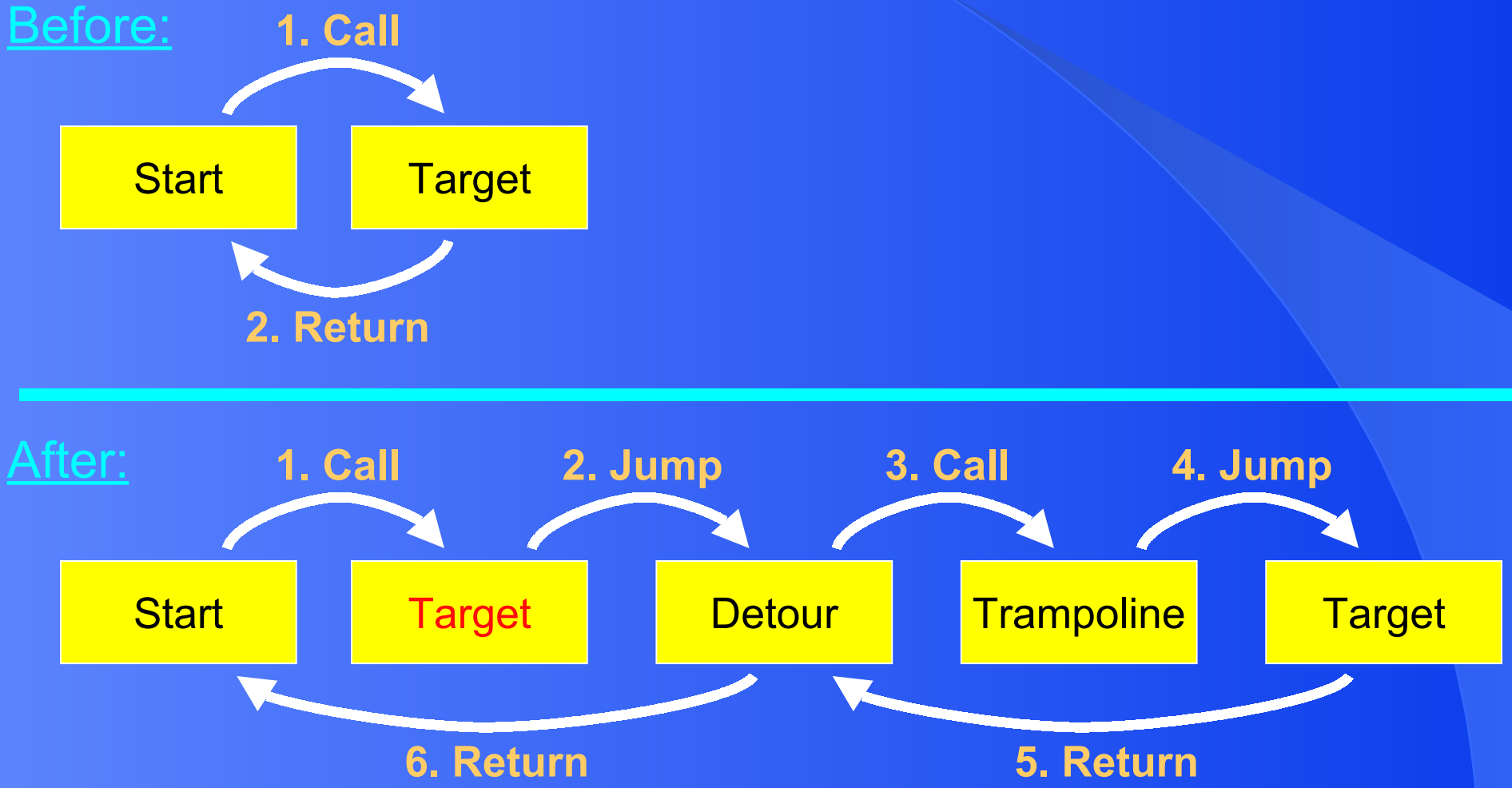
- One visible Layer to application
- Layer can call next layer below only
- Used to implement QoS

Winsock Direct

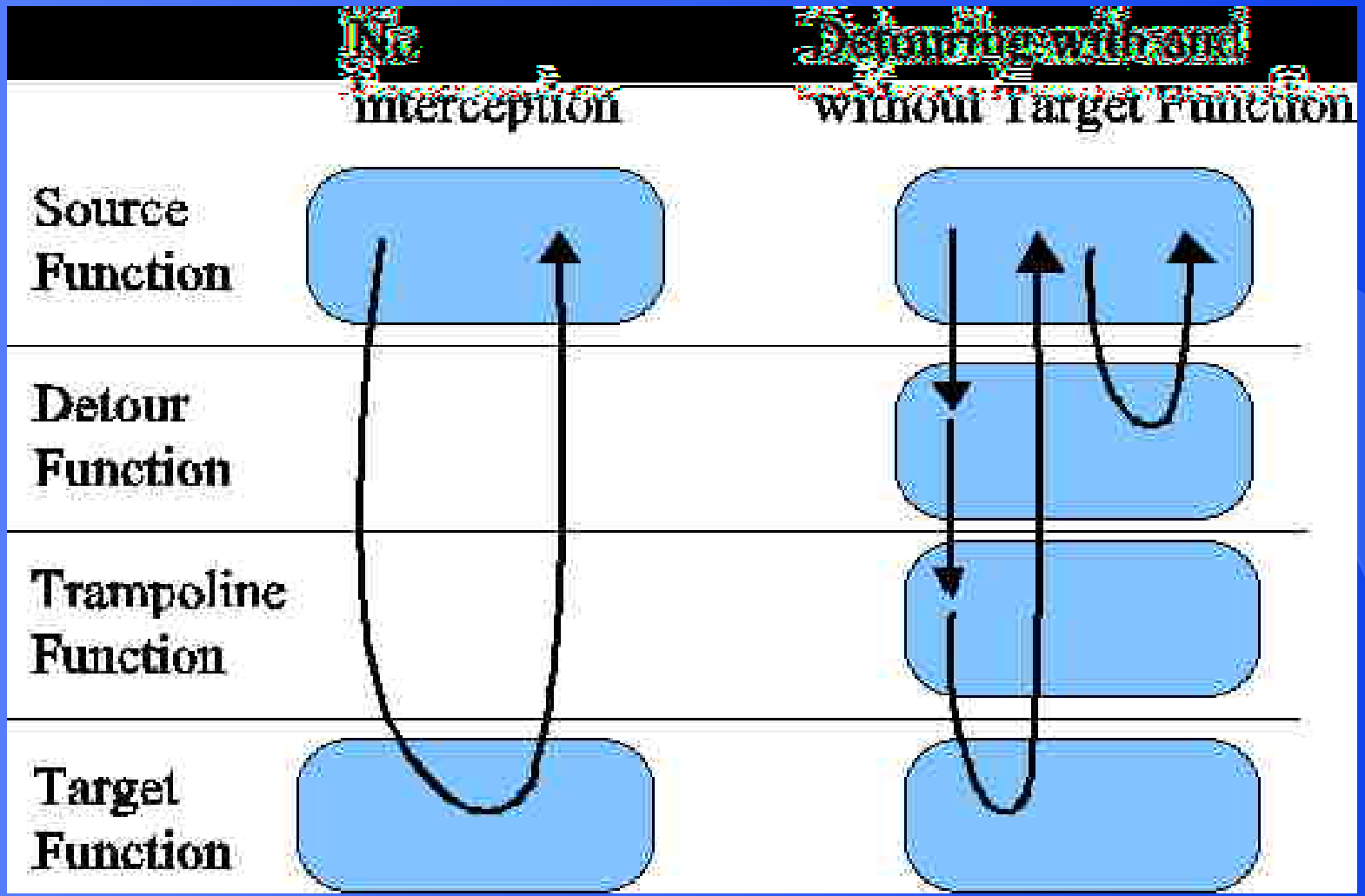
- available in Windows Advanced Server (with SP2!), Windows Datacenter Server => is an LSP !

(*) Microsoft Research Package

How Detours Works



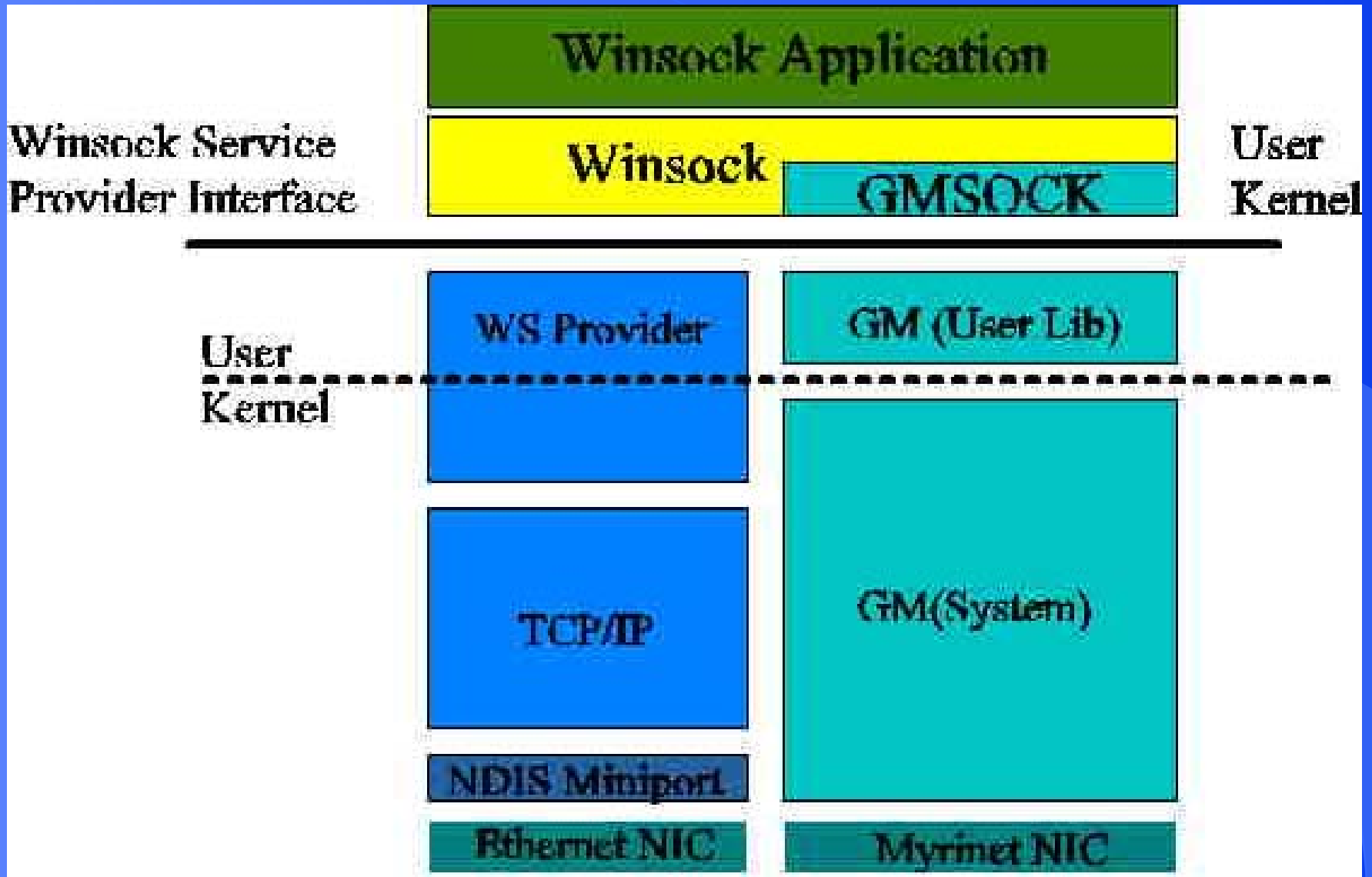
Source, Target and Trampoline Functions



GMSOCKS Detour Example

```
// Source -> Detours -> Tramp -> Target -> Detours -> Source

// Example:
// Declare Detour Function
GM-DETOUR-send(SOCKET s, ...);

DETOUR-TRAMPOLINE (int WINAPI GM-TRAMP-send(SOCKET s, ...), send);

GM-DETOUR-send(SOCKET s, void *buf, int len, int flags)
{
     if (have-gm-connection-with(s)
            use-gm-send()
     else
           GM-TRAMP-send(s, ...);
}

int DllMain(DWORD dwReason)
{   if (dwReason == ATTACH)
     DetourFunctionWithTrampoline((PBYTE)GM-TRAMP-send, (PBYTE)GM-
     DETOUR-send);
}
```

GMSOCKS using Detours



GMSOCKS as LSP

Vendor can provide a library that can be hooked into the WinSock2 DLL dynamically at run time

- Allows for accessing services from one or more transport protocols simultaneously.

Offers registration of new transport protocols in a protocol catalog

- any protocol stack registered is said to be a WinSock “service provider”

Layered Protocols are only used through the Service Provider Interface (SPI)

Install a Transport Service Provider

- Winsock2 architecture divides the Winsock subsystem into two general layers, the DLL providing the Winsock API and a series of service providers which plug in underneath via the SPI

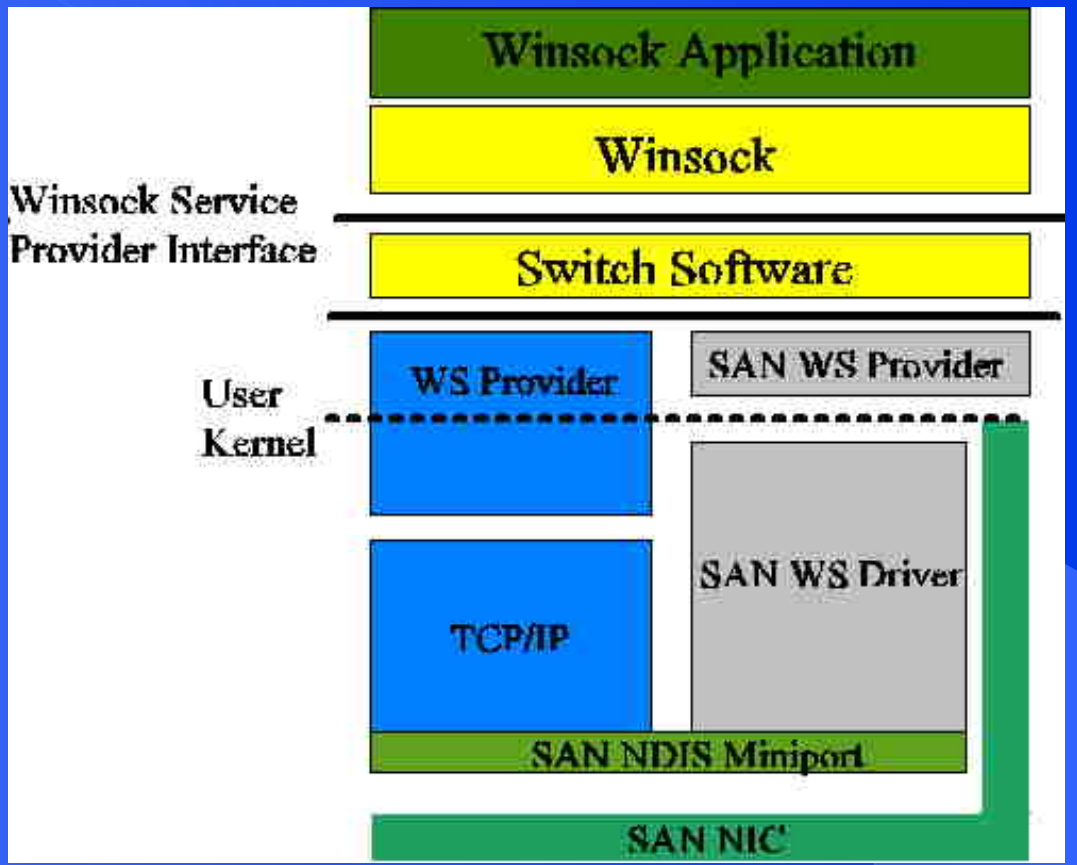
“layered protocol” relies on a base protocol for services

SPI provides an abstraction layer

- own communications transports/media can be provided

Winsock Direct

- Winsock Direct is an LSP which abstracts from different SANs.
- The counterpart, which is called by Winsock Direct has to be implemented (registration of data, message exchanges)



=> inserts an additional overhead, a fresh development of an LSP should be more efficient, since it can be optimized for a particular SAN

Winsock 1 Functions

```
// Winsock 1
int WINAPI PASCAL FAR WSAStartup(WORD wVersionRequired, LPWSADATA
    lpWSAData);
int WINAPI PASCAL FAR WSACleanup(void);
int WINAPI PASCAL FAR WSAGetLastError(void);
SOCKET WINAPI socket (int af, int type,int protocol);
SOCKET WINAPI accept (SOCKET s, struct sockaddr FAR *addr, int FAR *addrlen);
int WINAPI bind (SOCKET s, const struct sockaddr FAR *name,int namelen);
int WINAPI closesocket (SOCKET s);
int WINAPI connect (SOCKET s, const struct sockaddr FAR *name, int namelen);
int WINAPI send (SOCKET s, const char FAR *buf, int len, int flags);
int WINAPI recv (SOCKET s, const char FAR *buf, int len, int flags);
int WINAPI select (int nfds, fd_set FAR *readfds,fd_set FAR *writefds,fd_set FAR
    *exceptfds,const struct timeval FAR *timeout);
int WINAPI shutdown(SOCKET s, int how);
```

Winsock 2 Functions (1/3)

```
//////////////// WINSOCK 2 Functions
bool WINAPI AcceptEx (SOCKET sListenSocket, SOCKET sAcceptSocket,
                      PVOID lpOutputBuffer, DWORD dwReceiveDataLength,
                      DWORD dwLocalAddressLength,
                      DWORD dwRemoteAddressLength,
                      LPDWORD lpdwBytesReceived,
                      LPOVERLAPPED lpOverlapped );
int WINAPI WSAConnect (SOCKET s, const struct sockaddr FAR * name,
                       int namelen, LPWSABUF lpCallerData,
                       LPWSABUF lpCalleeData, LPQOS lpSQOS, LPQOS lpGQOS
                       );
SOCKET WINAPI WSAAccept (SOCKET s, struct sockaddr FAR * addr,
                         LPINT addrlen, LPCONDITIONPROC lpfnCondition,
                         DWORD dwCallbackData );
```

Winsock 2 Functions (2/3)

int WINAPI **WSADuplicateSocket** (SOCKET s, DWORD dwProcessId, LPWSAPROTOCOL_INFO lpProtocolInfo);

int WINAPI **WSARecv** (SOCKET s, LPWSABUF lpBuffers, DWORD dwBufferCount,
LPDWORD lpNumberOfBytesRecvd,
LPDWORD lpFlags,
LPWSAOVERLAPPED lpOverlapped,
LPWSAOVERLAPPED_COMPLETION_ROUTINE
lpCompletionROUTINE);

int WINAPI **WSASend** (SOCKET s, LPWSABUF lpBuffers, DWORD dwBufferCount,
LPDWORD lpNumberOfBytesSent, DWORD dwFlags,
LPWSAOVERLAPPED lpOverlapped,
LPWSAOVERLAPPED_COMPLETION_ROUTINE
lpCompletionROUTINE);

SOCKET WINAPI **WSASocket** (int af, int type,int protocol,
LPWSAPROTOCOL_INFO lpProtocolInfo,
GROUP g, DWORD dwFlags);

Winsock 2 Functions (3/3)

bool WINAPI **TransmitFile** (SOCKET hSocket, HANDLE hFile, DWORD nNumberOfBytesToWrite,
DWORD nNumberOfBytesPerSend,
LPOVERLAPPED lpOverlapped,
LPTRANSMIT_FILE_BUFFERS lpTransmitBuffers,
DWORD dwFlags);

bool WINAPI **WSAGetOverlappedResult** (SOCKET s, LPWSAOVERLAPPED lpOverlapped,
LPDWORD lpcbTransfer, BOOL fWait, LPDWORD lpdwFlags);

bool WINAPI **ReadFile** (HANDLE hFile, LPVOID lpBuffer,
DWORD nNumberOfBytesToRead,
LPDWORD lpNumberOfBytesRead,
LPOVERLAPPED lpOverlapped);

bool WINAPI **WriteFile** (HANDLE hFile, LPCVOID lpBuffer,
DWORD nNumberOfBytesToWrite,
LPDWORD lpNumberOfBytesWritten,
LPOVERLAPPED lpOverlapped);

bool WINAPI **ResetEvent** (HANDLE hEvent);

//////////////// END OF WINSOCK 2 Function Declaration

Implementation Details / Winsock 1

On a pt2pt connection establish companion socket with GM

Set up information to distinguish between socket ports on the same host (sender node id), add header to payload / message

header = identify src, identify special tags (control for EOF, shutdown, close…)

Start Thread to synch incoming messages

carefully look at Winsock specification

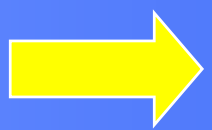
- Socket modes (blocking, non-blocking)
- send -> returning out of send means that buffer can be re-used
- recv -> performed on single descriptor, gm_receive picks next message in queue

Implementation Details

Select
- check for receive queue

closesocket (SOCKET s)
- **notify** partner about closing

shutdown (SOCKET s, int how)
- **notify** partner about RECV shutdown



notify = send encoded message via GM

Implementation Details / Winsock 2

Extend Winsock 1 Functionality By Overlapped Handling

- Overlapping: Post Functions, Query Status later
  - WSARecv / WSASend / WSAConnect / WSAAccept/ ...
- Modify Worker Thread to dispatch messages into overlapped structure
- Let Worker Thread mimic OS features (such as Event setting)

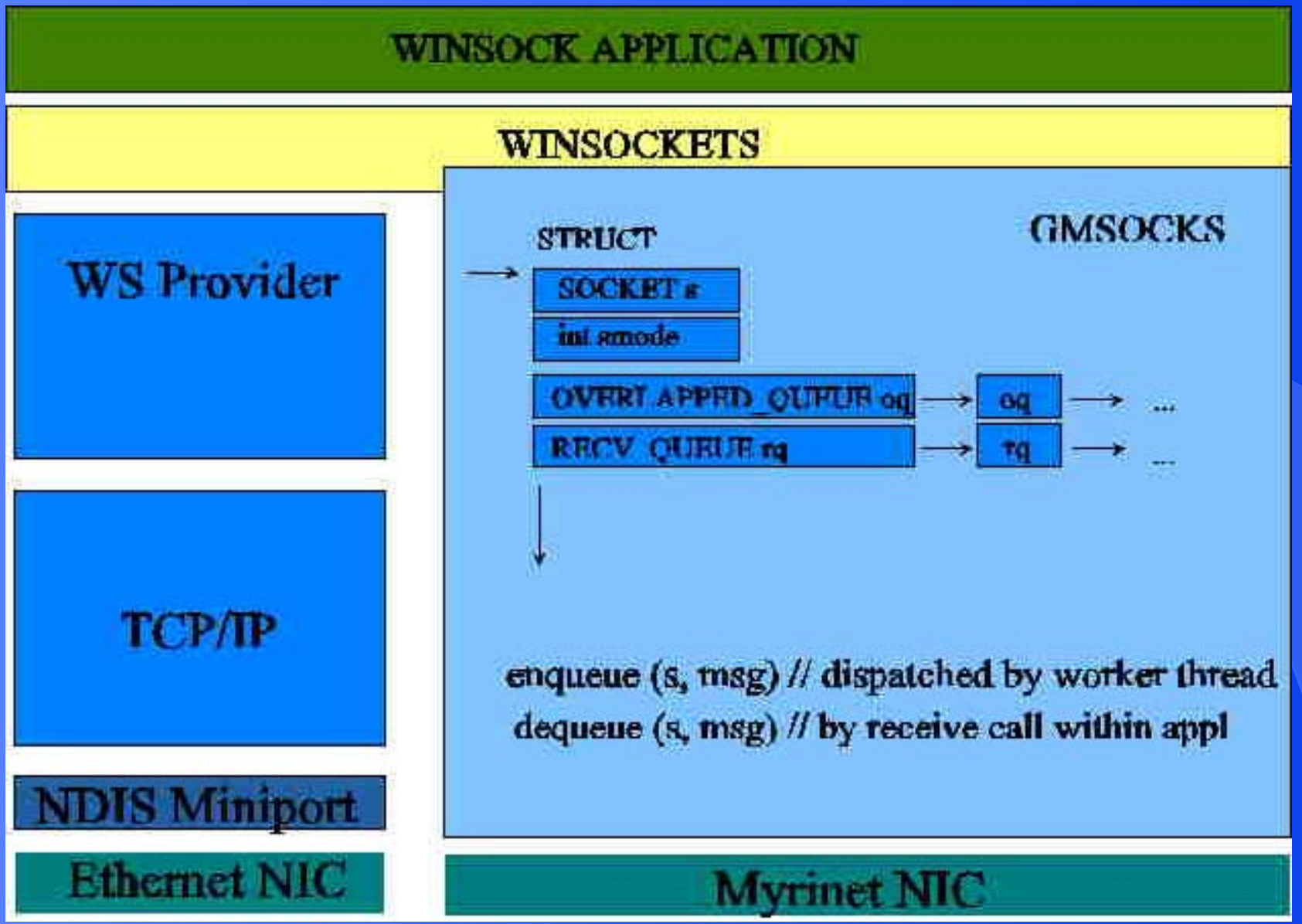
Winsock 2 Overlapped Example

```
if ((sock = WSASocket(AF_INET, SOCK_STREAM, 0, NULL, 0,
WSA_FLAG_OVERLAPPED))  == INVALID_SOCKET) {
        // HANDLE ERROR
        return 0;
}
// ACCEPT / CONNECT
olapEvent = CreateEvent (NULL, FALSE, FALSE, NULL);

recvBuffers = (LPWSABUF) malloc(sizeof(WSABUF) * numBufs);
for(i=0; i < numBufs; i++) {
        recvBuffers[i].len = pktSize;
        recvBuffers[i].buf = (char *) malloc(pktSize);
}
```

Winsock 2 Overlapped Example (cont'd)

```
recvBytes = 0;              flags = 0;
if (WSARecv(new_sock, recvBuffers, numBufs, &recvBytes, &flags, &olapStruct,
NULL) == SOCKET_ERROR) {
        if ((retErr = WSAGetLastError()) == WSA_IO_PENDING) {
                        olapFlags = 0;
                        if (WSAGetOverlappedResult(new_sock, &olapStruct,
&xferBytes, TRUE, &olapFlags)) {
                                /* data received thru overlapped i/o */
                                ResetEvent(olapEvent);
                                if (xferBytes != (DWORD) pktSize*numBufs)
                        }
                } else { // other error than WSA_IO_PENDING
                        /* free receive buffers, ERROR Handling */
                        exit(0);
                }
}
```

GMSOCK Data Structures



TCP/IP Tuning for W2K

Shareware ? : EasyMTU, TuneMTU, …

- poor performance, even: there goes your TCP/IP functionality

W2K has tuning build in

- Bottom Line:You can't do better

Registry Keys under Windows

- TcpWindowSize
- MTU
- ...

Benchmarks do have options to tune TCP/IP (setsockopt), however existing applications don't

oh, yes, the change of a registry value requires a REBOOT ...

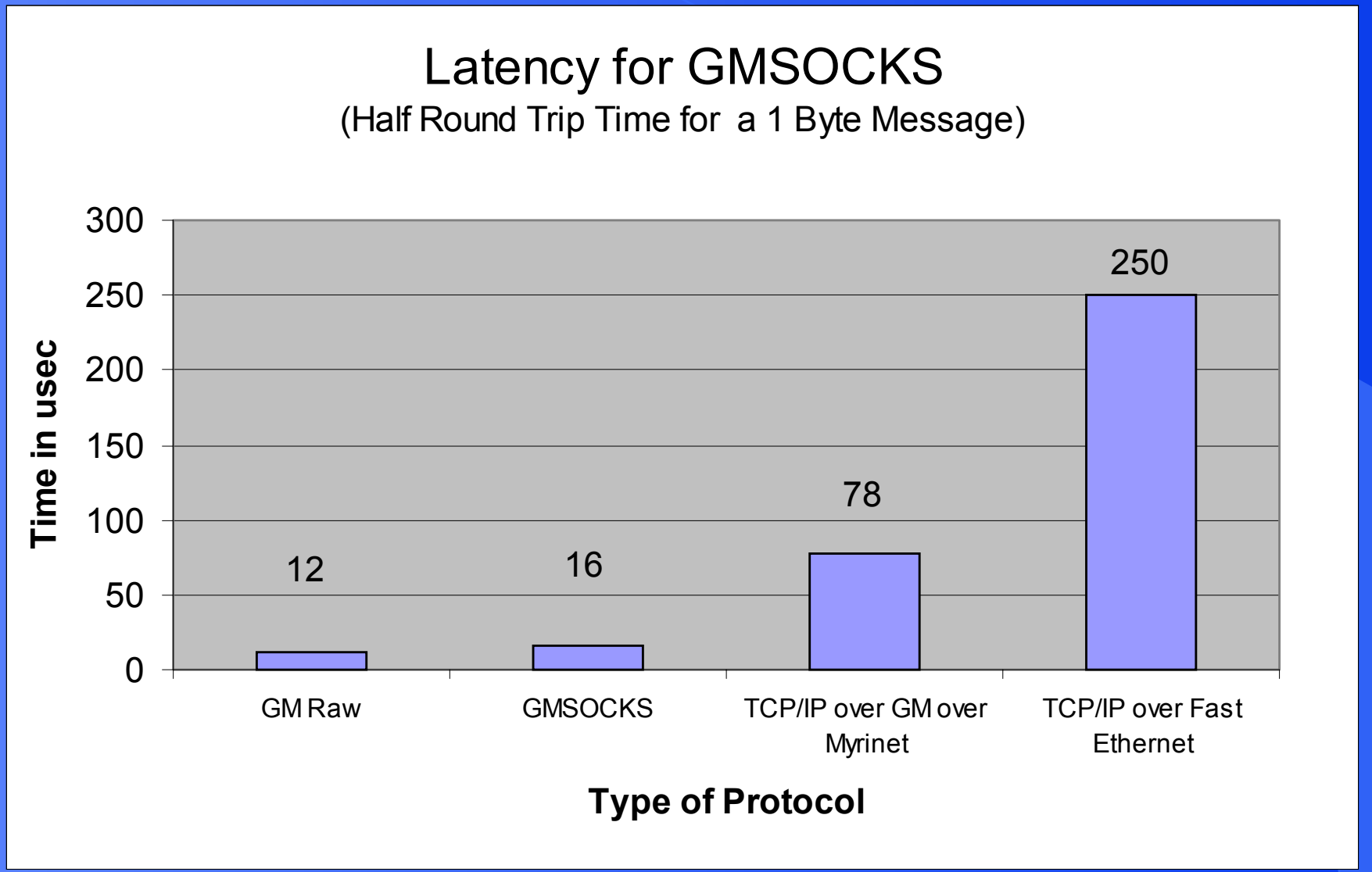
PCI 64 Bit / 66 Mhz Measurements

Myrinet 2000, Windows 2000

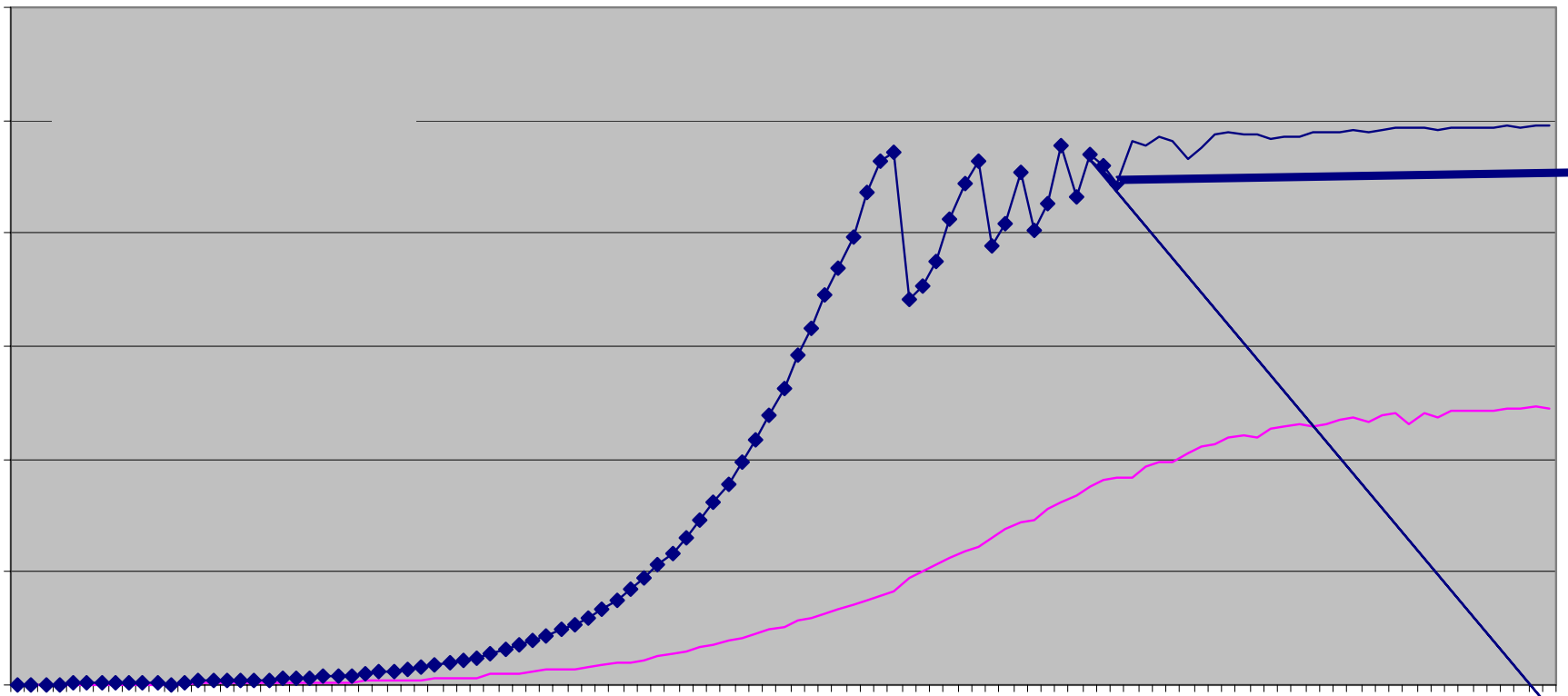
Supermicro 370DLE, PIII 1 Ghz

(455 MB/s bus read, 512 MB/s bus write)

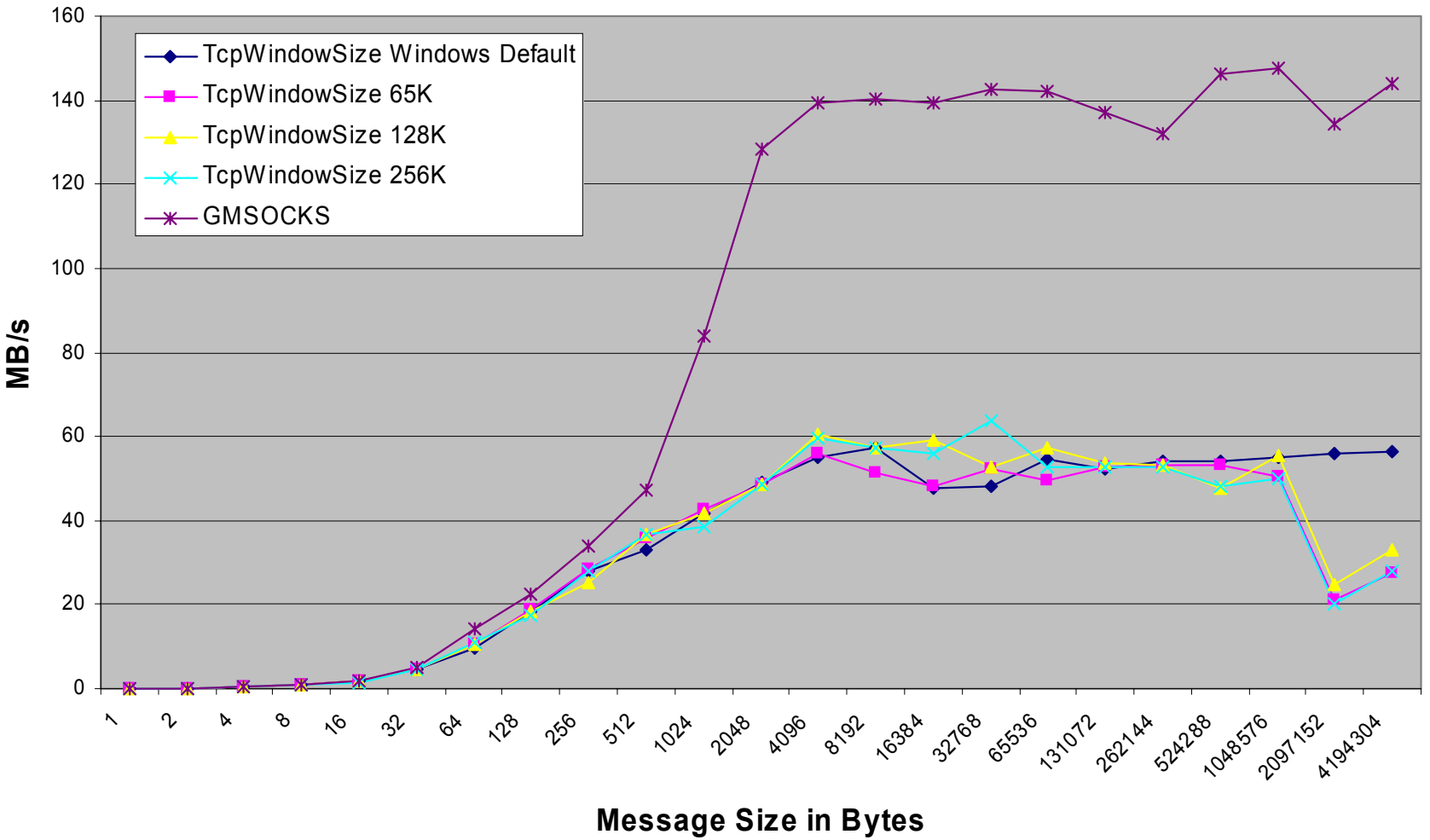
Latency Comparison



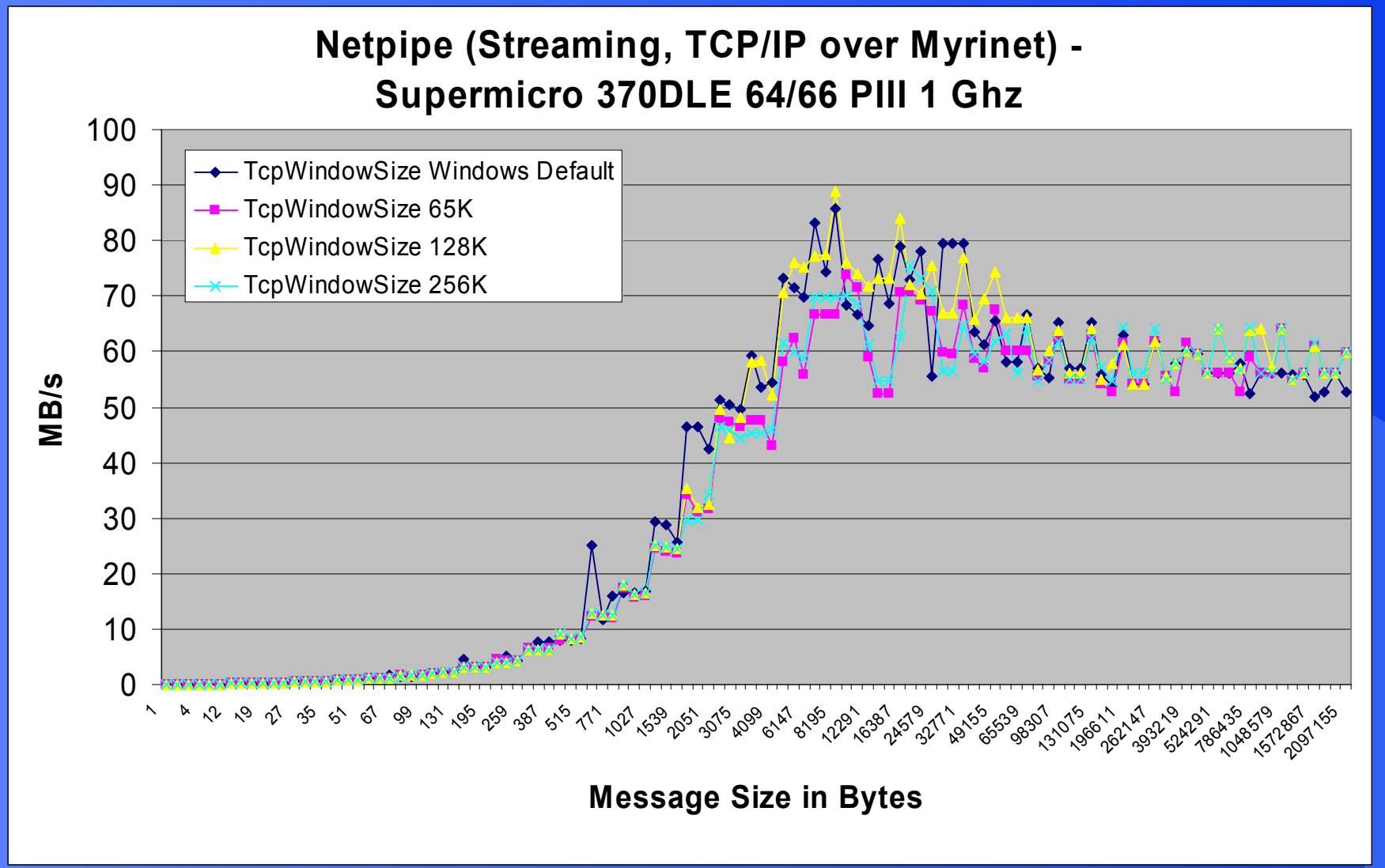
GM Raw API



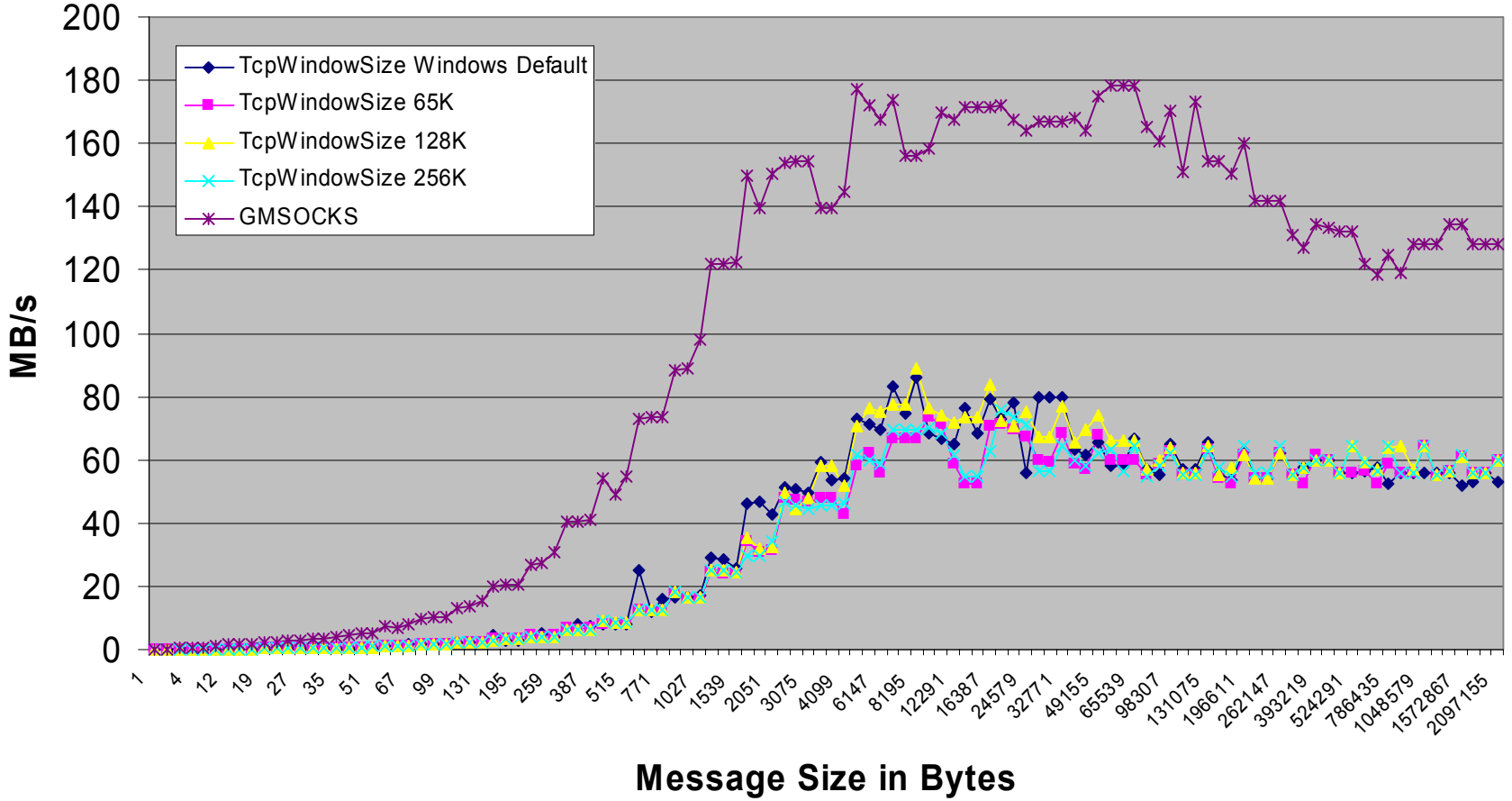
Netperf with TCP/IP over Myrinet VS GMSOCKS



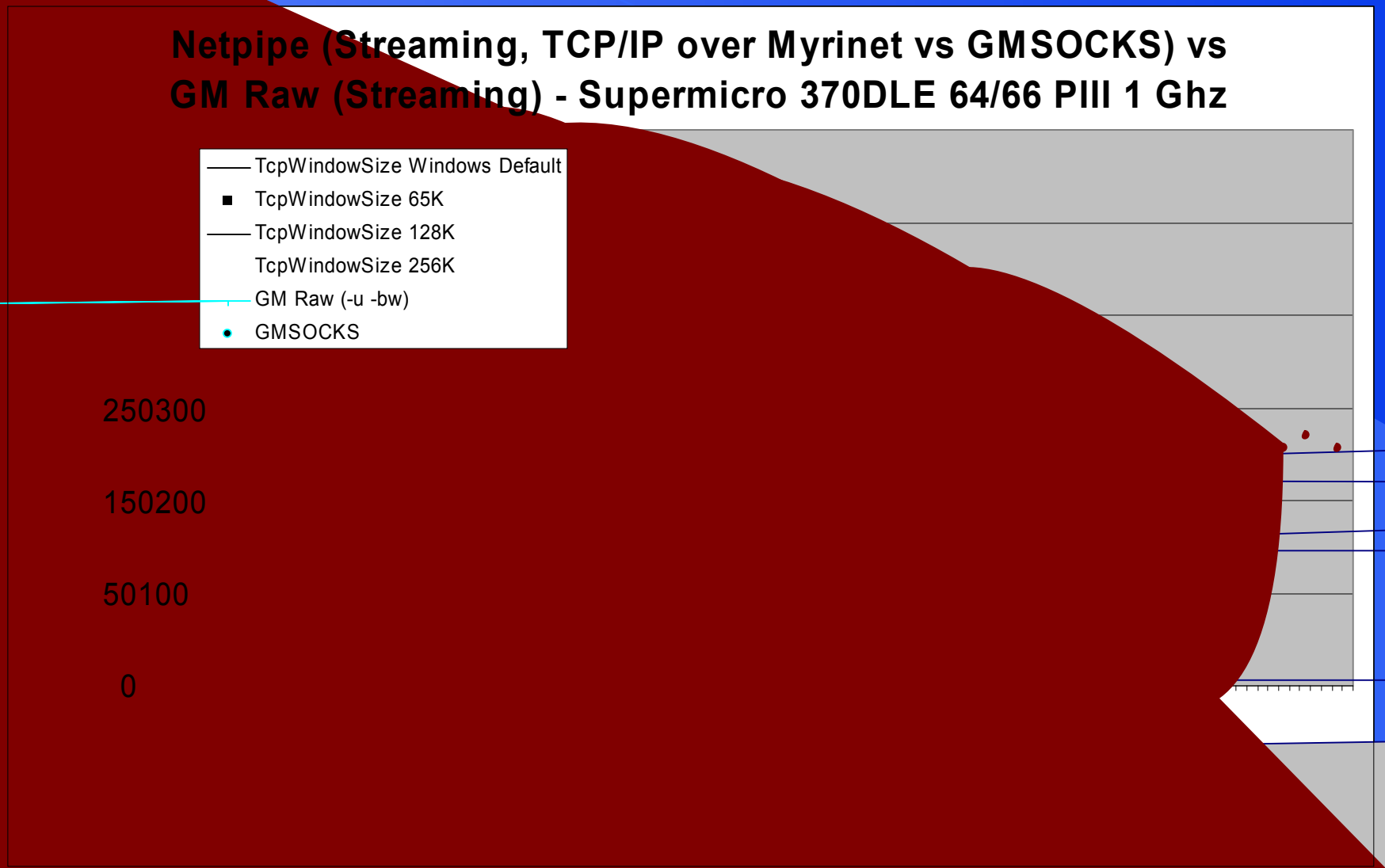
Netpipe (Streaming)



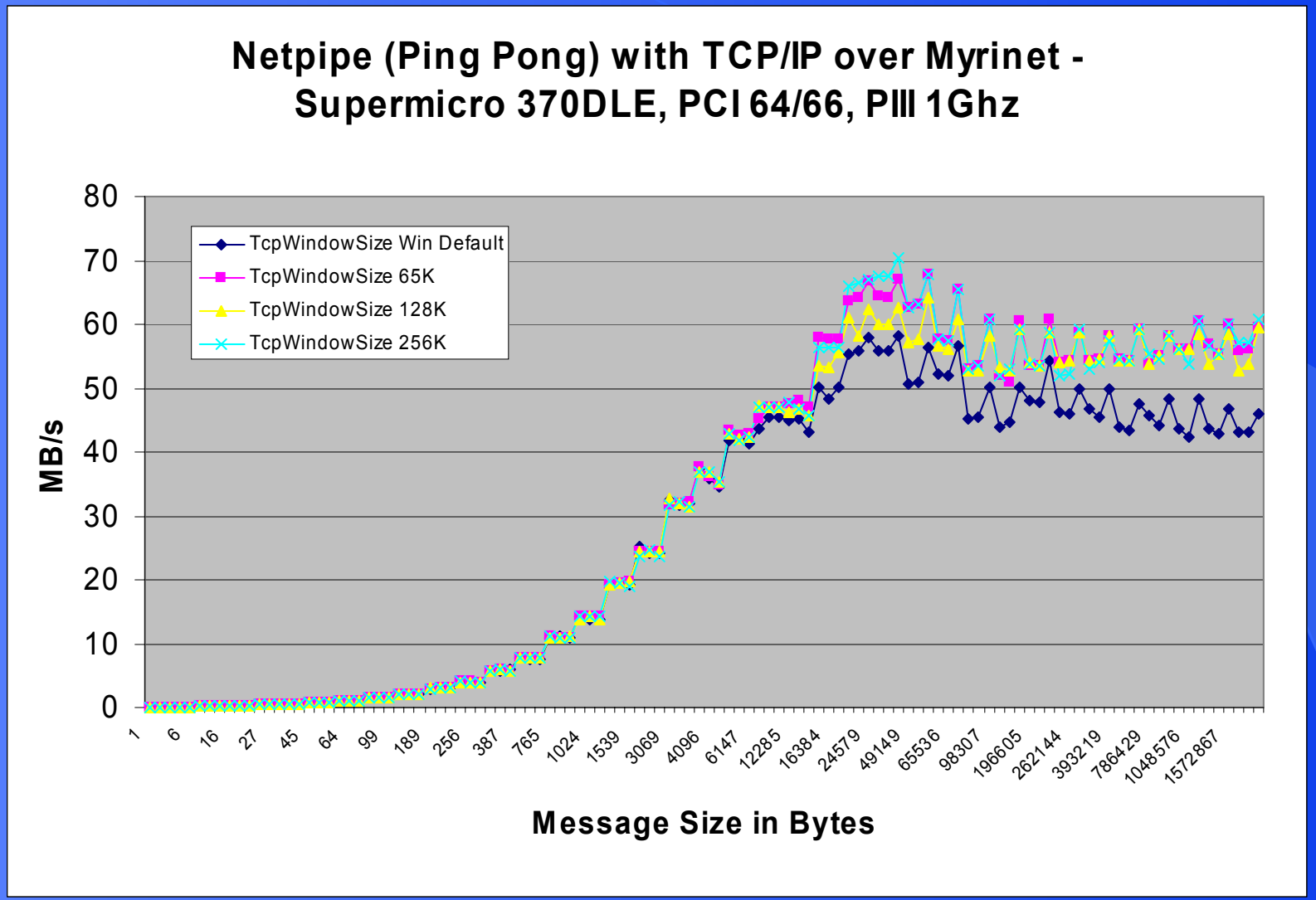
Netpipe (Streaming)



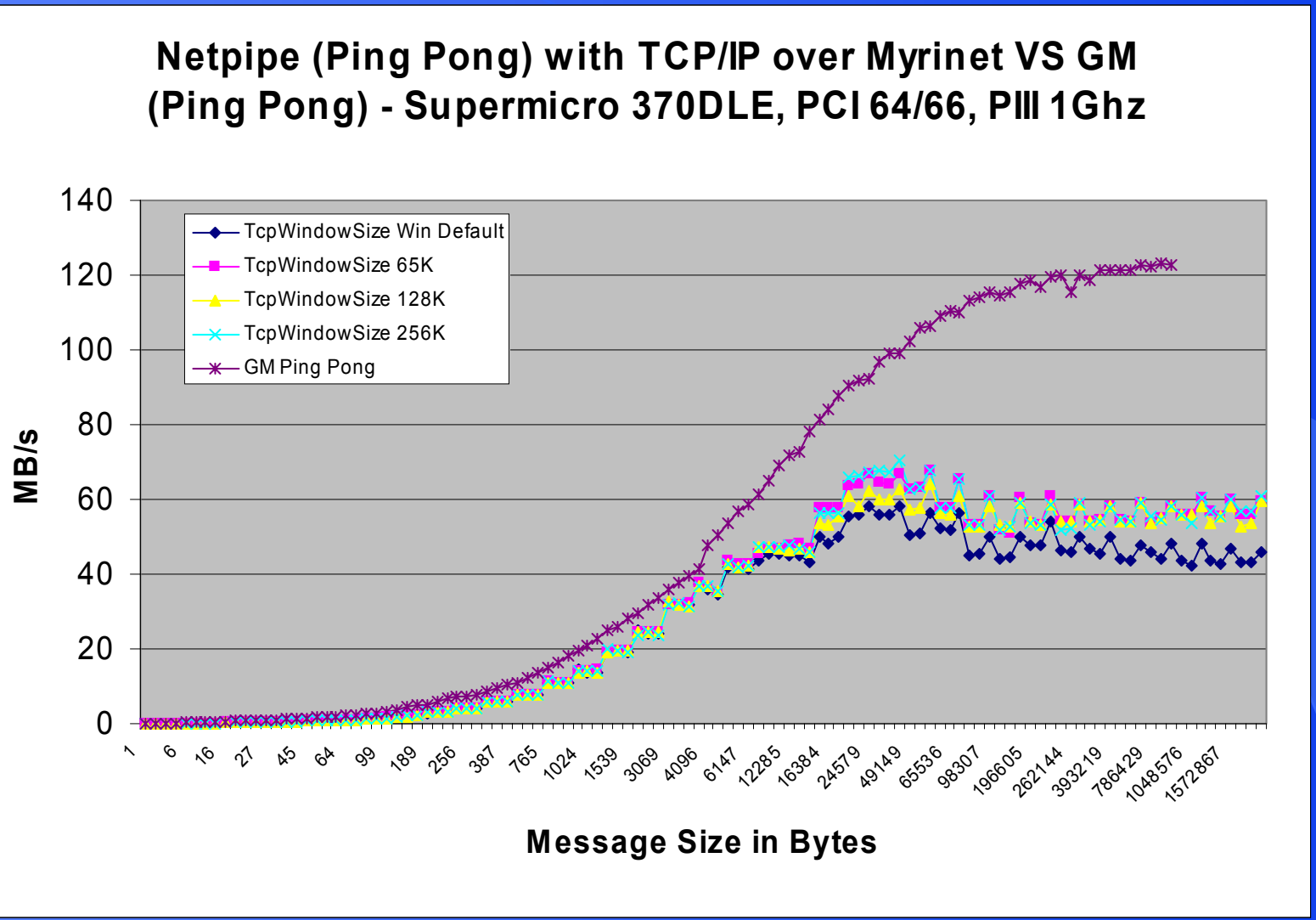
tpipe (Streaming) vs GM Raw(Streaming)



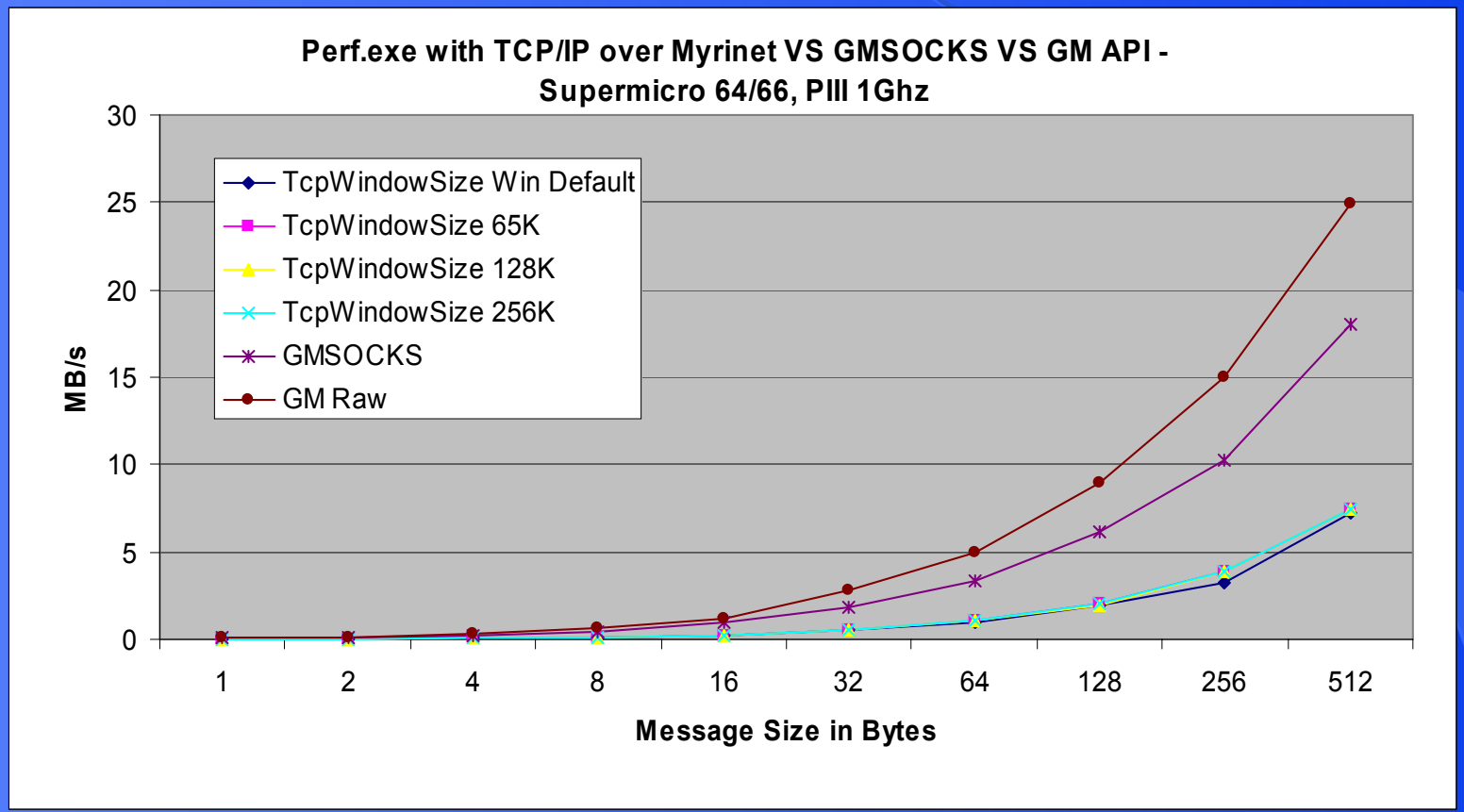
Netpipe (Ping Pong, TCP/IP over Myrinet)



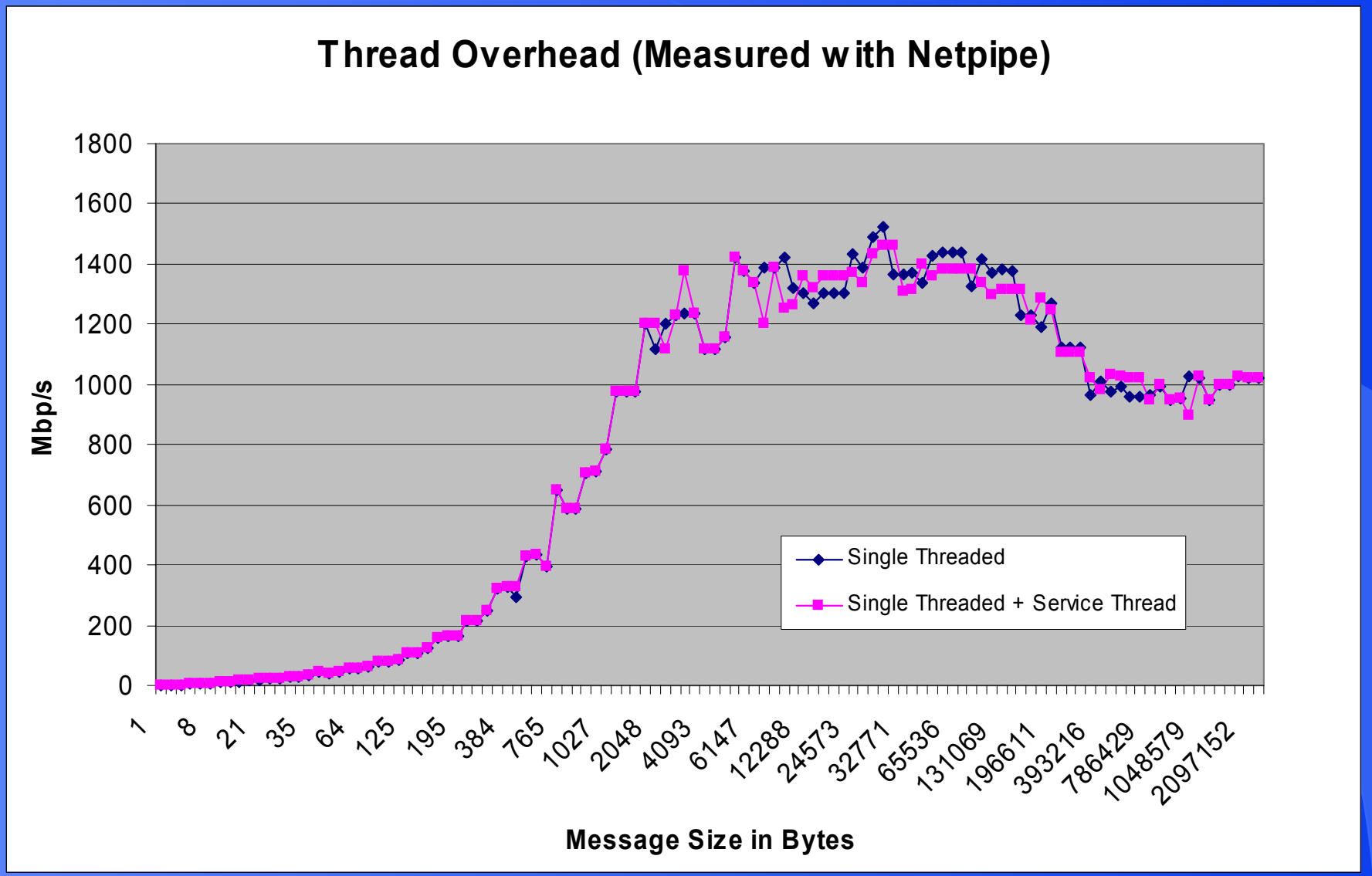
Netpipe (Ping Pong, TCP/IP over Myrinet VS GM)



Perf.exe (small messages) with TCP/IP over Myrinet vs GMSOCKS vs GM API



Thread Overhead (Single CPU System)



Conclusion & Outlook

GMSOCKS running (well) with Detours Package

Tuning for larger messages

Binary Compatibility !

LSP, Winsock Direct (work in progress)

Layering Techinque for Linux/Solaris

- Testing phase / including fork

Some more applications (ftp, ftpd)

Start looking for a commercial application

- Databases, …