CIFS Security Considerations Update

Paul J. Leach

Microsoft

Preliminary Draft – do not cite

Author's draft: 4

This is a preliminary draft of an update to the security considerations section of the CIFS specification, corresponding to an update to the security protocols described in companion documents. It is supplied here as a standalone document for ease of review; if accepted and implemented, it will be incorporated into a future release of the CIFS specification. (This specification may change without notice, and should not be construed as a product commitment from Microsoft Corporation.)

1 Security Considerations

There are four authentication mechanisms, each with their own strengths and weaknesses, as well as attacks that are independent of the authentication protocol.

If the CIFS authentication protocol is correctly implemented, then for all the attacks and weaknesses listed, there are client or server configurations or other countermeasures to defend against them. At the highest configurable security setting, with well-chosen passwords, the best known attack is as hard as attacking DES with chosen plaintext.

This section is for the guidance of implementers, to help them avoid implementation pitfalls, and for system administrators, to help them choose a security configuration that meets their installation's security requirements.

1.1 Security models

There are two security models – share level security and user level security. Even though the models are different, each model involves users demonstrating knowledge of a password to gain access, and the authentication protocols that prove knowledge of the password are the same for both models.

1.1.1 Share level security

When using share level security, there are no per-user passwords; instead, knowledge of the read or write password is what gives read or write access to a share. A session is established by connecting to a share using the SMB_TREE_CONNECT request, using one of the session authentication mechanisms described below.

Since the same passwords must be disclosed to many people to be used, and hence known by many people, while the authentication protocols are as strong as for user-level protection, share level security is much weaker that user level security in practice. This scheme may be appropriate for small workgroups of trusted/cooperative users, and where the overhead of managing user accounts is deemed to be too high. The fewer the number of people who know any given password, the better; this is especially true for administrative passwords.

Share level protection is at the discretion of the server, and does not compromise the ability of any other client or server to use stronger security amongst themselves.

1.1.2 User level security

When using user level security, users authenticate themselves by proving that they know a password which is known only to that user. A session is established by connecting to a server using the SMB_SESSION_SETUP_ANDX request, using one of the session authentication mechanisms described below. Thereafter, the identity of the user determines the access allowed to the resources on the server, not knowledge of the password.

As discussed above, user level security is usually preferable to share level security. In addition, it can allow access control at a per-resource granularity, instead of at the granularity of the whole share.

1.2 Attacks on session authentication with plaintext passwords

This authentication protocol sends the client's password in the clear. It is thus completely open to an eavesdropping attack. If eavesdropping is not possible, clients can have their passwords stolen by counterfeit servers. Password guessing attacks are possible, with success dependant on how well-chosen the passwords are.

Passwords sent to such servers should never be the same as passwords used for more secure servers.

It should be used only when absolutely needed for backwards compatibility, and only where the risks of eavesdropping is deemed acceptable, such as relatively isolated networks or on networks with secure (encrypted and/or authenticated) link layers.

In general, unless there are very special and well considered reasons, this protocol should not be used.

1.3 Attacks on session authentication using LM session key

This authentication protocol is more secure than plaintext password authentication, because passwords are never seen in the clear on the network.

The following sections analyze the vulnerability of this protocol to attack using the following techniques:

- Eavesdropping/ brute force attacks
- Chosen plaintext attack
- Dictionary attacks

These attacks, if successful, compromise the client's password, and allow the attacker access to the client's files even after the client's session has ended. Because the same protocol using the NT session key is stronger, its use is recommended except where backwards compatibility is required.

1.3.1 Eavesdropping/ brute force attacks

With the session authentication protocol , an eavesdropper can acquire challenge/response pairs. It can then test a password by using it to generate a key, encrypting the challenge, and comparing it to the corresponding response; by exhaustively trying all possible passwords, the correct one will eventually be found. This is as hard as finding a DES key given a known plaintext/cryptext pair.

1.3.2 Chosen plaintext attacks

With the session authentication protocol, a "man-in-the-middle" or a counterfeit server can choose the challenge which the client will then encrypt using a key derived from the client's password. The ability to choose the plaintext to be encrypted is known to make breaking many ciphers much easier, and it is possible that it also may help break cipher-based one-way functions such as the one used to compute the LM session key.

However, no way to break the DES one-way function used in the session authentication protocol using chosen plaintext is currently known.

1.3.3 Dictionary attacks

1.3.3.1 Online dictionary attacks

If the attacker can eavesdrop, but can not execute a chosen plaintext attack, then it can test any overheard challenge/response pair against a list of common words. Such a list is usually much smaller than the total number of possible passwords. The cost of computing the response for each password on the list is paid once for each challenge.

1.3.3.2 Stored dictionary attacks

If the attacker can execute a chosen plaintext attack, the attacker can compute the session key for many common words and use it to precompute the response to a challenge of its choice, and store a dictionary of (response, password) pairs. Such precomputation can often be done in parallel on many machines. It can then use the chosen plaintext attack to acquire a response corresponding to that challenge, and just look up the password in the dictionary. Even if most passwords are not in the dictionary, some might be. Since the attacker gets to pick the challenge, the cost of computing the response for each password on the list can be amortized over many passwords.

The countermeasure against both these types of dictionary attack is to require users to choose passwords that are not common words. For the second type of attack, if the key space derived from passwords is large enough, then it will be infeasible to store the dictionary.

1.3.4 Small key space resulting from badly chosen passwords

Even when passwords are not allowed to be common words, the combination of the use of only uppercase characters, the usual user practice of choosing passwords that have alpha and perhaps numeric characters, plus the fact that the LM session key construction treats the upper and lower halves of the 14 bytes key almost identically means that the key space may be rather small. Enumerating 7 uppercase characters and digits leads to a key space of 36**7, or 78.3 billion combinations. When this mechanism was introduced nearly a decade ago, this was probably an adequately large key space, but with today's much more powerful systems, it would now be small enough to make a brute force search expensive but feasible upon a challenge/response pair obtained via an eavesdropping attack if the attacker had many powerful hosts available or special purpose hardware.

The countermeasure to this problem is to require users to choose passwords that lead to more possible combinations. For example, just requiring one randomly chosen punctuation character in a password increases the key space by a factor of 13. If all characters of the password are chosen at random, there are 68**7, or 6.7 trillion, combinations. If this is still inadequate, then the NT session key should be used.

1.4 Attacks on session authentication using NT session keys

The session authentication protocol using NT session keys is the same as when using LM session keys; only the key construction is different. As a result, it may be attacked using the same techniques. However, because it uses MD4 to generate the keying material from the password, and because it preserves the password's case, the key space of this protocol is essentially the full 56 bits that single DES allows; this is probably an acceptable length for most purposes (although future dialects may use triple-DES for more assurance). It is still subject to the same chosen plaintext and dictionary attacks as above, and these will be feasible if passwords are badly chosen. As above, the countermeasure is to make sure that passwords are well-chosen, and long enough to have at least 56 bits of randomness.

Other considerations:

- o Transforming the password into Unicode leaves a pattern of alternating zeros and characters in the input to MD4. It is possible that this may allow MD4 to be reversed much more easily, although there is currently no known way to exploit this.
- o MD4 is known to be weak with respect to collisions, compared to MD5 and SHA. It is possible that there may be a way to exploit this to attack its one-wayness, or to exploit the collision properties to limit key search time, although there is currently no known way to do so.

1.5 Other attacks

1.5.1 Connection hijacking

Any attacker that can inject packets into the network that appear to the server to be coming from a particular client, can hijack that client's connection. Once a connection is set up and the client has authenticated, if subsequent packets are not authenticated the attacker can inject requests to read, write, or delete files to which the client has access.

Doing so requires that the injected packets have the right transport level sequence numbers. If the attacker can not eavesdrop, this will have very low probability of success, since the 32 bit initial sequence numbers may be randomly chosen. Even if it can eavesdrop, then it needs "gag" the client, otherwise the client will start getting packets with bad sequence numbers and reset the connection. This requires that the attacker is on a host on the same LAN segment as the client or server and has modified the hosts OS to get direct access to its network card, or has taken over a router between the client and the server. It is significantly more difficult to hijack a connection than to eavesdrop, and doing so only permits the attacker to access files as the client for the duration of the session. (See RFC 1948.)

The countermeasure against connection hijacking is to configure the client or server to require the use of message authentication.

1.5.2 Downgrade attacks

If a client is not appropriately configured, a "man-in-the-middle" can remove the bit in the SMB_COM_NEGPROT response that says the server supports challenge/response, thus fooling a client into thinking that it should supply a plaintext password.

The countermeasure against downgrade attacks is to configure the client or server to require either session or message authentication.

1.5.3 Rogue servers and spoofing by counterfeit servers

A counterfeit server is one that spoofs the DNS name resolution process so that the client gets the counterfeit's IP address instead of the genuine server's IP address, thus fooling the client into connecting to the counterfeit while believing it is connecting to the genuine server.

A rogue server is a server that entices a client into accessing it, and uses some aspect of the interaction to try to mount an attack.

Counterfeit and rogue servers are not detectable by the session authentication mechanism, which only authenticates clients to servers.

If a client is not appropriately configured, a rogue or counterfeit server can use the downgrade attack above to obtain a client's password. A counterfeit server can also execute a denial of service attack by ignoring the client's requests or returning bogus results.

A rogue or counterfeit server can authenticate to a real server as any client that attempts to log in to it, by getting the client to respond to the challenge from the real server.

The countermeasure against rogue or counterfeit servers is to require use of the message authentication protocol., which provides mutual authentication. Also, such attacks can be mitigated by deployment of DNSSEC.

1.5.4 Active message modification attacks

If a client or server not appropriately configured, a router or a host on a LAN segment between the client and server may be able mount an "active message modification attack": it may be able to modify messages sent between the client and server.

The countermeasure to such attacks is to use the message authentication protocol. Active message modification attacks are prevented, because unless the MAC key is known, tampering with the message will cause the MAC to fail to validate.

1.5.5 MAC key attacks

The best attack on the MAC key is to attack the session authentication protocol using one of the techniques above to obtain the password, and then just compute the MAC key per the specification – which reduces to brute force search of the key space. (Message authentication is intended to provide mutual authentication on each message, not increased password strength.)

The next best attack is a brute force search of the 128 bit key space. The message authentication protocol does not send the complete output of the keyed-MD5 hash; only half of it. As a result, it does not expose a full (plaintext, cryptext) pair to an attacker. This will make discovery of the MAC key more difficult, since there should be many potential MAC keys K' with the property that it produces the same first 8 bytes of the hash as the actual key K. However, if an attacker sees a second authenticated message, then the chances that K' produces the MACs on both of them correctly will be 1 in 2**128, so K' is then quite likely the actual key.

Since the key is formed from both the session key (which is per-user and long-lived) and the response (which is per-session), large quantities of data are not hashed using a long-lived key, which might subject it to attack.

See RFC 1828 for an example of keyed MD5 applied to IP security. If not used properly, keyed MD5 may have weaknesses as a MAC. Iterative hashes such as MD5 may be subject to message extension attacks and to cryptanalysis [Kal 95]. The CIFS MAC construction is not subject to the problems identified in [Kal 95], because the text contains an explicit length, which prevents message extension attacks; and because there are always two iterations of the compression function, and only 64 bits of the hash are output, which prevents known cryptanalysis techniques.

1.5.6 Replay attacks

An attacker who can eavesdrop and send packets can attempt a replay attack: resend a request or response previously overheard.

The countermeasure against replay attacks is to use the message authentication protocol. Replay is prevented, because each request and response has a unique, strictly increasing sequence number which is incorporated into the computation of the MAC. (Multi-message requests and responses all have the same request or response sequence number, but contain a unique fragment sequence number which prevents replay.)

1.6 Other considerations

1.6.1 Privacy

This version of the CIFS protocol does not support privacy protection. In order to obtain it, one may use a privacy protecting network layer protocol (such as IPSec, PPTP, or L2F) or a privacy protecting transport layer protocol (such as SSL or TLS) to transport CIFS protocol messages.

1.6.2 Performance

The use of message authentication causes the complete contents of each message to be hashed using MD5, which can decrease performance. Very high speed implementations of MD5 are available (20 megabytes/sec on a 166 mhz Pentium) that can minimize the impact.

1.7 Storing Passwords Securely

The passwords used in any of the authentication mechanisms used by this protocol have to be protected from access from over the network and from physical access. If the server does not support access control at the individual file level, but only at the file tree level, then password files can not be placed in a file tree that is accessible from the network, as all files in such a tree have to be at least equally readable.

References

[FIPS 81] DES, FIPS PUB xxx

[RFC 1320] RFC 1320, R. Rivest, The MD4 Message-Digest Algorithm

[RFC 1321] RFC 1321, R. Rivest, The MD5 Message-Digest Algorithm

[RFC 1828] RFC 1828, P. Metzger, W. Simpson, "IP Authentication using Keyed MD5", August 1995

{Kal 95] B. Kaliski, M.Robshaw, "Message Authentication with MD5", CryptoBytes, Sping 1995, RSA Inc, (http://www.rsa.com/rsalabs/pubs/cryptobytes/spring95/md5.htm)