

A Simple Method for Authenticating Radio Messages

Introduction

Because bad actors (those who would willfully interfere with communications, in this case by impersonating others) are a real possibility, being sure of the integrity of a message and the authenticity of the sender are both crucial for emergency communications. I propose herein a simple method for doing both with reasonable surety.

Integrity and Authenticity

Message integrity refers to the successful transmission of the message so that both sides have the exact same information. Message authenticity means that the receiving side is reasonably certain that the message comes from the presumed source.

Integrity: Solved with Cyclic Redundancy Check

One of the oldest methods of verifying the integrity of a digital transmission is the Cyclic Redundancy Check (CRC). It is computationally inexpensive, easily understood and available, and works well. It is calculated by the sender, sent and received, then re-calculated (using the same algorithm) and compared by the receiver (see Appendix).

Authenticity: Solved with Pre-Exchanged Private Information

To authenticate a sender with reasonable surety, parties must have a previously agreed upon exchange that identifies them. An ancient example of this is a code word. Parties exchange a code word that they both already know, and they are authenticated.

A code word only works when it can be shared and compared in private. In a radio message, a code word is immediately compromised and is only effective once. A One Time Pad (OTP) solves this problem, but requires the constant private (non-radio) exchange of new codes, and coordination to ensure that codes aren't re-used or misaligned.

An Alternative: Codeplug Modifiable CRC

Privacy is not required for a CRC to work, when the CRC is being used to validate integrity. However, if you could generate a CRC with a modifiable algorithm, which is altered by a private codeplug (analogous to a code word), you could produce a CRC that both parties could validate. This would provide both integrity and authentication. Since the codeplug would not be shared over the air, you would not need to alter the codeplug frequently, as with an OTP solution

Proposed Codeplug Format

Rather than alter the established math in the CRC algorithm itself, the output CRC could be modified by altering the input on both sides, before the CRC is generated, with a private codeplug. A strong alteration is not needed but a simple codeplug is, and so a rotation encoding is suggested. Each letter of

the message could be rotated by +/- n, as defined in the codeplug. The following simple format is proposed: <+ or -><00-99>, repeated a minimum of ten times. For easy use in all digital radio modes, as well as easy phonetic communication, the three character CRC would be converted to fit into 36 letters and numbers (A-Z0-9).

Example Use Case

As an example, take the codeplug "+03+12-42+18-06-94+20+67-17+00" and encode it against the message "SITREP 113114NN REGION SECURE". Iterate through each letter of the message, and rotate that letter by the corresponding portion of the codeplug ("S" would rotate +3 to "V", etc), looping the codeplug as needed. After encoding, you would run a standard CRC on the encoded version of the message. You would transmit over the air the un-encoded original message, and the CRC (see appendix diagram).

Cracking a Codeplug With Only the Message and CRC Output

To crack a private codeplug using only the transmitted CRC and message, a bad actor would need at least two transmissions that used the same codeplug (due to the fact that the same 36-possibility three-character CRC could be generated by multiple codeplugs). If they intercepted transmissions from the same two people during a short window of time, or from two people in the same group, etc., they could assume this may be the case, but not guarantee it. With a ten segment codeplug and an ASCII subset of 95 characters (32 through 126), we have 95^{10} , or 59 quintillion (59 billion billion) possible combinations. Given the fact that the codeplug length is not enforced means that number may be even higher. These numbers may not be too difficult for governments to deal with, but should be sufficient to deter bad actors in amateur radio.

Increasing Surety of Authentication

To increase the possible combinations, and therefore increase the likelihood of reasonable surety in authentication, operators could work with sets of codeplugs of varied lengths. For example, a group might have ten codeplugs of random length between ten and fifteen segments each. Starting on the first of the month they could rotate through the ten codeplugs based on the day. This would vastly increase the number of possible combinations a bad actor would need to test to become a nuisance, and add a secondary validation (the right codeplug on the right day).

Operators would want to avoid short messages (which would only use a portion of the codeplug, reducing the computing power needed to guess a portion of the codeplug).

Operators would want to avoid codeplugs that contain identifiable or guessable information (such as birth dates, dates in general, addresses, etc.).

Special one-time-use codeplugs could be reserved for challenging suspicious or high-priority / high-value transmissions.

Operators could include the to and from callsigns in the message before the CRC check, which would provide further surety for those two vital pieces of information, and which would increase the message length and use of more of the codeplug. This is done automatically in the Message Authenticator software.

Where Codeplug Modifiable CRC Could Be Used

A codeplug modifiable CRC requires a fair amount of math, and should be generated and validated on a computer. To do so would take a small piece of software such as the Message Authenticator that is included with this document, and which is available on <https://www.kf7mix.com>. Additional training, insights and information are available on <https://sitrepnet.com/>

Even though a computer is used to generate and test the modifiable CRC, the resulting three-character code could be easily used to validate radio messages shared via digital modes such as JS8Call, PSK31, MFSK16, etc., as well as on messages shared via normal voice communications, in emails, or shared in any other way. In all cases, operators would need to understand exactly what the message text format is when signed, so that they could validate it exactly as it was used to create the original CRC.

Appendix

Diagram of a Standard CRC Check Interaction

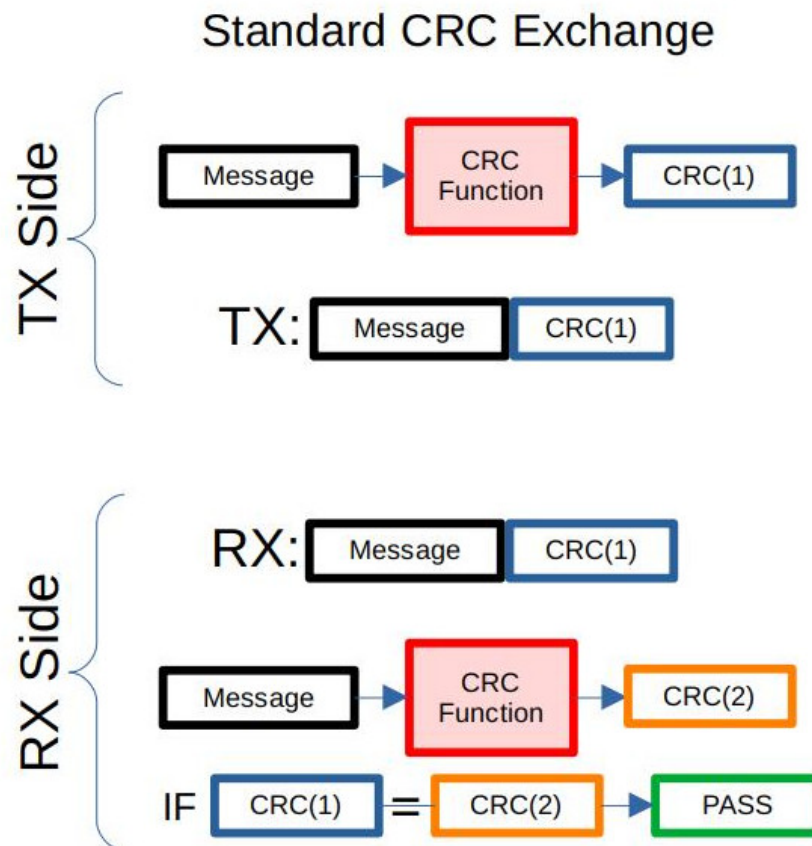


Diagram of a Codeplug Modified CRC interaction

