SMB Analyzer (Server Message Block)

Seth Hall
ICSI/Broala/LBNL

INTERNATIONAL COMPUTER SCIENCE INSTITUTE

```
------------------------------------------------------------------------
r44 | vern | 2004-06-09 10:29:27 -0400 (Wed, 09 Jun 2004) | 2 lines

SMB analyzer from Ruoming Pang.
```

It was only SMB1 and architected in a hybrid Binpac/C++ style.

Server Message Block

From Wikipedia, the free encyclopedia

In computer networking, **Server Message Block** (**SMB**), one version of which was also known as **Common Internet File System** (**CIFS**, /ˈsɪfs/),[1][2] operates as an application-layer network protocol[3] mainly used for providing shared access to files, printers, and serial ports and miscellaneous communications between nodes on a network. It also provides an authenticated inter-process communication mechanism.

How many versions?!

Contents [hide]

1 Features

2 History

- 2.1 SMB 2.0
- 2.2 SMB 2.1
- 2.3 SMB 3.0
- 2.4 SMB 3.0.2
- 2.5 SMB 3.1.1

Fortunately everything after 2.0 is small revisions on the same thing!

Wikipedia forgot about the SMB 1.0 protocol?!

Business [illegible] SMB.

There was some pain…

```
type SMB1_read_andx_response(header: SMB_Header, offset: uint16) = record {
    word_count          : uint8;
    andx                : SMB_andx;
    available           : uint16;
    data_compact_mode   : uint16;
    reserved1           : uint16;
    data_len_low        : uint16;
    data_offset         : uint16;
    data_len_high       : uint16;
    reserved2           : uint64;

    byte_count          : uint16;
    pad                 : padding to data_offset - SMB_Header_length;
    data                : bytestring &length=data_len;

    extra_byte_parameters : bytestring &transient &length=(andx.offset == 0 |
    offset+offsetof(extra_byte_parameters))+2) ? 0 : (andx.offset-(offset+off

    andx_command        : SMB_andx_command(header, 0, offset+offsetof(andx_comm
} &let {
    is_pipe     : bool   = $context.connection.get_tree_is_pipe(header.tid);
    pipe_proc   : bool   = $context.connection.forward_dce_rpc(data, 0, false

    padding_len : uint8  = (header.unicode == 1) ? 1 : 0;
    data_len    : uint32 = (data_len_high << 16) + data_len_low;
    proc        : bool   = $context.connection.proc_smb1_read_andx_response(h
} &byteorder=littleendian;
```

```
type SMB2_write_request(header: SMB2_Header) = record {
    structure_size      : uint16;
    data_offset         : uint16;
    data_len            : uint32;
    offset              : uint64;
    file_id             : SMB2_guid;
    channel             : uint32; # ignore
    data_remaining      : uint32;
    channel_info_offset : uint16; # ignore
    channel_info_len    : uint16; # ignore
    flags               : uint32;
    pad                 : padding to data_offset - header.head_length;
    data                : bytestring &length=data_len;
} &let {
    is_pipe: bool = $context.connection.get_tree_is_pipe(header.tree_id);
    pipe_proc : bool = $context.connection.forward_dce_rpc(data, file_id.persistent+file_id._volatile, true)
    &if(is_pipe);

    proc : bool = $context.connection.proc_smb2_write_request(header, this);
};
```

There was one h… on…

Raw materials for new logs (events!)

SMB1

smb1_check_directory_request,
smb1_check_directory_response, smb1_close_request,
smb1_create_directory_request,
smb1_create_directory_response, smb1_echo_request,
smb1_echo_response, smb1_logoff_andx,
smb1_negotiate_request, smb1_negotiate_response,
smb1_nt_cancel_request, smb1_nt_create_andx_request,
smb1_nt_create_andx_response,
smb1_open_andx_request, smb1_open_andx_response,
smb1_query_information_request,
smb1_read_andx_request, smb1_read_andx_response,
smb1_session_setup_andx_request,
smb1_session_setup_andx_response,
smb1_transaction_request, smb1_transaction2_request,
smb1_trans2_find_first2_request,
smb1_trans2_query_path_info_request,
smb1_trans2_get_dfs_referral_request,
smb1_transaction2_response,
smb1_tree_connect_andx_request,
smb1_tree_connect_andx_response,
smb1_tree_disconnect, smb1_write_andx_request,
smb1_write_andx_response, smb1_message,
smb1_empty_response, smb1_error

SMB2

smb2_close_request, smb2_close_response,
smb2_create_request, smb2_create_response,
smb2_negotiate_request,
smb2_negotiate_response, smb2_read_request,
smb2_session_setup_request,
smb2_session_setup_response,
smb2_set_info_request, smb2_file_rename,
smb2_file_delete, smb2_tree_connect_request,
smb2_tree_connect_response,
smb2_write_request, smb2_message

Other

ntlm_negotiate, ntlm_challenge,
ntlm_authenticate, gssapi_neg_result,
dce_rpc_message, dce_rpc_bind,
dce_rpc_bind_ack, dce_rpc_request,
dce_rpc_response

Logs!

- **smb_mapping.log**
  - When a client maps a drive share, that mapping is documented here.
- **smb_files.log**
  - When an action on file is seen on a share, it's presence is documented along with timestamps. The user can customize what actions to log. This is where things like renames and deletes will go (SMB2 only for now!). Files that are actually transferred will be logged in *files.log*
- **dce_rpc.log**
  - Remote procedure calls. Watch for remote admin!
- **ntlm.log**
  - Authentication using NTLM. This is only integrated into the SMB analyzer right now, but later will be integrated in other places.

smb_mapping.log

Fiel

ts, uid, id, path, service, nati

Multiplexing pain!
These were over the same TCP connection.

\\\\nas1.ads.bigco.com\\APPS	-	-	DISK
\\\\nas1.ads.bigco.com\\IPC$	-	-	PIPE
\\\\nas1.ads.bigco.com\\APPS	-	-	DISK
\\\\fs2.ads.bigco.com\\HOME	-	-	DISK

smb_files.log

Fields

ts, uid, id, fuid, action, path, name, size, prev_name, times

dce_rpc.log

Fields

ts, uid, id, rtt, named_pipe, endpoint, operation

```
0.009484 \\pipe\\lsass     lsarpc     LsarOpenPolicy2
0.008416 \\pipe\\lsass     lsarpc     LsarLookupNames
0.009191 \\PIPE\\srvsvc    srvsvc     NetrShareGetInfo
0.010550 \\pipe\\lsass     samr       SamrConnect5
0.010242 \\pipe\\lsass     samr       SamrOpenUser
```

ntlm.log

Fields

ts, uid, id, username, hostname, domainname, success, status

```
alice      BRO-X1225   ADS   T  SUCCESS
bob        BRO-R105    ADS   T  SUCCESS
caroline   BRO-D1225   ADS   T  SUCCESS
-          ARG-5655    -     F  ACCESS_DENIED
david      BRO-E1105   ADS   T  SUCCESS
```

Back to “Business Runs on SMB”

```
action: SMB::FILE_RENAME
path: -
name: BUDGET\\XXXXXXXXXXX\\SALARY POOL\\3.5% Increases\\2017.xlsx
size: 522901
prev_name: BUDGET\\XXXXXXXXXXX\\SALARY POOL\\3.5% Increases\\6A498300
times.modified: 1457402865.456526
times.accessed: 1457402865.222149
times.created: 1457402865.222149
times.changed: 1457402865.456526
```

Uhhhhh.....

More ideas

- Search for: “finance”, “tax”, “accounting”, “backup”, “audit”, “hr”, “merger”, “acquisition”
  - Look for servers and clients using and looking at those files.
  - Create HoneyPot directories that would match those and watch for access to them.
- Analyze GPO policies (they’re just files!)
- Bitlocker recovery keys being stored as PDFs.
- Ransomware detection!
- File hash detection with Intel framework already works.

Using it today

- If you want to use it today, install git master, the 2.5 Beta, or the 2.5 release when it's available.
- Add **"@load protocols/smb"** to local.bro
- We decided to leave it disabled by default in 2.5 because it's a lot of new code and everyone may not be ready for it.

@load protocols/smb

seth@icir.org
Twitter: @remor