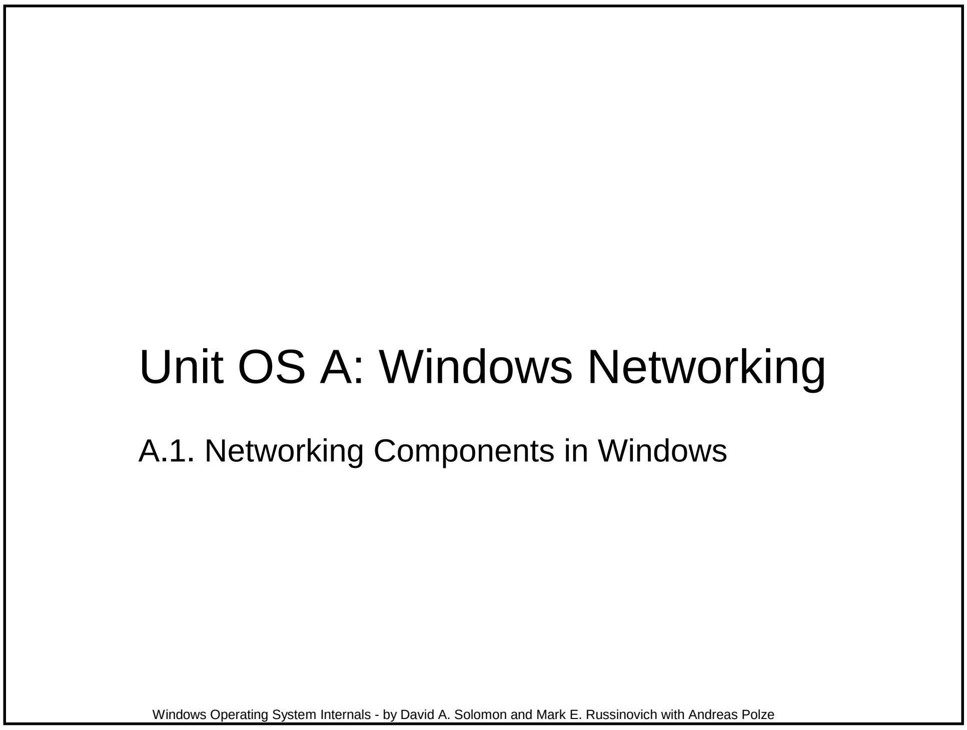
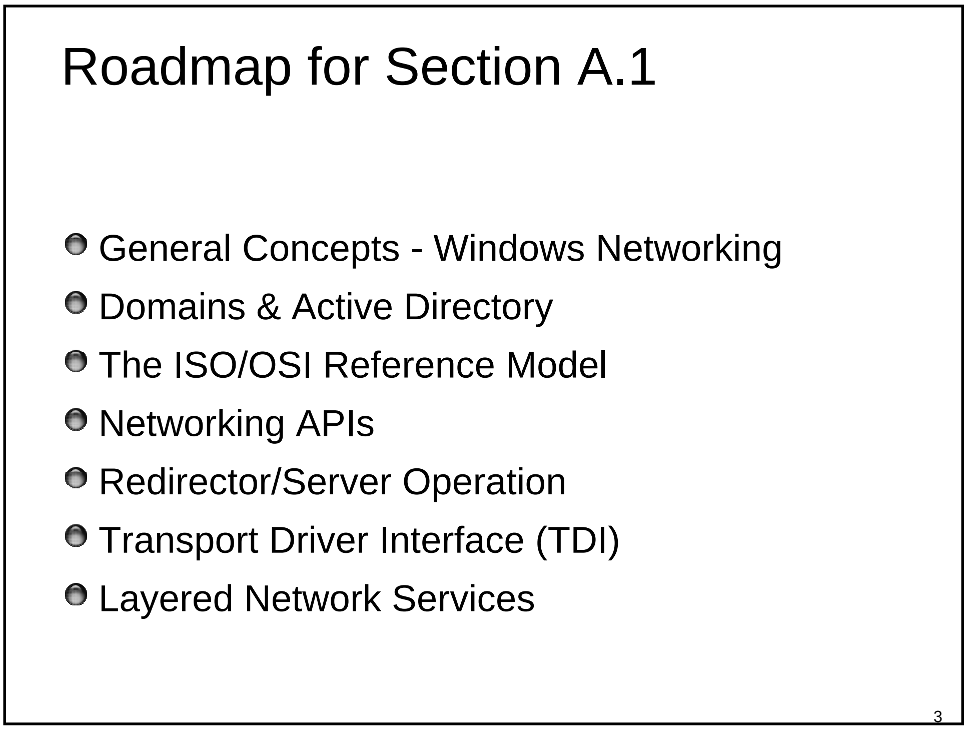


Unit OS A: Windows Networking

A.1. Networking Components in Windows

Windows Operating System Internals - by David A. Solomon and Mark E. Russinovich with Andreas Polze



Roadmap for Section A.1

- General Concepts - Windows Networking
- Domains & Active Directory
- The ISO/OSI Reference Model
- Networking APIs
- Redirector/Server Operation
- Transport Driver Interface (TDI)
- Layered Network Services

3

Roots of Windows Networking

- MS-DOS 3.1:
  - Added file-locking and record-locking to FAT file system
  - Product: Microsoft Networks (MS-NET; 1984)
  - Uniform naming convention (UNC): NET USE X: \\SERVER\SHARE
- MS-NET established some traditions:
  - Redirector traps I/O requests destined to remote file, directory, printer
  - MS-NET redirector sends request to remote server
  - NT networking supports multiple redirectors
- Server Message Block protocol (introduced in MS-NET)
  - NetBIOS interface (API) to pass I/O requests in SMB format
- Network Server
  - Accepts and handles SMB requests; peer-to-peer networking
- LAN Manager
  - Network domains; share account/security info

Networking in Windows

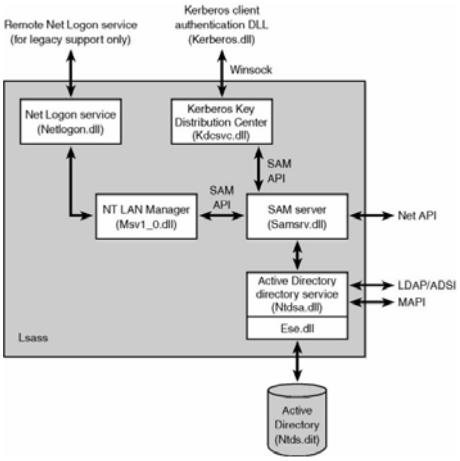
Design goals

- Integral, application-transparent networking services
  - Basic file and print sharing and using services
- A platform for distributed applications
  - Application-level inter-process communication (IPC)
- Windows should provide an expandable platform for other network components

Domains

- Allow a shared security database across a group of computers
  - Each domain controller has a copy
  - Member computers refer to the domain controllers for authentication
- Two styles:
  - Legacy NT 4 Domains
    - Security database stored in Registry SAM & SECURITY hives
    - Limited support for relationships between domains
    - Netlogon for authentication
  - Windows 2000 Active Directory-based Domains
    - Security database stored in Active Directory
    - Win2000/XP/2003 domains support forests – domain hierarchies – for better scaling in large organizations
    - Kerberos authentication

Active Directory


Remote Net Logon service (for legacy support only)
Kerberos client authentication DLL (Kerberos.dll)
Winsock
Net Logon service (Netlogon.dll)
Kerberos Key Distribution Center (Kdcsvc.dll)
SAM API
SAM API
NT LAN Manager (Msv1_0.dll)
SAM server (Samsrv.dll)
Net API
Active Directory directory service (Ntdsa.dll)
Ese.dll
LDAP/ADSI
MAPI
Lsass
Active Directory (Ntds.dit)

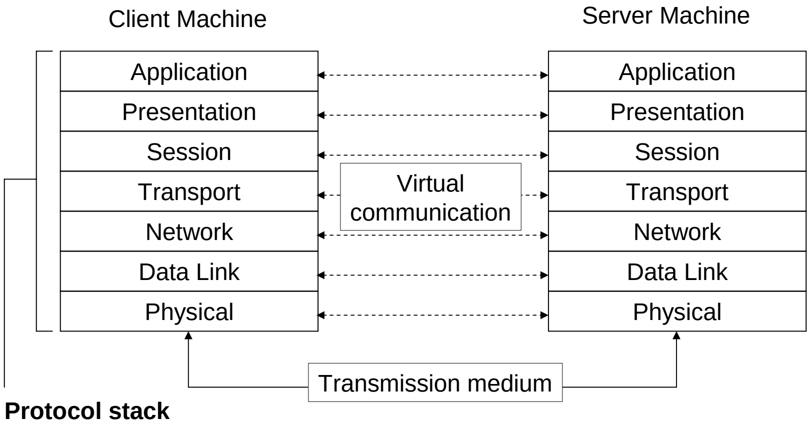
Active Directory

- Active Directory is the Windows implementation of Lightweight Directory Access Protocol (LDAP) directory services
- Active Directory’s core is a database that stores objects representing resources defined by applications in a Windows network
  - File is ntds.dit
- Active Directory supports a number of APIs
  - LDAP C API
  - Active Directory Service Interfaces (ADSI) COM interface
  - Messaging API (MAPI)
  - Security Account Manager (SAM) APIs
  - Windows NT 4 networking APIs (Net APIs)

OSI Reference Model

- Computer network is an *interconnected collection of autonomous computers* (Tanenbaum)
- Standardize and integrate networking software:
  - International Standards Organization defined a software model for sending messages between machines
- Open Systems Interconnection (OSI) reference model
  - Idealized scheme
  - Each layer on one machine assumes that it is „talking“ to the same layer on the other machine
- Each layer provides services to higher layers and abstracts from implementation of services at lower layers

OSI Reference Model (contd.)



Client Machine

Server Machine

Client Machine		Server Machine
Application	↔	Application
Presentation	↔	Presentation
Session	↔	Session
Transport	↔ Virtual communication ↔	Transport
Network	↔	Network
Data Link	↔	Data Link
Physical	↔	Physical

Transmission medium

Protocol stack

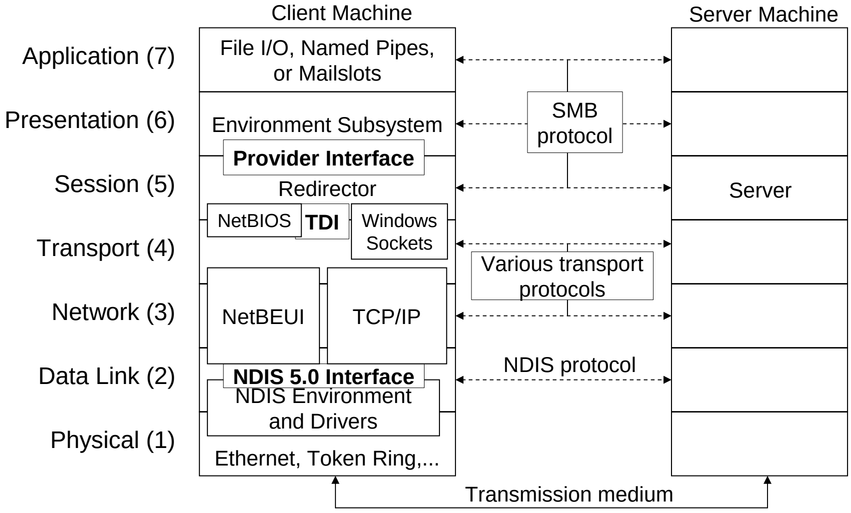
Layers in the OSI Model

- Application layer (7)
  - Information transfer between network apps.,Initiation of data exchange
  - Security checks, identification of participating machines
- Presentation layer (6)
  - Data formatting, data compression, encoding, etc.
- Session layer (5)
  - Manages connection between cooperating applications
  - High-level synchronization and monitoring: who is talking/listening
- Transport layer (4)
  - Divides messages into packets, assigns sequence numbers
  - Segmentation, assembly; hides changes in networking hardware

Layers in the OSI Model (contd.)

- Network layer (3)
  - Routing, congestion control, internetworking
  - Highest layer, that understands network topology
    (physical configuration of machines, type of cabling, bandwidth limits)
- Data-link layer (2)
  - Transmits low-level data frames, waits for acknowledgements
  - Re-transmission of lost packets
- Physical layer (1)
  - Passes bits to the network cable/physical transmission medium

OSI Model and Windows Networking



Client Machine

Server Machine

Application (7)

Presentation (6)

Session (5)

Transport (4)

Network (3)

Data Link (2)

Physical (1)

File I/O, Named Pipes, or Mailslots

Environment Subsystem

Provider Interface

Redirector

NetBIOS

TDI

Windows Sockets

NetBEUI

TCP/IP

NDIS 5.0 Interface

NDIS Environment and Drivers

Ethernet, Token Ring,...

SMB protocol

Various transport protocols

NDIS protocol

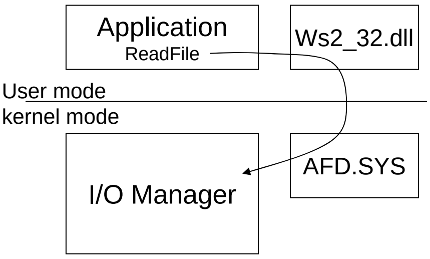
Server

Transmission medium

Networking APIs

- Windows I/O API
  - Open, close, read, write with UNC names referring to remote files
- Windows network (WNet) API
  - Browse file systems via LAN Manager, NetWare, VINES, nfs,...
- Windows named pipe and mailslot APIs
  - Message passing between apps., broadcasting
- NetBIOS API
  - Backward compatibility for MS-DOS, 16-bit Windows, OS/2 apps.
- Windows Sockets API
  - 16/32-bit UNIX-style standard interface for networking
- Remote Procedure Call (RPC) facility
  - Compatible with Distributed Computing Environment (DCE) RPC

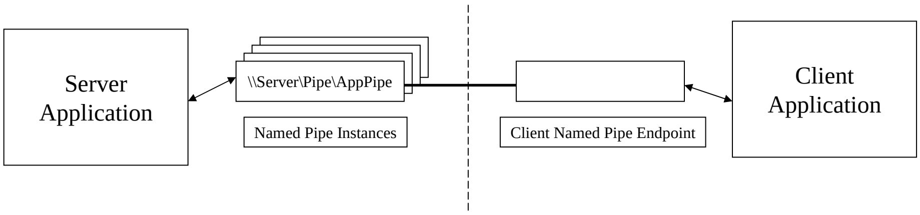
Networking APIs



- Applications link with user-mode DLLs that present the networking API
- Example:
  - Winsock, WS2_32.DLL
- Networking API DLLs often rely on kernel-mode drivers (TDI clients) to interface to network protocol drivers
  - The Winsock libraries maintain socket state information, but also rely on an API driver, AFD, in kernel mode as a foundation
  - Kernel-mode integration with I/O Manager allows file system APIs to also work for networking

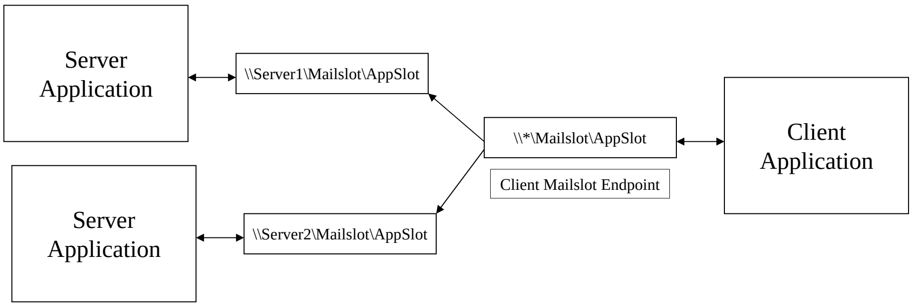
Named Pipes

- Microsoft originally developed these APIs for OS/2 LAN Manager
- Bi-directional, reliable connection-oriented communication
  - Messaging mode for transmitting and receiving full messages
- Fully implemented on Windows, only partially on Win9x (only client support)



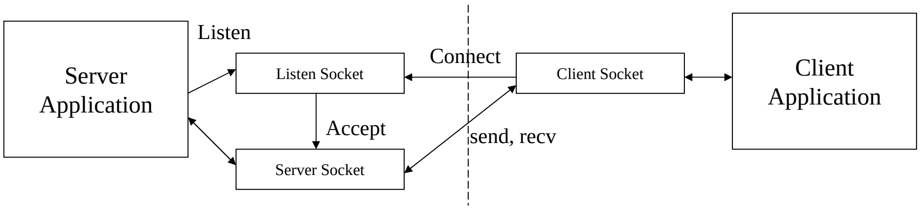
Mail Slots

- Like Named Pipes, are a LAN Manager API
- Supports unidirectional, unreliable broadcast
- Fully implemented on Win9x



Winsock

- Microsoft’s implementation of BSD Unix (Berkeley Software Distribution) Sockets
  - BSD Sockets are the Internet API
  - Used widely on UNIX
  - Winsock consortium helps define Winsock API
  - Reliable connection-oriented (streams) and unreliable connectionless (datagram) modes



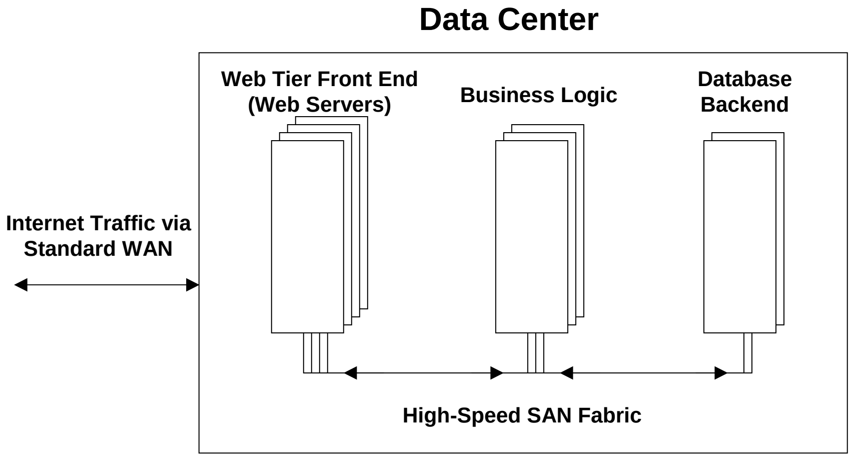
18

System Area Networks

- System Area Networks (SAN) is a connection-oriented server interconnect
  - Not to be confused with Storage Area Networks (SAN)
- Provides reliable, in-order delivery
  - Both network and bus semantics:
    - Messages
    - Remote DMA (memory semantics)
  - Segmentation/reassembly in hardware
- Interconnect types include
  - InfiniBand
  - Ethernet
  - FiberChannel
  - Proprietary
  - Even shared memory

19

System Area Networks



Data Center

Web Tier Front End (Web Servers)

Business Logic

Database Backend

Internet Traffic via Standard WAN

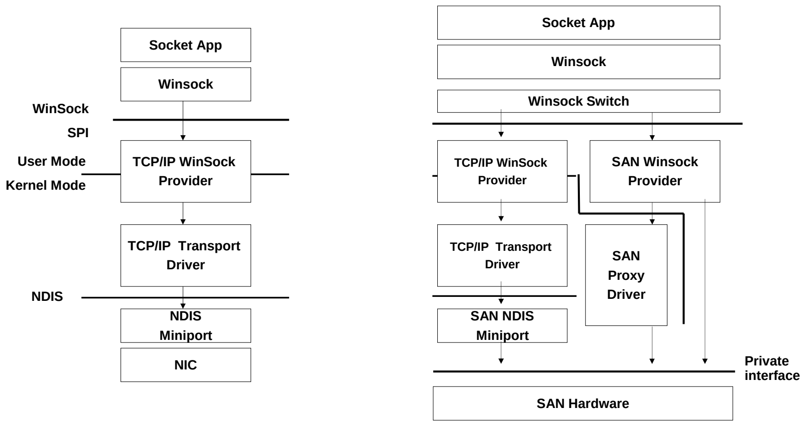
High-Speed SAN Fabric

System Area Networks

- WinSock Direct (WSD) allows applications to get performance benefits of SANs
  - No application modification needed
  - Provides third generation task offload

System Area Networks

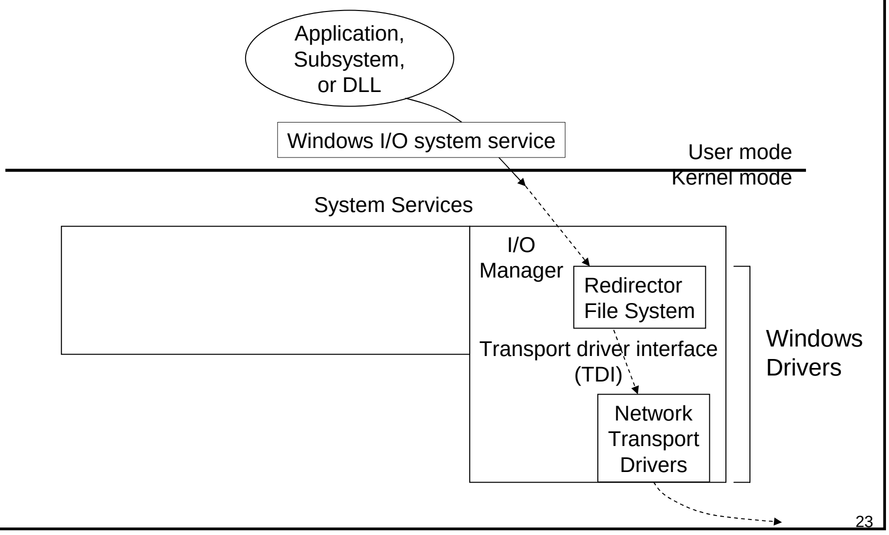
Traditional Model **Winsock Direct Model**



Socket App
Winsock
WinSock SPI
User Mode
Kernel Mode
TCP/IP WinSock Provider
TCP/IP Transport Driver
NDIS
NDIS Miniport
NIC

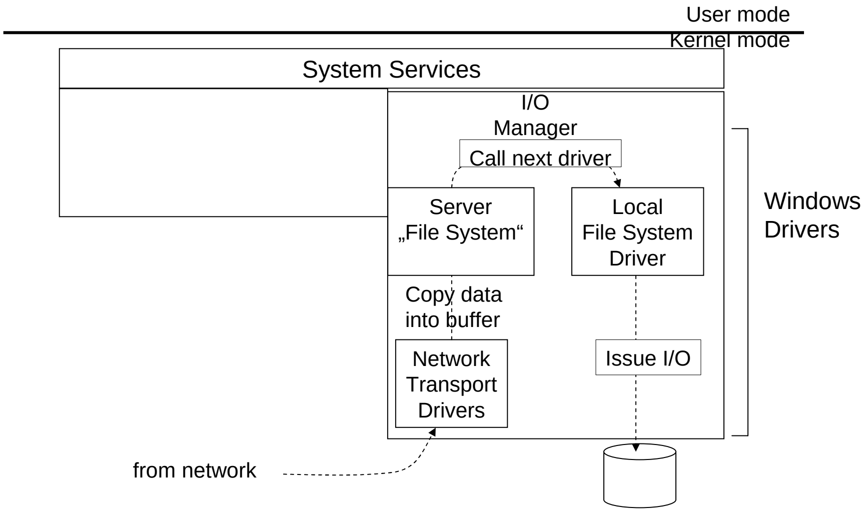
Socket App
Winsock
Winsock Switch
TCP/IP WinSock Provider
SAN Winsock Provider
TCP/IP Transport Driver
SAN Proxy Driver
SAN NDIS Miniport
Private interface
SAN Hardware

Client-Side View of Network I/O



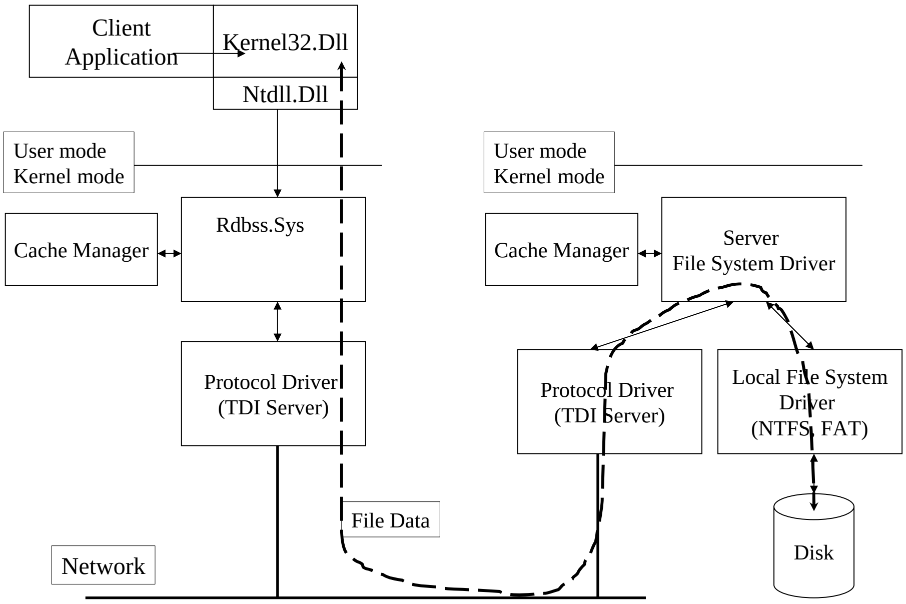
Application, Subsystem, or DLL
Windows I/O system service
User mode
Kernel mode
System Services
I/O Manager
Redirector File System
Transport driver interface (TDI)
Network Transport Drivers
Windows Drivers

Server-side View of Network I/O



User mode
Kernel mode
System Services
I/O Manager
Call next driver
Server „File System“
Local File System Driver
Windows Drivers
Copy data into buffer
Network Transport Drivers
Issue I/O
from network

Network I/O - the complete Picture

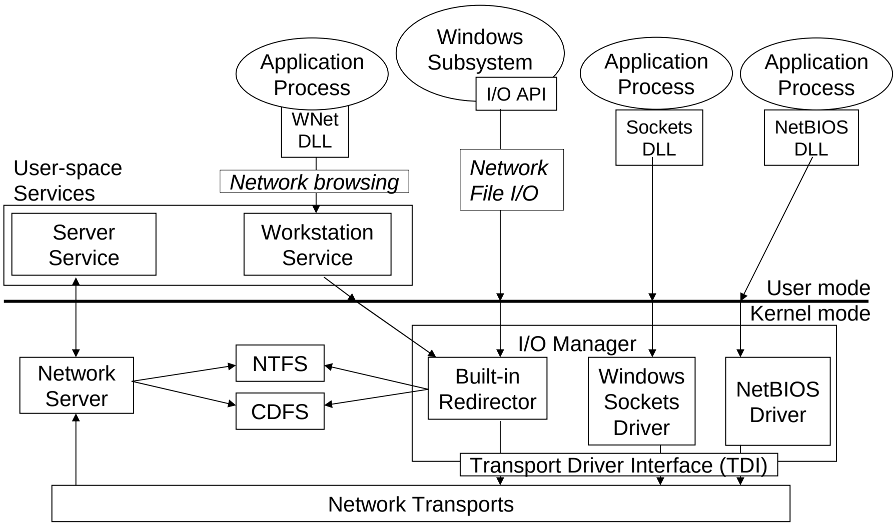


Client Application
Kernel32.Dll
Ntdll.Dll
User mode
Kernel mode
Cache Manager
Rdbss.Sys
Protocol Driver (TDI Server)
User mode
Kernel mode
Cache Manager
Server File System Driver
Protocol Driver (TDI Server)
Local File System Driver (NTFS, FAT)
Disk
File Data
Network

Routes to the Network

- Each API finds its way to the network through a different route
  - Windows API I/O routines call I/O system services;
    I/O manager sends IRPs to redirector
  - Sockets API and NetBIOS API are DLLs, that call I/O services
    I/O manager sends IRPs to Sockets and NetBIOS drivers
- Services – comparable to UNIX daemon processes
  - *Service controller* manages loading and starting of NT services
  - Services may export an API to support specific functions, e.g.:
  - Administering built-in redirector (LAN Man *WS service*, *Server service*)
  - Sending alert messages (disk full) to logged-on users (*alerter service*)
  - Receiving messages (print job notification) from other systems (*messenger service*)

Routes to the Network (contd.)



Built-in Networking Components

- Redirector and network server:
  - Introduced with MS-NET (assembly lang.);
  - completely re-written (C) for Windows NT/2000
  - Implemented as loadable file system drivers
  - Can coexist with other vendor's redirectors and servers
- Implemented as file system drivers, that means:
  - Part of the Windows executive
  - Access to I/O manager's driver interfaces
  - Ability to call cache manager functions directly
  - I/O manager's layered model reflects layering of network protocols
  - Redirector/server can be layered on top of any transport protocol driver – modular components

Redirector/Server Operation

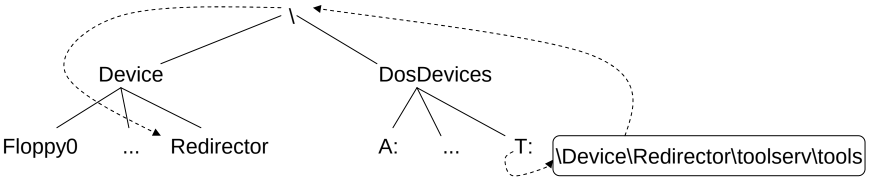
- Compatibility:
  - Works with existing MS-NET & LAN Manager servers (MS-DOS, OS/2, Windows)
  - Can access remote files, named pipes, printers
- Initialization:
  - Driver's init routine creates object *Device**Redirector*
  - Registers dispatch routines for driver operations (open, close, read,..)
- Reliability:
  - Periodic reconnect to servers; mask transient faults, if possible
  - Maintains tables of open files; reopens files on reconnect
- Asynchronous operation: (support for asynch. I/O)
  - Return immediately to user-space process
  - Employ thread in initial system process to wait for I/O completion

Resolving a Network Filename

Extend the reach of local I/O to include remote resources

- All these resources are objects
- Object manager gets involved in opening files

1. User assigns drive letter NET USE T: \\TOOLSERV\TOOLS; workstation service creates symbolic link
2. Windows app. opens file T:\editor.exe
3. Windows subsyst. Translates name to NT object \DosDevices\T:\editor.exe; calls NT executive to open file
4. Object manager substitutes symbolic link to \Device\Redirector



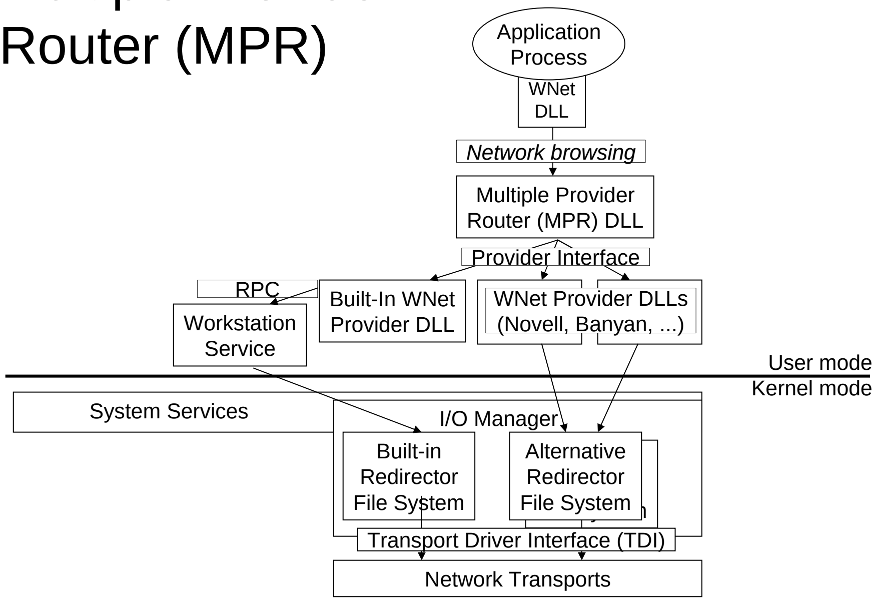
Name Resolution (contd.)

- Device objects:
  - Launching point into an object namespace that is not controlled by the NT object manager
  - Object manager calls parse method associated with the device object
- In our case:
  - Method is an I/O manager routine that calls redirector
  - Redirector builds SMBs (Server Message Blocks)
  - Remote SMB server opens file \editor.exe on \\TOOLSERV\TOOLS
- Locally:
  - NT object manager creates local file object to represent opened file
  - Returns object handle to caller; subsequent op. go directly to redirector
- Remote object namespace:
  - Contains \Device\Server; used to manage the server by name
  - Not used when server receives request

Open Architecture

- Redirector, network server, transport drivers can be loaded/unloaded dynamically
  - A variety of such components can coexist
- Windows supports multiple networks:
  - Access to file systems for resource connection, network browsing, and for remote file and device I/O through common Windows WNet API
  - Multiple network transport protocol drivers can be loaded simultaneously; redirectors access them through common interface
  - Supplies interface and environment (NDIS 3.0) for network card drivers to access NT transport drivers
- Access to remote files systems via:
  - **Multiple provider router** (MPR) – a DLL which determines which network to access when an app uses Windows WNet API
  - Multiple UNC provider (MUP) – a driver that determines which network to access when an app uses Windows I/O API to open remote files

Multiple Provider Router (MPR)



Application Process

WNet DLL

Network browsing

Multiple Provider Router (MPR) DLL

Provider Interface

RPC

Workstation Service

Built-In WNet Provider DLL

WNet Provider DLLs (Novell, Banyan, ...)

User mode

Kernel mode

System Services

I/O Manager

Built-in Redirector File System

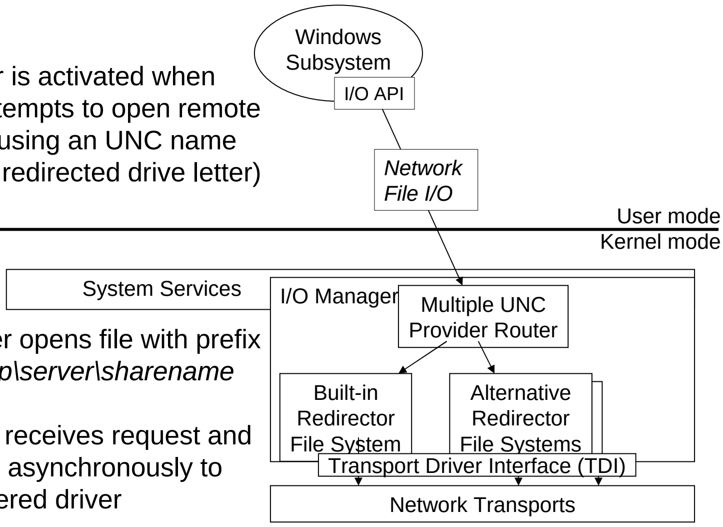
Alternative Redirector File System

Transport Driver Interface (TDI)

Network Transports

Multiple UNC Provider (MUP)

MUP driver is activated when app first attempts to open remote file/device using an UNC name (instead of redirected drive letter)


Windows Subsystem
I/O API
Network File I/O
User mode
Kernel mode
System Services
I/O Manager
Multiple UNC Provider Router
Built-in Redirector File System
Alternative Redirector File Systems
Transport Driver Interface (TDI)
Network Transports

I/O manager opens file with prefix \Device\Mup\server\sharename

MUP driver receives request and sends IRPs asynchronously to each registered driver

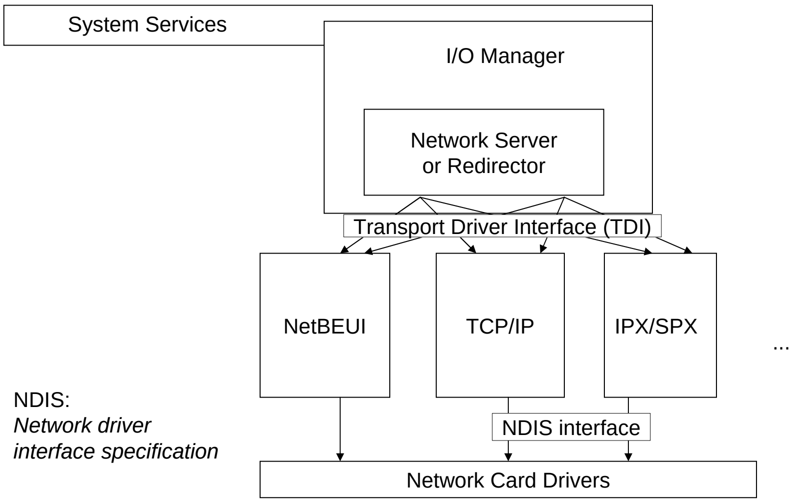
Transport Driver Interface

- Transport protocols are implemented as drivers
- Windows provides a single programming interface for redirectors and other high-level network drivers
  - Transport Driver Interface – TDI – allows redirectors and servers to remain independent from transports
- A single version of a redirector or server can use any available transport mechanism
- TDI is asynchronous,
  - Implements generic addressing mechanism
  - Variety of services and libraries

TDI Clients

- Transport Device Interface (TDI) clients:
  - Support network interfaces
  - Show up as file systems – have some characteristics of file systems
- TDI is a kernel-mode interface
- The TDI interface is a convention for IRP device I/O commands and formatting IRP buffers
- A single interface allows a client use the same interface semantics to communicate with multiple TDI protocol drivers
- Example TDI clients:
  - AFD – Winsock TDI client
  - MSFS – MailSlot TDI client
  - NPFS – Named Pipe TDI client

Transport Driver Interface (contd.)



System Services

I/O Manager

Network Server
or Redirector

Transport Driver Interface (TDI)

NetBEUI

TCP/IP

IPX/SPX

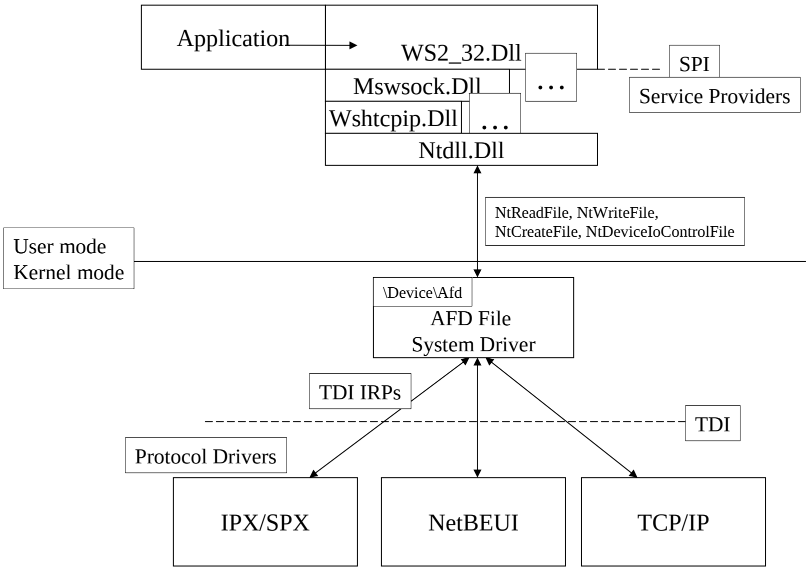
...

NDIS:
Network driver
interface specification

NDIS interface

Network Card Drivers

TDI - The bigger Picture



Application

WS2_32.Dll

SPI

Service Providers

Mswsock.Dll

…

Wshtcpip.Dll

…

Ntdll.Dll

NtReadFile, NtWriteFile,
NtCreateFile, NtDeviceIoControlFile

User mode
Kernel mode

\Device\Afd

AFD File
System Driver

TDI IRPs

TDI

Protocol Drivers

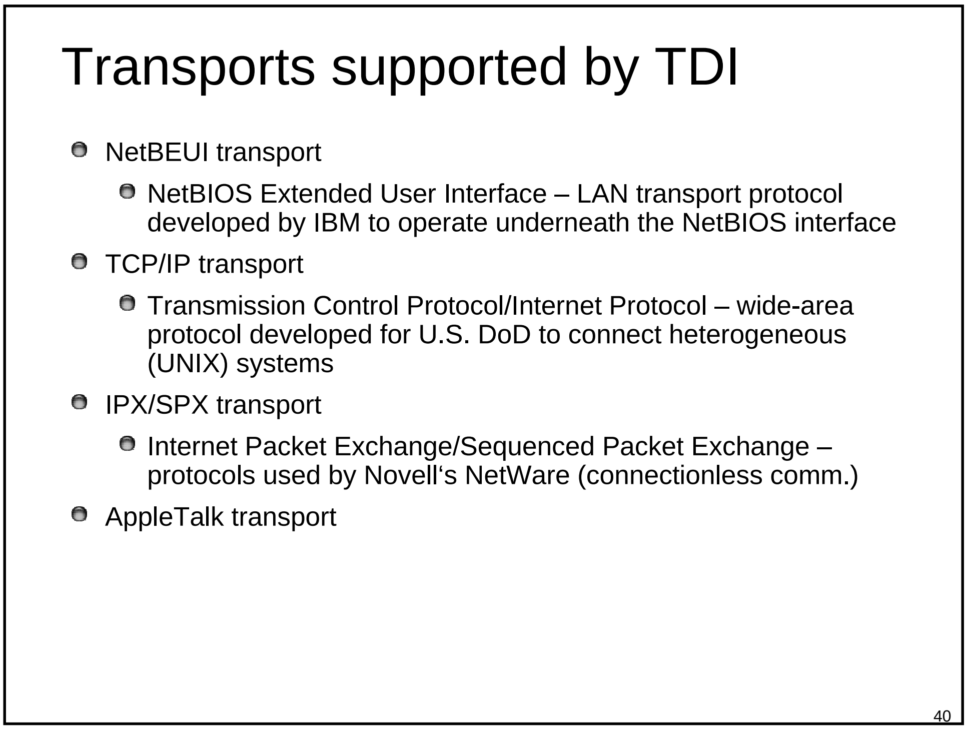
IPX/SPX

NetBEUI

TCP/IP

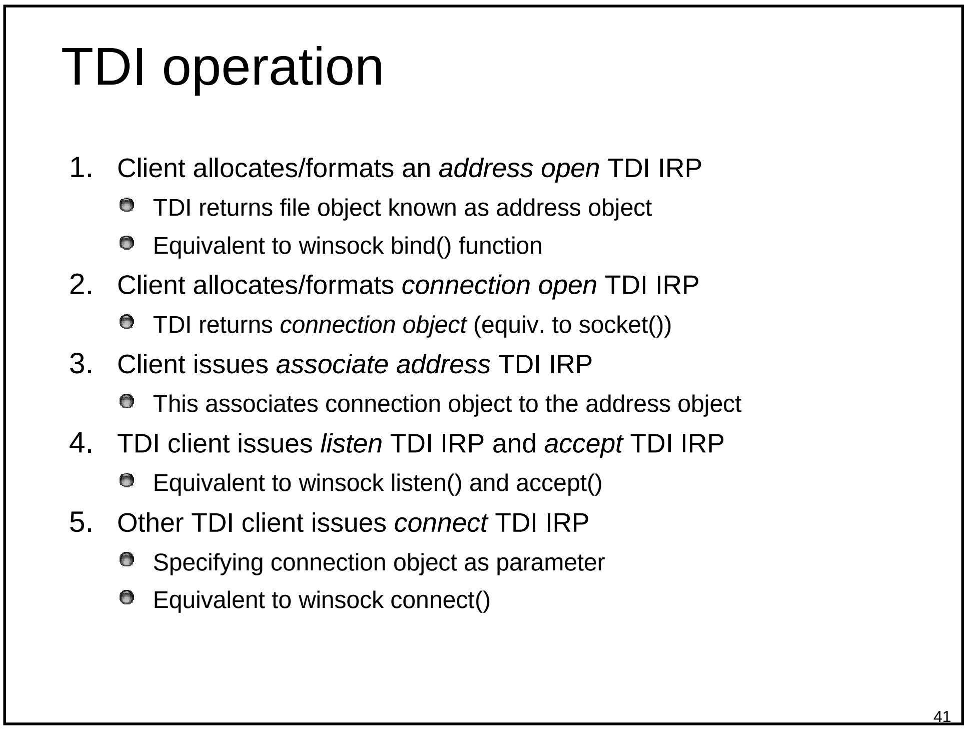
TDI Transports

- API providers rely on TDI Transports (also called transport protocol drivers) in kernel mode to take API requests and translate them into low-level network protocol requests for transmission across the network



Transports supported by TDI

- NetBEUI transport
  - NetBIOS Extended User Interface – LAN transport protocol developed by IBM to operate underneath the NetBIOS interface
- TCP/IP transport
  - Transmission Control Protocol/Internet Protocol – wide-area protocol developed for U.S. DoD to connect heterogeneous (UNIX) systems
- IPX/SPX transport
  - Internet Packet Exchange/Sequenced Packet Exchange – protocols used by Novell‘s NetWare (connectionless comm.)
- AppleTalk transport



TDI operation

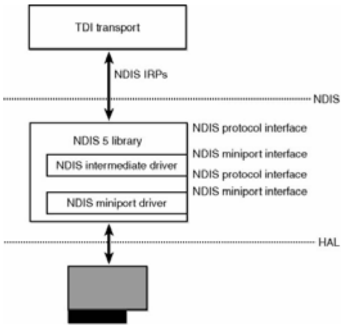
1. Client allocates/formats an *address open* TDI IRP
   - TDI returns file object known as address object
   - Equivalent to winsock bind() function
2. Client allocates/formats *connection open* TDI IRP
   - TDI returns *connection object* (equiv. to socket())
3. Client issues *associate address* TDI IRP
   - This associates connection object to the address object
4. TDI client issues *listen* TDI IRP and *accept* TDI IRP
   - Equivalent to winsock listen() and accept()
5. Other TDI client issues *connect* TDI IRP
   - Specifying connection object as parameter
   - Equivalent to winsock connect()

TDI operation (contd.)

- TDI also supports connectionless protocols (UDP)
- TDI supports registering *event callbacks*
  - Functions directly invoked by TDI (event notification)
  - No need to pre-allocate resources (buffers)
- TDI uses NDIS 5 interface to talk to drivers
  - Network Driver Interface Specification (Microsoft/3Com spec., 1989)
  - NDIS hides IRP mechanism from network driver:
    same driver may work for Windows 2000/XP/ME
  - NDIS 4 did serialization of requests on driver level (MP scalability ??)
  - NDIS 5 allows driver to specify concurrency constraints

NDIS Miniport Drivers

- NDIS drivers are also called NDIS Miniport drivers
  - Microsoft supplies the NDIS port driver
  - The miniport interface hides the specifics of the Windows I/O model, which allows miniport drivers function on Win9x as well
- NDIS processes own I/O request packets, NDIS packets



NDIS Intermediate Driver

- Lie between the NDIS miniport driver and the TDI transport driver
- Can add functionality such as load-balancing and quality of service
- Examples:
  - Windows 2000 QOS packet classifying driver
  - Network Monitor driver

NDIS 5 Features

- Report whether network medium is active
  - TCP/IP uses this information to reevaluate DHCP addressing info.
- TCP/IP task offloading
  - Packet checksums or IPsec can be handled at network adaptor level
- Fast packet forwarding
  - Network adaptor may perform routing (without delivering them to CPU)
- Wake-on-LAN
- Connection-oriented NDIS
  - Manage connection-oriented media such as Asynchronous Transfer Mode (ATM) devices

Microsoft TCP/IP - Overview

- Core protocol elements, services, and the interfaces between them.
- Transport Driver Interface (TDI) and Network Device Interface (NDIS) are public
  - specifications are available from Microsoft.
- A number of higher level interfaces available to user-mode applications.
  - The two most commonly used are Windows Sockets and NetBIOS.

TCP/IP Implementation in Windows

- **Support for Standard Features**
  - Ability to bind to multiple network cards with different media types
  - Logical multi-homing
  - Internal IP routing capability
  - IGMP (IP Multicasting) support
  - Duplicate IP address detection
  - Multiple default gateways
  - Dead gateway detection
  - Automatic Path Maximum Transmission Unit (PMTU) discovery
- **Performance Enhancements**
  - Greatly reduced broadcast traffic
  - Shorter code paths/reduced CPU utilization
  - Self-tuning features

TCP/IP in Windows (contd.)

- Services Available
  - Dynamic Host Configuration Protocol (DHCP) client and server
  - Windows Internet Name Service (WINS), a NetBIOS name server
  - Domain Name Server (DNS) (added in Windows NT 4.0)
  - Point-to-Point Tunneling Protocol (PPTP) used for virtual private remote networks
  - Dial-up (PPP/SLIP) support
  - TCP/IP network printing (lpr/lpd)
  - SNMP agent
  - Wide Area Network (WAN) browsing support
  - High-performance Microsoft Internet Information Server
  - Basic TCP/IP connectivity utilities, including: finger, FTP, rcp, rexec, rsh, Telnet, and tftp
  - Server software for simple network protocols, including: Character Generator, Daytime, Discard, Echo, and Quote of the Day
  - TCP/IP management and diagnostic tools, including: arp, hostname, ipconfig, lpq, nbtstat, netstat, ping, route, and tracert

Windows Sockets 2 in Windows

Windows Sockets 2 Features

- **Access to protocols other than TCP/IP**
  - Windows Sockets 2 allows an application to use the familiar socket interface to achieve simultaneous access to a number of installed transport protocols
- **Overlapped I/O with scatter/gather**
  - Windows Sockets 2 incorporates the overlapped paradigm for socket I/O and incorporates scatter/gather capabilities as well, following the model established in Windows environments
- **Protocol-independent name resolution facilities**:
  - Windows Sockets 2 includes a standardized set of functions for querying and working with the myriad of name resolution domains that exist today (for example DNS, SAP, and X.500)

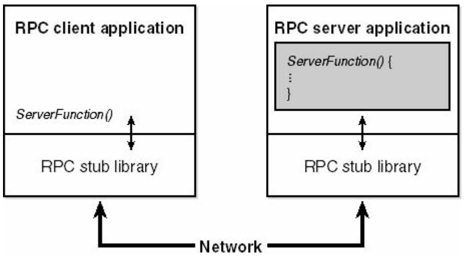
Windows Sockets 2 (contd.)

- **Protocol-independent multicast and multipoint**:
  - Windows Sockets 2 applications discover what type of multipoint or multicast capabilities a transport provides and use these facilities in a generic manner.
- **Quality of service**
  - Window Sockets 2 establishes conventions applications use to negotiate required service levels for parameters such as bandwidth and latency. Other QOS-related enhancements include mechanisms for network-specific QOS extensions.
- **Other frequently requested extensions**
  - Windows Sockets 2 incorporates shared sockets and conditional acceptance; exchange of user data at connection setup/teardown time; and protocol-specific extension mechanisms.

RPC

- Remote procedure call (RPC) is a network programming standard originally developed in the early 1980s
  - The Open Software Foundation (now the Open Group) made RPC part of the Distributed Computing Environment (DCE)
  - Although there is a second RPC standard, SunRPC, the Microsoft RPC implementation is compatible with the OSF/DCE standard
- RPC builds on other networking APIs, such as named pipes or Winsock, to provide an alternate programming model that in some sense hides the details of networking programming from an application developer
- An RPC compiler generates networking code that application executes as network-transparent functions
- Examples of applications that use RPC:
  - Remote Registry service

RPC



Networking APIs (summary)

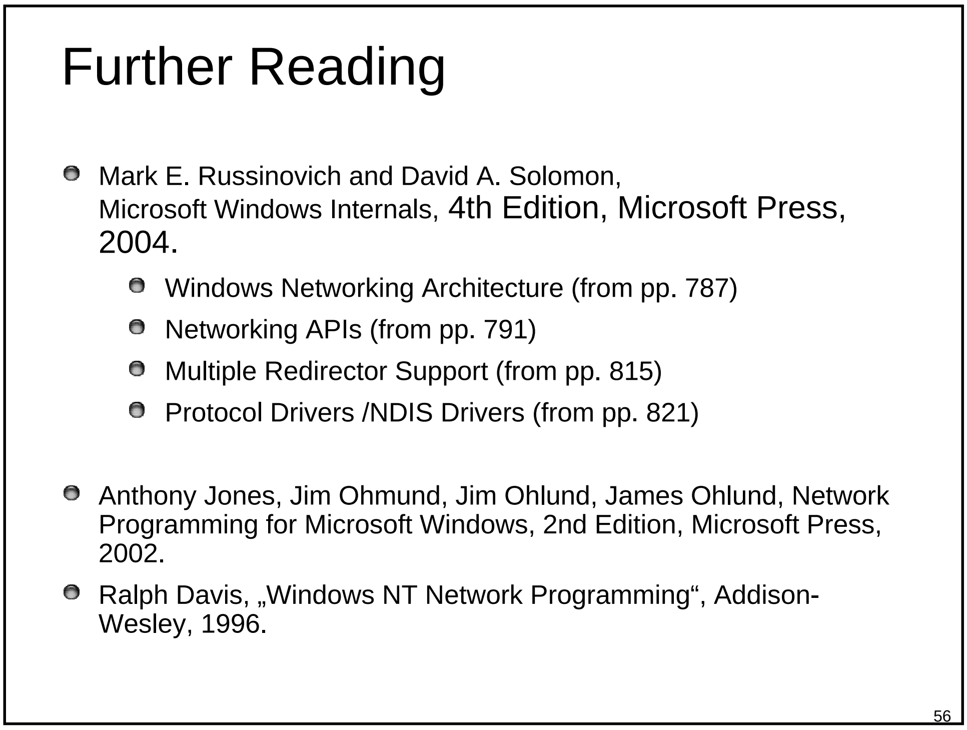
- Named Pipes and Mailslots
- Windows Sockets (winsock)
  - Extensible API on Windows (via service provider interface – SPI)
  - Transport service providers: TCP/IP, NetBEUI, AppleTalk, IPX/SPX, ATM, IrDA (Infrared Data Association)
  - Namespace service providers: DNS, Active Directory, IPX/SPX
- Remote Procedure Call (DCE RPC)
- Common Internet File System (CIFS – SMB)
- Network Basic Input/Output System (NetBIOS)
- Telephony API
  - TAPI 2.2 for C Apps, TAPI 3.0 for COM Apps
- Component Object Model – COM+
  - Message Queuing

Layered Network Services

- Remote Access
  - Dial-up remote access via Telco-infrastructure
  - Virtual private network (VPN):
    virtual point-to-point connection via IP network (Internet)
- Active Directory: Windows impl. of LDAP
  (Lightweight Directory Access Protocol)
  - LDAP C language API
  - Active Directory Service Interfaces (ADSI) – COM Interface to AD
  - Messaging API (MAPI) – compatibility with Exchange/Outlook
  - Security Account Manager (SAM) APIs interface with auth. packages
    - MSVl_0 (\Winnt\System32\Msvl_0.dll – legacy LanManager auth.)
    - Kerberos (\Winnt\System32\Kdcsvc.dll – Kerberos auth.)

Layered Network Services (contd.)

- Network Load Balancing
  - With Windows Advanced Server, NDIS intermediate driver
  - Useful for certain TCP/IP-based cluster-aware applications
- File Replication Service (FRS)
  - Used to replicate a domain controller‘s \SYSVOL directory
  - Relies on NTFS change journal
- Distributed File System (DFS)
  - Location-transparent resource access
- TCP/IP Extensions
  - Network Address Translation (IP masquerading)
  - Internet Protocol Security (IPsec)
  - Quality-of-Service



Further Reading

- Mark E. Russinovich and David A. Solomon,
  Microsoft Windows Internals, 4th Edition, Microsoft Press, 2004.
    - Windows Networking Architecture (from pp. 787)
    - Networking APIs (from pp. 791)
    - Multiple Redirector Support (from pp. 815)
    - Protocol Drivers /NDIS Drivers (from pp. 821)
- Anthony Jones, Jim Ohmund, Jim Ohlund, James Ohlund, Network Programming for Microsoft Windows, 2nd Edition, Microsoft Press, 2002.
- Ralph Davis, „Windows NT Network Programming“, Addison-Wesley, 1996.