# Another Look at Authentication

*Phil Karn, KA9Q*

*ABSTRACT*

A simple and effective technique for packet authentication in a datagram network is described that is based on the Data Encryption Standard (DES). In accordance with FCC rules, the actual data is not encrypted; rather DES is used to compute a special "cipher checksum" that is appended to the unencrypted user data before transmission. The recipient recomputes the cipher checksum and compares it against the incoming value, thereby detecting bogus or altered packets. This technique is potentially useful in a wide variety of amateur radio applications in addition to packet radio; for example, it could provide a secure control link for a remote repeater site.

## 1. Introduction

At the 5th ARRL Computer Networking Conference, Hal Feinstein described a technique for authenticating amateur packet transmissions [1] useful for protecting radio control links. In the commercial and military worlds, the solution is simple: just encrypt everything. Not only does this protect against unauthorized commands from those not knowing the key, it also hides the command information itself. In the amateur service, however, this is prohibited by FCC rules. [2] The only exception is for command links in the Amateur Satellite Service; [3] there is no corresponding exception for terrestrial links.

Fortunately, as Hal showed in his paper, a rule change is not necessary to add security legally to a control link. Cryptographic techniques can be used to add "authentication" to an unencrypted command to verify that it was sent by an authorized station. Hal assumed that a virtual circuit would be set up between the control station and the remote site, and as a result his technique is quite involved. I have devised a similar but much simpler approach made possible by the use of a datagram protocol.

## 2. The Data Encryption Standard

As in Hal's approach, I use the Data Encryption Standard (DES). [4,5] A full description is outside the scope of this paper; however, I will review two of its properties since they are important to the proposed authentication scheme.

1. The internals of DES are public knowledge, since the complete specifications have been widely published. Like a well-designed safe, however, knowing how DES works isn't much help in cracking it; only the key (or the combination) need be secret for good security. This is what allowed DES to become the first-ever cryptographic standard; similarly it allows us to establish an amateur authentication standard so that each system operator doesn't have to reimplement the wheel.
2. In practice, DES has been highly resistant to "known plaintext" attacks. [6,7,8,9,10] That is, even with a plaintext/ciphertext "matched pair", the algorithm is so nonlinear that at present there is no known way (outside of the NSA, at least) to find the key that produced the transformation other than by trying all possible $2^{56}$ keys in the algorithm until you find the one that works. As we will see, this property is very important to the authentication scheme described here, since each transmission contains both the ciphertext and the plaintext that produced it. It

should be pointed out, however, that as yet there is no published mathematical *proof* that a known-plaintext attack against DES requires an exhaustive search. Despite some "suspicious" structure in the algorithm that reduces the required brute-force effort somewhat, the accusation that a "trap door" may have been planted in the algorithm by its designers has yet to be either proven or disproven.

As an aside, it is this resistance to known plaintext attack that makes the M/A-Com Videocipher system (which encrypts the audio with DES) so hard to break. Anyone can buy a box, subscribe to a service and get millions of ciphertext/plaintext pairs. However, that doesn't help in finding out the DES key in use, which you would need to build a pirate decoder. The key is kept inside the decryption device in a battery-backed register which can't be read from the device pins. Only a physical attack is likely to work here, e.g., dissolving the epoxy off the chip with solvents and reading the key from the exposed chip with a scanning electron microscope of the type designed for debugging ICs under test. This illustrates an important practical point in cryptography: key security, not the mathematical strength of the encryption algorithm, is usually the weak link. Key security is harder than you might think, especially when many people are involved.

### 3. DES Modes of Operation

The basic DES algorithm transforms 64-bit data blocks between plaintext and ciphertext form under control of a 56-bit key. As long as the key is constant, enciphering a given 64-bit data block always gives the same 64-bit block of ciphertext. If you encrypt data directly in this way, you are using DES in the Electronic Code Book (ECB) mode. ECB can result in repeated patterns when, for example, strings of blanks are encrypted. To avoid this potential problem, several modes all involving feedback are recommended by the NBS. [11] One of these is "cipher block chaining" (CBC). In this mode, each 64-bit block of plaintext is exclusive-ored with the DES ciphertext output for the *last* 64-bit data block before being encrypted. (Since there is no preceding block of ciphertext the first block of plaintext is instead exclusive-ored with a prearranged "initialization vector"). The ciphertext at a given point in the message therefore depends on *all* of the data preceding it, not just the current block. Any change therefore "propagates" throughout the rest of the message.

### 4. Authentication With DES

This error-propagation property is the basis of the authentication scheme, which can now be described.

Encrypt each packet with DES in the cipher block chaining mode. Then send the original unencrypted packet along with the last cipher word (64 bits) of the encrypted version. The receiver also encrypts the packet and compares its final cipher word with the received version; if they match, the packet is accepted. Changing even one bit in the message results in a completely unpredictable change in the cipher checksum with only 1 chance in $2^{64}$ of escaping detection by the receiver.

This technique amounts to adding a "cipher checksum" to the packet, an apt description since it functions much like an ordinary checksum or CRC. The only difference is that the "checksum algorithm" must protect against corruption or spoofing by malevolent humans as well as by nature, and therefore must be more sophisticated.

For example, it would not be sufficient to generate the cipher checksum by computing a normal checksum over the packet and then encrypting it before transmission. A spoofer could carefully construct a packet to have the same checksum as a valid packet he had seen earlier on the channel, and then append the same cipher checksum. The size of the cipher checksum is also very important. Ordinary checksums, designed to protect only against random natural corruption, are often only 16 bits wide. It is not that impractical to try all 65,536 possible checksums until the correct one is found by chance. Since the cipher checksum produced by DES is 64 bits wide, trying all possible values is out of the question. Naturally, such a wide field also provides superior protection against corruption by natural causes.

## 5. Playback Attacks

Hal's paper discusses in detail the problem of the "playback attack". Even though the bad guy might not be able generate his own message or corrupt a real one without upsetting the authentication mechanism, he could record and play back a valid message at a later time in an attempt to repeat the same operation.

However, using a transport (level 4) protocol designed for a datagram network neatly solves this problem. Such protocols are already designed to detect and reject duplicate packets arriving minutes or even hours after the original. This protection is necessary because a datagram network will occasionally deliver a long-delayed duplicate of a packet, usually when its routing algorithm is trying to recover from a failing or congested link. For example, TCP [12] uses 32-bit sequence numbers, so over 4 gigabytes must be sent on a given connection before the sequence numbers wrap around. In most implementations, subsequent connections generally cycle through all possible port numbers on the originating end, and this adds another 16 bits to the "sequence space". Even when the exact same pair of port numbers is reused, a clock has incremented the initial sequence number for the connection fast enough that it is very unlikely to reuse the sequence numbers from the last incarnation of the connection. In short, a properly implemented TCP can go a *very* long time before it reuses the exact same combination of source address, destination address, source port, destination port, send sequence number and receive sequence number, long enough for the DES key to have been conveniently changed in the meantime!

Connectionless, transaction-oriented transport protocols (such as might be used for the remote control of a packet switch or repeater) can protect against duplicates in several ways: with sequence numbers as in a connection oriented protocol, with timestamping (rejecting packets older than some limit), or by designing the set of commands in an "idempotent" fashion, which means that receiving a command more than once causes no further change in the state of the system being controlled.

## 6. Implementation Issues

How could a cipher checksum be added to TCP/IP, and how much of the packet should it cover? If the authentication is to be end-to-end, it can't include the IP header, [13] since at least the time-to-live (TTL) field is modified by each IP packet switch. The data covered by the cipher checksum must be delivered unchanged for the cipher checksum to be valid; therefore it should cover only the data following the IP header. While this would seem to open up devious opportunities based on modifying the IP header, this really isn't a problem. TCP and UDP [14] incorporate "pseudo-headers" into their checksum algorithms that include the IP source and destination addresses, protocol type and data length. Changing any of these fields in the IP header would result in a checksum error when the packet reaches the destination, and of course any attempt to modify the checksum field in the TCP or UDP header would be detected by the cipher checksum.

Another question is, where should the cipher checksum go? It would be nice to find a place to put it that wouldn't make the resulting datagram incompatible with versions of TCP/IP that didn't understand it. This suggests placing it in one of the option fields, either in IP or TCP (assuming TCP is used). If it is put in the TCP options field, one runs afoul of the specification that says options should be present only in SYN segments (connection requests). New TCP options are also discouraged. This leaves the IP options field, of which there are relatively many, some of which are not understood by every implementation. To allow for this, option codes always include the length of the option, so that an unknown option type can be skipped over. Putting the cipher checksum into an IP option also allows it to be used with transport protocols other than TCP (e.g., UDP).

Other minor issues, such as padding and the choice of initialization vector, are easily resolved. Since the DES CBC mode operates on 8-byte blocks of data, data fields not a multiple of 8 bytes long should be padded out with zeros before encryption. The initialization vector (IV) required by the CBC algorithm could serve as an additional key element; it would be then necessary to know both the 56-bit DES key and the 64-bit IV in order to generate and check authenticators. For the sake of simplicity, however, the IV should probably be standardized (e.g., all zeroes) and only the DES key used for security.

## 7. Summary

Authentication is far easier to implement in a datagram network than in one based on virtual circuits, resulting in a much simpler and more elegant design. Since there is only one type of packet at the datagram level, there is only one type of authentication operation. There is no need for a "session key" or "challenge" at the beginning of a connection should one exist at a higher protocol layer. Each datagram is individually authenticated to protect it against spoofing or corruption, and eliminating the possibility of a bad guy taking over a connection (assuming one exists) after it has been established. The measures already in place to protect against the accidental packet duplication possible in a datagram network automatically guard against playback attacks.

A public-domain implementation of DES in C is available from the author.

## 8. References


1. Feinstein, Hal, WB3KDU, "Authentication of the Packet Radio Switch Control Link", *Proceedings of the ARRL Amateur Radio 5th Computer Networking Conference*, p 5.12.
2. Federal Communication Commission rules, Section 97.117 (Codes and Ciphers Prohibited).
3. Ibid, section 97.42 1 (a) (Telecommand Operation).
4. Federal Information Processing Standards Publication 46, "Data Encryption Standard", January 15, 1977.
5 American National Standards Institute, "American National Standard Data Encryption Algorithm", ANSI X3.92-1981.
6. Davio, et al, "Analytical Characteristics of the DES", *Crypto '83*, Santa Barbara.
7 Sugarman, "On Foiling Computer Crime", *ZEEE Spectrum,* July 1979.
8. Davies, "Some Regular Properties of the 'Data Encryption Standard' Algorithm", National. Physical Laboratory, Teddington, Middlesex, UK.
9. Lexar Corporation, "An Evaluation of the NBS Data Encryption Standard".
10. Branstad et al, "Report of the Workshop on Cryptography in Support of Computer Security", National Bureau of Standards report NBSIR 77-1291.
11. Federal Information Processing Standards Publication, "Announcing the Standard for DES Modes of Operation".
12. Postel, ed, "Transmission Control Protocol Specification", ARPA RFC 793.
13. Postel, ed, "Internet Protocol Specification", ARPA RFC 791.
14. Postel, ed, "User Datagram Protocol", ARPA RFC 768.