WHITE PAPER: A tool for capturing SMB files with Wireshark

AUTHORS: DAVID PEREZ & JOSE PICO

DATE: 27/05/2010

ABSTRACT

This white paper describes a plugin that we have created and made publicly available for the popular network analyzer Wireshark. The plugin adds to Wireshark the ability to extract and save separately, from any network capture, either live or previously saved, the contents of any files transferred between a server and a client using the SMB protocol.

OBJECTIVES

Most corporate networks include one or more file servers where shared information is stored and shared across the network using the SMB protocol. These servers are used as a repository for different departments, which share the same infrastructure but must have access to different and separate information sets, some of which will probably be very sensitive and confidential, like files belonging to top management, Human Resources or the Legal departmens, just to name a few examples.

The access control to the information in the file servers is enforced using the SMB protocol authentication, usually integrated with some unified directory (like Microsoft Active Directory).

While the authentication can be performed in a secure way, the information flow between the server and consumer is usually not encrypted, as it happens with the default SMB configuration. This makes this information vulnerable to any sniffing activity performed in the company's internal network.

In our effort to identify weak points of corporate networks, we wanted to demonstrate how this vulnerability could be easily exploited, so that organizations better understand the risk this vulnerability poses for them, and how to protect themselves from it.

Prior to deciding to develop this tool, we evaluated the existing commercial and non-comercial tools available in the market, but none showed satisfactory results for our purposes, so we decided to build our own tool.

Once we decided to write our own tool, we immediately decided that the right way to do it would be to build a plug-in for Wireshark[1], probably the best open source packet sniffer available, thus extending his functionality.

Since then, we have successfully used the plug-in in some real penetration tests, demonstrating how vulnerable a corporate file server can be to this kind of attack, and how big the impact on the business could be if such an attack were performed by a real attacker.

We hope this work will help corporations in better understanding the risk and thus motivate them to take additional measures in order to protect the information contained in theirs file servers.

THE PLUG-IN: "EXPORT OBJECT SMB"

DESCRIPTION

The "export object smb" is a plugin for Wireshark software that extends its functionality in order to allow the user to save to disk partial or complete SMB objects (files) contained in a Wireshark capture. It has been inspired in the exiting "export http" functionality in the sense that we tried to make it work in a similar way.

It is important to remark that the plugin does not perform any transformation of the data received. If a full file gets transferred across the network, and Wireshark captures that traffic, and the plugin is used to save a copy of that file to disk, it should be possible to open it with the right software (for example, if the file is a MS Excel spreadsheet, it should be possible to open it with MS Excel), because the plugin does not make any alterations on the contents of the file. On the other hand, if only part of a file gets transferred and captured, the plugin will save that data to a file, but it will probably not be possible to open it directly with the appropirate application (for example, MS Excel may refuse to open a partial XLS file because it will consider it to be

[1] http://www.wireshark.org

corrupt). The information on those partially captured files may still be accessible, though, using special tools, like the "strings" command to extract the strings of text, for example.

The plugin works for Wireshark captures, both saved and online. In this whitepaper we use saved captures to illustrate the examples, but the plugin works just as well with ongoing Wireshark captures.

The plugin also has its limitations. In its current form, the plugin can only capture objects that fit in memory. This means that the plugin will not be able to save captured files that are really big. This might be a problem if the tool was to be used for other purposes, but since the goal of the tool is simply to demonstrate the potential risk of sending and receiving unencrypted SMB traffic, it this limitation is not really a problem.

At the time of release of this whitepaper, the plugin (in the form of a patch) is in the revision process by the Wireshark team, so it has not been included yet in any official release. The plugin is already available for everyone to use, but until it gets included in the official build, manual modification and compilation of Wireshark is required in order to get it to work (see “installation” below).

We will try and maintain the patch up to date, but be aware that with some of latest trunks of Wireshark the patch may not work until we provide the corresponding update. If you detect that situation, please let us know and we will update the patch as soon as we can. Keep in mind, however, that we cannot make any promises as to what the time frame will be.

INSTALLATION

From Wireshark development version 33229 on, this functionality has been included in Wireshark official development trunk. That means that, in order to be able to use it along with other Wireshark features, you only have to download the latest Wireshark development trunk and compile it (compilation has been tested by Wireshark team in different environments).

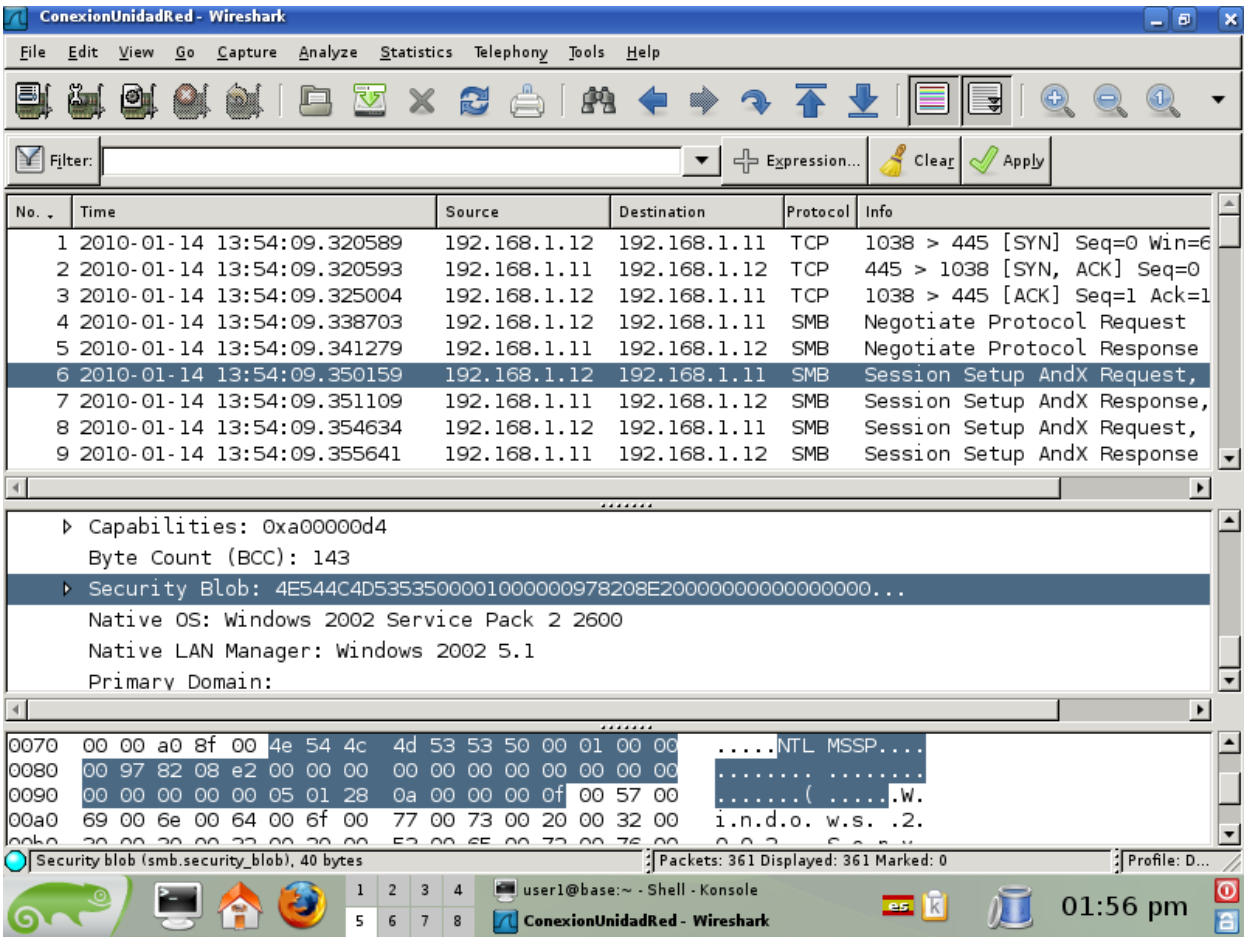
Instructions for compiling Wireshark on different environments can be found in the Wireshark developer's guide:

http://www.wireshark.org/docs/wsdg_html_chunked/

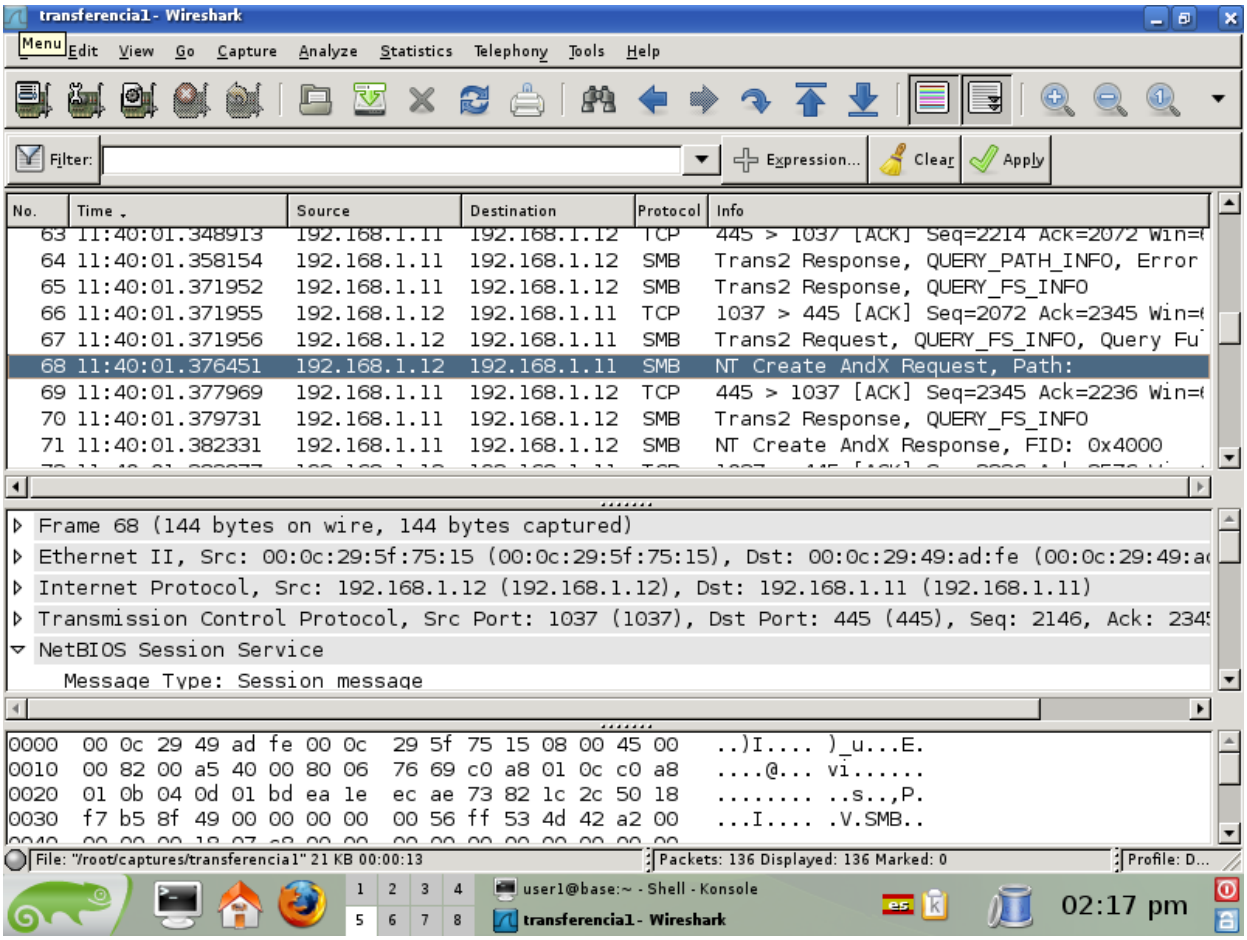
USAGE

IDENTIFYING SMB STREAMS

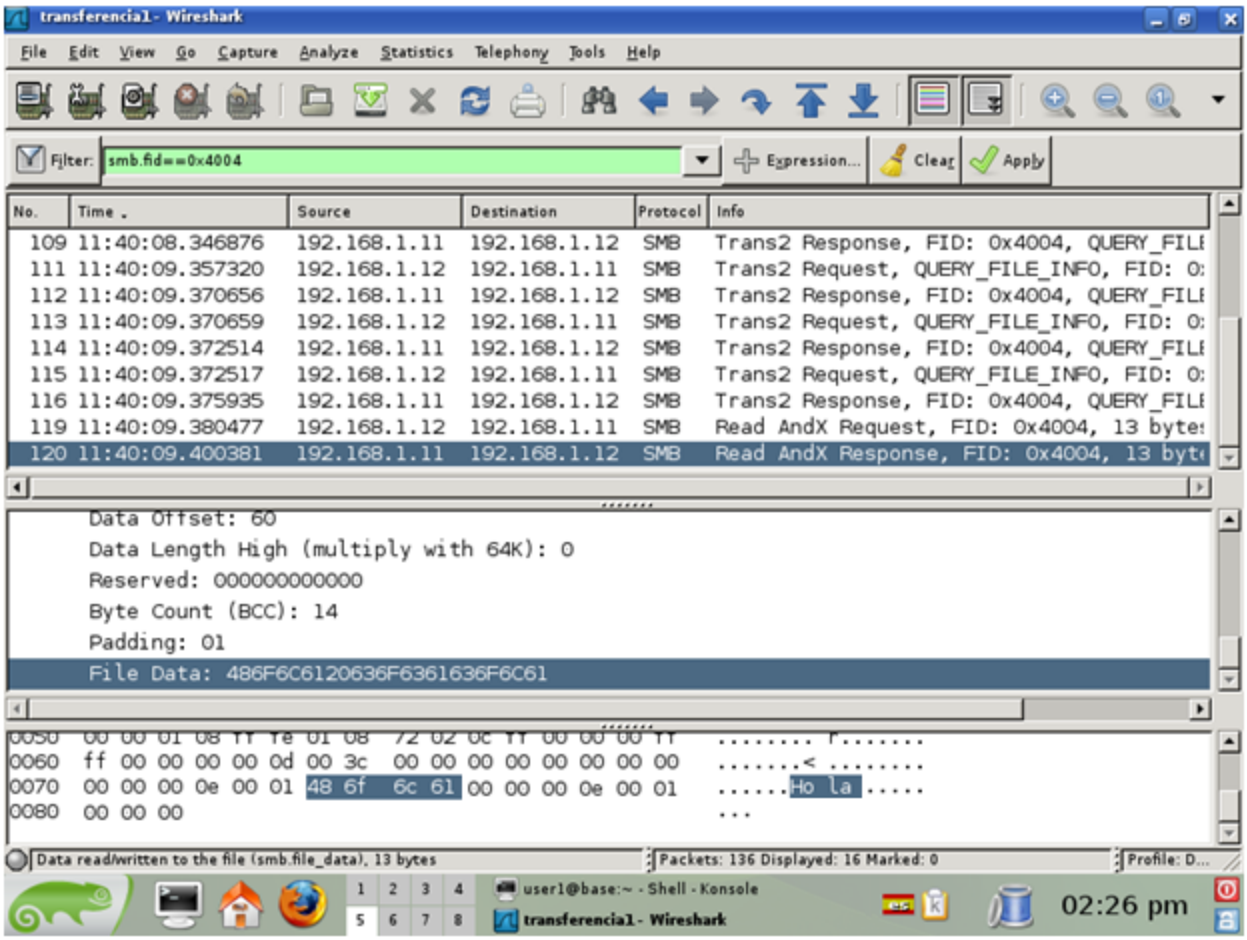
The first thing to do in order to extract SMB files from a network capture, is to identify what SMB streams our pcap file contains. This can be done by manually observing the trace with appropriate filters, where we could see, for example, the SMB authentication and session setup process (see the following figure),



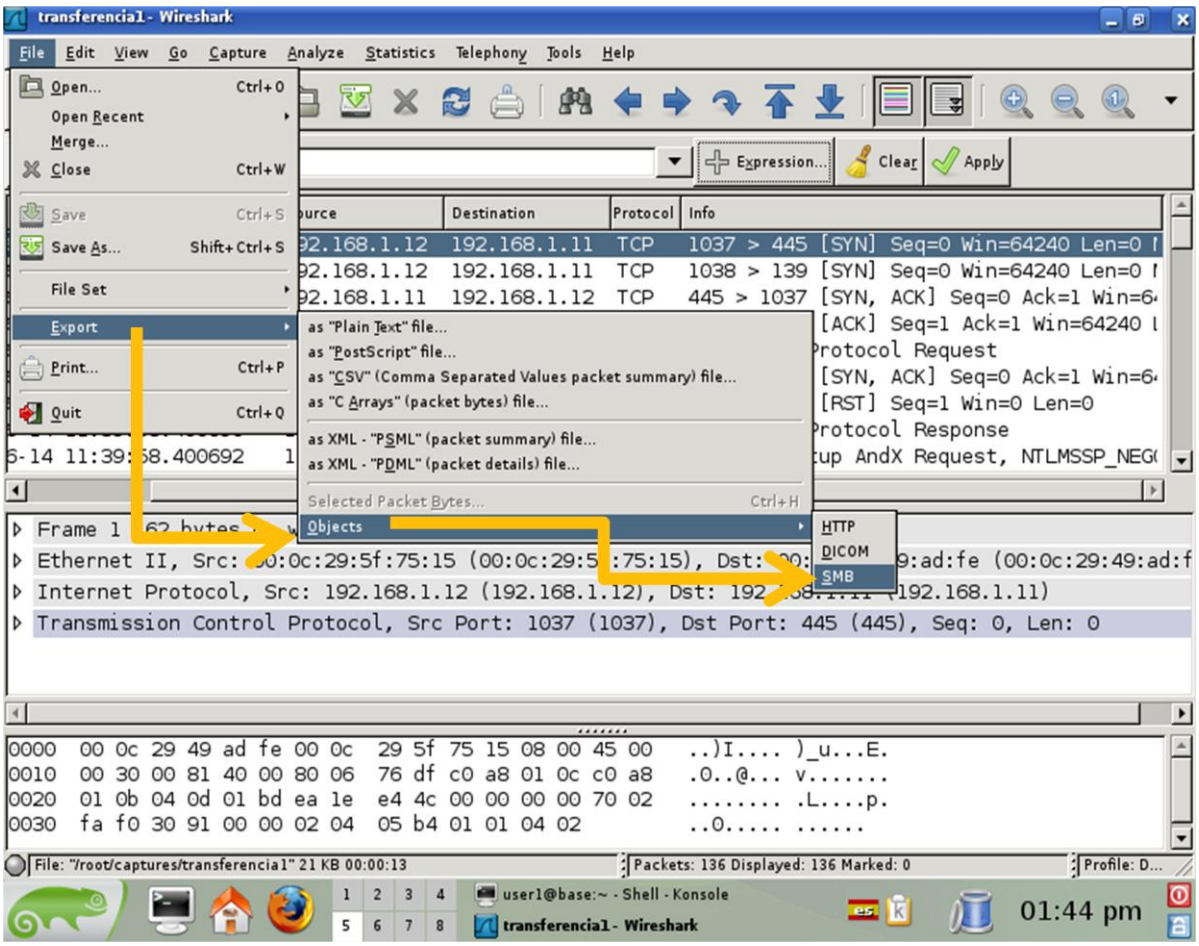
the SMB protocol usual commands (see the following figure),



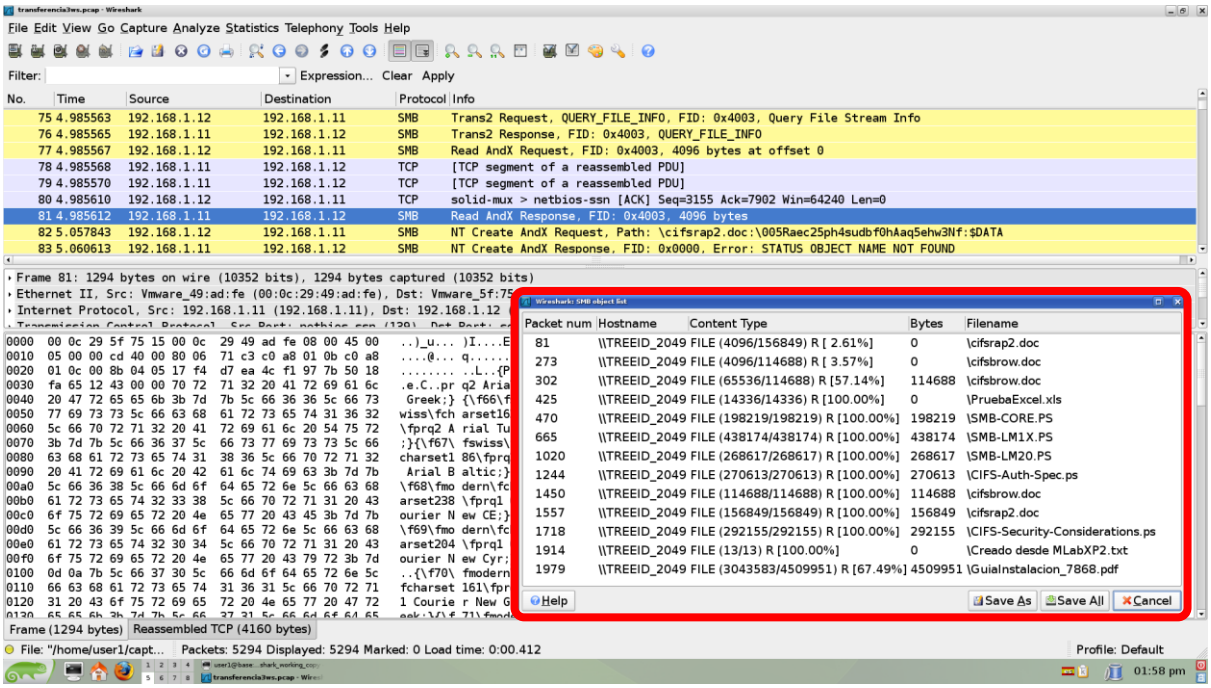
or even the plaintext of a file flowing from server to client (see the following figure):



Alternatively, you may use the SMB plugin to directly identify in your pcap file all objects (files, directories or pipes)transferred by means of SMB streams:



This will give you a list of the SMB objects contained in the trace:



In order to keep modifications of Wireshark code to a minimum, we have maintained the *export object* functionality untouched as much as possible. That is the reason why we have not changed the name of the columns even though the meaning for our plugin is not exactly accurate. Let us explain with what information we populate each column:

- *Packet num:* this column shows the packet number (Wireshark's frame number) of the first packet belonging to a particular file transfer that the plugin has analyzed. Usually, it corresponds to the first SMB command of a transfer that has data to capture (typically read_andx_response or write_andx_request).
- *Hostname:* here, you can find the path to the share that is being served. It is presented in 2 ways:
    - *\\Hostname\sharename*: if the connection between the client and server has been made while Wireshark was capturing, then the plugin identifies the name of the server and the path of the shared folder, and shows this information here.

  - *\\TREEID_<treeID#>*: if, on the other hand, the trace did not include the moment when the connection was established, the plugin cannot obtain the hostname and share name and thus it only shows an identifier of the share to which the corresponding object belong.
- *Content Type*: in this field the plugin shows s whole bunch of data, in the following format: "*type (bytes captured/bytes in file) mode [%captured]*", where:
  - type: possible values are FILE, DIRECTORY, or PIPE, but only FILE type objects are tracked and can be saved.
  - bytes captured: amount of bytes captured in this trace for that file. It may be different from the file length or the bytes in memory, since holes in the file are filled out with zeros.
  - bytes in file: length of the file, in bytes. Initially, this is calculated from the information in the SMB calls, but it may be modified afterwards, if bytes with greater offset are captured.
  - mode: possible values are "R" for files captured in read operations, "W" for files captured in write operations or "R&W", for files captured in read and write operations.
  - [% captured]: percentage reflecting how much of the file was actually captured. It may show “!mem” value if the file doesn’t fit into memory.
- *Bytes*: actual size, in memory, of the captured file. It corresponds to the biggest reported offset of all of the captured bytes belonging to the file.

SAVING A FILE

Once the trace has been analyzed, saving a file is as easy as selecting the file in the *export object smb* window and then clicking the *save* button. You may also save all the files in the window at once, but remember that only those that have been completely captured will be directly readable.

REFERENCES

[1] Version Control with Subversion. Ben Collins-Sussman/Brian W. Fitzpatrick/C. Machael Pilato. http://svnbook.red-bean.com/

[2] Re: [Ethereal-dev] Plugin for making RTP analysis. Ethereal-dev: January 2003. http://www.ethereal.com/lists/ethereal-dev/200301/msg00116.html

[3] Wireshark Developer's Guide. The Wireshark Documentation. http://www.wireshark.org/docs/wsdg_html_chunked/

[4] Wireshark Developer's README file. The Wireshark Documentation. http://anonsvn.wireshark.org/wireshark/trunk/doc/README.developer

[5] A Common Internet File System (CIFS/1.0) Protocol - Preliminary Draft. Paul J. Leach, Dilip C. Naik. http://samba.org/samba/ftp/specs/cifs6.txt