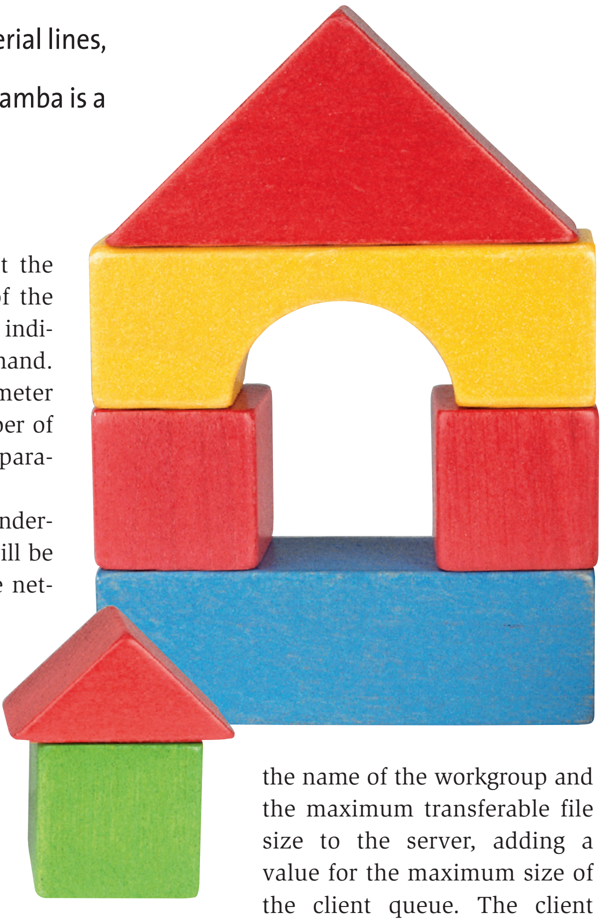
SMB Protocol

Blocks and Messages

The SMB (Server Message Block) protocol specifies how Windows computers communicate on a network. SMB provides access to files, printers, serial lines, and communication channels, such as named pipes and mailslots. Samba is a free SMB implementation. BY ANDREAS ROESCHIES



SMB is client/server protocol based on the request/response principle. The client transmits a request to a server, which in turn transmits a reply to the client. The server will not contact the client without a prior request with the exception of a few specific cases. In the world of SMB, clients are defined as systems that access shares, which are provided by servers. Most SMB capable computers can be used both as clients and as servers (peer-to-peer networking).

Microsoft has referred to the SMB as CIFS (Common Internet File System) for quite a while now, after all, it does sound more modern. However, the documentation provided by Microsoft is scanty and incomplete – an official standard does not exist. The developers of Samba were thus forced to analyze network dialogs and draw their own conclusions.

SMB messages have a simple format. An SMB message (sometimes known as an SMB packet) comprises a header and the actual data content. The header always begins with a protocol ID, followed by a single byte of command code. The additional header fields are the error class, the error code, the tree identifier (TID), and the individual ID of the calling process, a user ID and a multiplex identifier.

Simple Message Format

Some of these fields are interspersed with blank fields, which are reserved for future use. The content of an SMB message comprises a command or the reply to a command. The length of the command or the reply is variable. Each command has a series of parameters and data that are transmitted at the same time. The first field of the command is 1 byte long and indicates the length of the command. It is followed by the parameter words, a value for the number of parameters, and finally the parameters themselves.

To gain a more complete understanding of the protocol it will be helpful to examine a simple network dialog. When a user accesses a shared directory or printer, the session layer is provided by the NetBIOS software interface, which is thus the only dialog partner available to applications. The client transmits a session request packet containing the NetBIOS name of the server and the client to the server. The server replies with *session granted* to set up the session.

Initiating a Session

The client and server can then begin to negotiate the protocol variant. To do so, the client sends another command, *SMBnegprot*, to the server. This packet contains a list of all SMB protocol variants that the client is capable of. The server replies with a pointer to the protocol variant it prefers, where *0* will point to the first protocol dialect in the list sent by the client. A reply of *0xFF* indicates that the server does not speak a compatible protocol and prevents any further communication.

In the next step, the client sends a username and the appropriate password, the name of the workgroup and the maximum transferable file size to the server, adding a value for the maximum size of the client queue. The client uses the same messages to send the next command, that is the connect request (tree connect). The server now sends a tree identifier (TID) back to the client, allowing the client to open, read, write and close files.

The server additionally sends the service identifier, *A*, to the client to indicate a shared directory. The other possible service IDs are *PT1* for a print queue, *COMM* for a serial line and *IPCQ* for a named pipe.

Name Resolution under NetBIOS

As is the case for other networks each node must be uniquely identifiable. 16 byte NetBIOS names are used for this purpose. As the 16th byte is used as a service identifier (with a similar function to TCP/IP ports), only the first 15 characters are visible and usable. In contrast to DNS, SMB/CIFS is not hierarchical, that is, NetBIOS provides a flat namespace where each host can possess multiple names, provided they are unique on the network.

The Name Management Protocol, which is similar to the Appletalk Binding Protocol, handles name management on the network. New nodes repeatedly broadcast a name request. If no other computer replies that it is already using the name, the requesting computer will add the name to its local table. Broadcasts are also used to assign NetBIOS names to network addresses.

When setting up a connection, the client will send a packet with a specific name request to the local network, and the machine with the specified name will reply. The Name Management Protocol (NMP) is responsible for name and address negotiation.

Samba Components

smbd: This daemon provides file and print services on the network. The configuration file is called *smb.conf*.
nmbd: This daemon provides NetBIOS name resolution and browsing services. It is also configurable in *smb.conf*.
smbsh: This program allows a user to access a directory on another SMB server as if it were local. SMB drives are available below */smb* by default.
smbclient: This program is a simple SMB client which is similar in functionality to an FTP client. It additionally provides a number of diagnostic routines.
smbstatus: Shows active connections. The first section of output shows the share accessed by users. The second section lists any files currently locked by Samba and the third section lists the memory usage of each share.
smbtar: A shell script that calls *smbclient* and performs tar operations.
nmblookup: The NetBIOS equivalent to **nslookup** for TCP/IP, allows the user to resolve NetBIOS names to IP addresses.
smbpasswd: Allows users to change their own passwords, and administrators to define passwords for arbitrary users.
testparm: This program checks the syntax and consistency of the central smb.conf configuration file.
testprns: This tool checks whether the specified printer is defined in the **printcap** printer configuration file.

Browsing lists allow clients to locate servers and shares quickly, in fact, the browsing list is what the Windows user sees in the network neighborhood. The browse service compiles browsing lists and distributes them on the network. The Local Master Browser (LMB), which permanently provides a more or less current browsing list for its own subnet, is one of the most important components on the network.

Browse Services

In principle, any SMB server can assume the role of the LMB; various criteria such as the uptime or operating system version are applied to decide which server will be the master browser. When an SMB server that is configured as an LMB candidate starts up, it can initiate an election to decide the LMB in the local subnet. The Local Master Browser can promote one or multiple LMB candidates to backup LMBs by transferring a copy of its own browsing list to them.

If an LMB goes down, its role is assumed by a backup LMB. Clients can continue to browse the network without needing to reconstruct their browsing lists. A Microsoft SMB server decides the number of backup LMBs by reference to the number of NetBIOS nodes on the subnet.

Additionally, the administrator can assign hosts to be Domain Master Browsers (DMB). DMBs manage the browsing lists of entire domains, which can comprise multiple subnets. List management on a network will tend to be slow, and this means that SMB servers take a few moments to appear in the lists, and that downed servers appear as ghosts in the browsing lists for almost an hour after being shut down.

SMBs Ancient History

The roots of SMB go back to NetBIOS, which was programmed for IBM by Sytec in 1983. NetBIOS was a software interface, as an Application Programming Interface (API) and a matching transport protocol. Although the transport protocol is more or less unused today (and is no longer included in NetBIOS), the API has survived.

Servers and clients communicated by exchanging message, so-called Server Messages Blocks (SMB). The transport protocol that NetBIOS comprises is referred to as IBM NetBIOS Frame Protocol (NBF) by IBM. You will often hear references to the NetBIOS protocol, although, strictly speaking, this is a misnomer. The protocol was used exclusively in proprietary IBM PC networks, and was capable of addressing a maximum of 80 nodes.

As the IBM PC network proved inadequate over the years IBM developed Token Ring, which can address up to 260 nodes, and also supports subneting (rings can be attached by means of bridges). In 1985 IBM wrote an emulation which allows the use of the NetBIOS interface in Token Ring networks to allow applications to be ported to this environment.

NBF was replaced by the NetBIOS Extended User Interface Protocol (NetBEUI). It was at this point that NetBIOS became a pure API. The NetBEUI transport protocol, which was developed for Token Ring, also works in Ethernet networks and is a good choice for small Windows networks without Internet access as it does not require any configuration. The naming scheme, where each node possesses one or more NetBIOS names, was retained for NetBEUI.

Working with Friends

Things start to get complicated when other transport protocols are used. In this case NetBIOS names need to be mapped to network addresses. Additionally, addresses in routable networks possess a network address that neither NetBIOS nor NetBEUI can understand in addition to the host address. The only protocol that plays a significant role in the communication between Linux and Windows computers is TCP/IP.

Not until later, when techniques were developed to encapsulate SMB data in other transport protocols, was it possible to access shares in routed networks. The encapsulation in TCP/IP is often referred to as NetBIOS over TCP/IP (or NBT/NetBT for short) by Microsoft. This technique is specified in RFCs 1001 (Protocol Standard for a NetBIOS Service on a TCP/UDP Transport; Concepts and Methods) and 1002 (Protocol Standard for a NetBIOS Service on a TCP/UDP Transport; Detailed Specifications). The approaches detailed in these RFCs are designed to allow existing NetBIOS services to run under open standards.

The central issue of integration is resolving NetBIOS names to IP addresses. As broadcasts, which are used for name resolution in NetBEUI networks, cannot cross router boundaries, another method of name resolution needed to be found. The technique involves a name server for NetBIOS, or NBNS (NetBIOS Name Server), although Microsoft refers to NBNS as WINS (Windows Internet Name Service).

All that Glitters is not DNS

NetBIOS name servers work in a similar way to DNS servers, but for the SMB protocol instead of TCP/IP. One major advantage of WINS over DNS is the fact that clients can register independently with the server, thus avoiding admin intervention. Manual assignments are unnecessary, but possible.

SMB servers that do not report to a name server will not appear in the browsing list, but are still accessible on the network. You need to know their names to access them. If there are multiple NBNSs on the network, they can exchange data to ensure that they stay synchronized. Clients can request the nearest NBNS if they need to resolve a name to an address. This reduces network traffic and the load on the routers.

Babylonian Node Configurations

A TCP/IP environment provides various configuration options for clients and servers. An SMB computer configured as a *B* (for broadcast) node will use broadcasts both to register its own name and for name resolution. Normally, routers will refuse to relays broadcasts, which means that *B* nodes will only see servers on the local network.

B nodes on Microsoft Windows first inspect their local NetBIOS name cache when performing name resolution, and only broadcast if they do not come up with a result. If the broadcast is equally unsuccessful they then inspect their local *lmhosts* files (similar to *hosts* for TCP/IP).

P (for point-to-point) nodes only communicate with NetBIOS name servers, whether they need to register their own names or discover another machine's NetBIOS name. This assumes that they know the IP of at least one NBNS, of course. The admin user can supply this information manually, or use the DHCP (Dynamic Host Configuration Protocol) option 044.

The disadvantage of the *P* node configuration is that SMB communication will collapse if the NBNS goes down, something that will even affect the local subnet. *M* (for mixed) mode provides a solution to this issue; the node will first attempt to broadcast before resorting to point-to-point communication with an NBNS if this fails. This theoretically increases the performance in local networks. If the NBNS is not available, the machine can still access SMB servers in its own subnet.

H (for hybrid) nodes do exactly the opposite, that is, they resort to broadcasts only when the NBNS is unavailable. Additionally, an *H* node will attempt to contact the NBNS at regular intervals to reinstate point-to-point operations as soon as the NBNS becomes available.

As work on the Samba project started a long time after SMB was introduced, the developer team did not bother to look into the NetBIOS Frame Protocol, instead opting to author a TCP/IP implementation. To retain compatibility to other implementations, Samba still needs to respect some quirks under TCP/IP such as special name resolution techniques.

More Flexible Than the Original

The *smb.conf* provides a series of configuration parameters that influence the behavior of the server software. Samba provides far more configuration options than Windows itself. These include useful features such as the simultaneous use of multiple SMB names. This is particularly useful if you want to use a single Linux system to replace multiple Windows servers. And Samba also provides more enhanced security features than the original, such as IP address based access control. ■

INFO

[1] Richard Sharpe: "Just what is SMB" (detailed introduction): *http://www.samba.org/cifs/docs/what-is-smb.html*

[2] CIFS-Homepage from Microsoft: *http://www.microsoft.com/mind/1196/cifs.asp*

GLOSSARY

Backup Browser: An SMB server that provides backup browsing services. The server regularly receives a copy of the current browsing list from Local Master Browser and replaces the LMB if it goes down.

Browser Election: When an SMB server that is a potential browser candidate starts up, it can force a browser election. This procedure decides which server will be the Local Master Browser.

CIFS (Common Internet Filesystem): Another name for the Server Messages Blocks (SMB) protocol.

Domain Master Browser: An SMB server that manages browsing lists over multiple subnets. The Domain Master Browser receives the browsing lists compiled by Local Master Browsers.

Local Master Browser: The Local Master Browser has the authoritative browsing list for its own subnet.

Master Browser: A short form of Local Master Browser.

NBF (NetBIOS Frame Protocol): The transport port of the first NetBIOS implementation, replaced by NetBEUI in 1985, sometimes referred to as the NetBIOS protocol, which is, strictly speaking, a misnomer.

NBNS (NetBIOS Name Server): A server that manages a table containing NetBIOS name to IP address mappings. An NBNS is only required in routed SMB networks where TCP/IP is used as the transport protocol.

NetBEUI (NetBIOS Extended User Interface): A simple, self-configuring, non-routable protocol that works both in Ethernet and in Token Ring networks.

NetBIOS (Netword Basic Input Output System): An Application Programming Interface for network applications. Formerly included the NBF transport protocol.

NetBT (NetBIOS over TCP/IP): Also known as NBT.

Browser candidate: An SMB server capable of maintaining a browsing list.

SMB (Server Message Block): A protocol for file, printer and serial line access. Also known as CIFS.

WINS (Windows Internet Name Server): The Microsoft implementation of a NetBIOS name server.