# CIFS Authentication Protocol

Paul J. Leach

Microsoft

*Preliminary Draft – do not cite*

Author's draft: 4

This is a preliminary draft of a portion of specification of a proposed new version of the CIFS authentication protocol. It is supplied here as a standalone document for ease of review; if accepted and implemented, it may be incorporated into a future release of the CIFS specification. (This specification is subject to change without notice and should not be construed as a product commitment from Microsoft Corporation.)

The original protocol from which this version descends was designed more than a decade ago; recently, quite a few weaknesses have been found in previous versions. This latest revision is an attempt to repair those weaknesses with as small a change to the protocol as possible, so that it can be incrementally and rapidly deployed. In particular, it must not be necessary for all users to change their passwords to deploy the upgraded protocol, or to deploy new key server software. Also, efficiency is an issue, so some more robust MAC schemes that could have been used weren't.

This portion of the specification describes the authentication protocol abstracted from the implementation details, in order to make scrutiny of its security properties easier. It also only describes the strongest of several variants of the authentication protocol; a brief summary of the other variants is at the end of the document, together with a description of how the real protocols vary from this abstraction. The full details of the protocol are in the companion CIFS Authentication Protocols Specification document; a broader discussion of the security properties, including other attacks, may be found in the CIFS Security Considerations document.

## 1.1 Overview

Session authentication is done via a challenge response protocol based upon the shared knowledge of the user's password. Message authentication is done by attaching a message authentication code (MAC) to each message.

The response is computed by DES encrypting a challenge (a nonce) selected by the server with three keys derived from the user's password.

The MAC is a keyed-MD5 construction (see [RFC 1828]), using a key derived from the user's password and the client and server nonces. Each message is either of known fixed length or contains an explicit length, and is longer than an MD5 block, which avoids the known weaknesses of MD5 as a MAC (see [Kal 95]). Each message includes an implicit sequence number, to avoid replay.

We describe the authentication protocols as if the CIFS server communicates over some secure (private, authenticated) channel to a key server (KS) which keeps a database of hashes of clients' passwords, but a server might actually store the hashed passwords itself and be its own KS. Also, either type of server could store the passwords instead of a hash of the passwords. We consider these topics to be outside the scope of this protocol. One of the design goals for this version of the protocol was to leave the server to key server protocol and the hashed password format unchanged from the previous version.

## 1.2 Definitions

Let

| | |
|---|---|
| `U` | be the user's name, blank padded to 16 bytes |
| `P(U)` | be U's password |
| `Ks, Ks'` | be a 128 bit session key |

| | |
|---|---|
| Ka, Ka' | be a 56 bit DES key extracted from the first seven bytes of Ks |
| Kb, Kb' | be another 56 bit DES key extracted from the second seven bytes of Ks |
| Kc, Kc' | be another 56 bit DES key extracted from the last two bytes of Ks, padded with zeros |
| SN, SN' | be a 32 bit sequence number |
| Km, Km' | be a 40 byte key for a keyed-MD5 MAC |
| [s]<n:m> | be the "n" bytes of s starting at byte "m" (the first byte is numbered 0). |
| [s]<n> | be the first "n" bytes of s |
| a,b,…z | be the concatenation of the byte strings a, b, … z |
| { m }K | be the DES encryption [FIPS] of the byte string m with key K |
| MD4(m) | be the MD4 message digest [RFC 1320] of the byte string m |
| MD5(m) | be the MD5 message digest [RFC 1321] of the byte string m |
| Z(n) | be a byte string of zeros of length n |
| CS | be an 8 byte nonce chosen by the server, used as a challenge |

## 1.3 Application protocol messages

The application protocol being secured is a request/response protocol that has the following characteristics. Authentication is carried out in the process of setting up a session, during which session features are negotiated, the exact details of which do not affect the authentication protocol. Each of the elements of protocol messages is either a fixed length byte string, or contains an explicit length, and is at least 32 bytes long;, and requests are guaranteed to be distinguishable from responses.

| | |
|---|---|
| Mneg | be a session negotiation request containing supported features |
| Mnegr | be a negotiation response indicating selected features |
| Msess | be a session request |
| Msessr | be a session response |
| Mreq | be a subsequent protocol request |
| Mrsp | be a subsequent protocol response |

## 1.4 Session authentication protocol

1. The client computes the session keys from the user's password, initializes its sequence number, and sends a session negotiation request to the server.

```
C:          Ks  = MD4(P(U))
            Ka = [Ks]<7>
            Kb = [Ks]<7:7>
            Kc = [Ks]<2:14>, Z(5)

C->S:       Mneg
```

2. The server responds with the features negotiated, and a challenge:

```
S->C:       Mnegr, CS
```

3. The client computes a response to the challenge. It computes the MAC key, and the MAC of the message, and send the user name, challenge response, and session request parameters to the server. Its message uses a sequence number of 0, and it expects a sequence number of 1 to be used in the response.

```
C:          R = {CS}Ka, {CS}Kb, {CS}Kc
            Km = Ks, R
            SN = 0
            MC = [MD5(Km, SN, Msess, U, R)]<8>
```

```
              SN = 1

C->S:         Msess, U, R, MC
```

4. The server send the user's name, the challenge, and the response to a key server (KS) over a secure (private, authenticated) channel.

```
S->KS:        U, CS, R
```

5. The key server looks up the session key by looking up the user's name in a database containing the MD4 hash of users' passwords, and from the key and the client's challenge, computes the expected response. If the expected response matches the actual response (R == R'), then it sends Ks' to the server, otherwise it tells the server to deny access.

```
KS:           Ks'  = MD4(P(U))
              Ka' = [Ks']<7>
              Kb' = [Ks']<7:7>
              Kc' = [Ks']<2:14>, Z(5)
              R'  = {CS}Ka', {CS}Kb', {CS}Kc'

KS->S:        Ks'
```

6. The server computes the MAC key, and the MAC for the request. If MC' == MC, then the client has authenticated to the server. The server computes the MAC of its response, then sends the session acknowledgment message, then sets its sequence number to 2, the expected value in the next request.

```
S:            Km' = Ks', R
              MC' = [MD5(Km, SN', Msess, U, R)]<8>
              MS  = [MD5(Km', SN', Msessr)]<8>

S->C:         Msessr, MS

S:            SN' = 2
```

7. The client checks if MS' == MS; if so,  then the server has authenticated to the client, and the client's sequence number is set to the value to be used in the next request.

```
C:            MS' = [MD5(Km, SN, Msessr)]<8>
              SN  = 2
```

## 1.5 Message authentication protocol

For each request/response interaction thereafter the following procedure is used:

1. The client send the request together with a MAC of the request computed using the current sequence number, then bumps its sequence number to the one expected in the response:

```
C->S:         Mreq, [MD5(Km, SN, Mreq)]<8>
C:            SN = SN + 1
```

2. The server checks the MAC, and if correct, sends the response with a sequence number one higher, then bumps its sequence number by 2 to the expected value in the next request:

```
S->C:         Mrsp, [MD5(Km', SN'+1, Mrsp)]<8>

S:            SN' = SN' + 2
```

3. The client checks the MAC and if correct accepts the response and bumps its sequence number:

```
C:          SN  = SN  + 1
```

### 1.6 Summary of other variants and differences

There are variants of the authentication protocols; they exist for backwards compatibility. The variants are creating by taking certain allowed combinations of the following differences:

- The session key Ks is computed differently.
- The message authentication protocol is omitted.
- A plaintext password may be sent

The feature negotiation step (the exchange of Mneg and Mnegr above) is where the exact variant is selected. Both client and server can force the use of as strong a variant as they require to meet their security policy.

The actual authentication protocols differ from the one described in the following ways:

- The order of fields in messages may be different
- The MAC value is calculated by inserting the implicit sequence number into a field of the message, and computing the MAC; then that field is overwritten with the MAC value for transmission.
- Multiple requests or responses may be "batched" together into one message; single requests or responses may be spread out over multiple messages; and some requests have no response.

It is not believed that any of these differences affect the security of the protocol. The full details of the protocol are in the CIFS Authentication Specification document; a broader discussion of the security properties, including other attacks, may be found in the CIFS Security Considerations document.

### 1.7 References

[FIPS] DES, FIPS PUB 46-1, 1988.

[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)