Cryptographic Standards in HOL

Aubin Whitley

March 19, 2025

Abstract

In this set of theories, we express well-known crytographic standards in the language of Isabelle. The standards we have translated so far are:

FIPS 180-4	NIST's Secure Hash Standard, rev 4
FIPS 186-4	Only the elliptic curves over prime fields, i.e. NIST's "P-" curves
FIPS 198-1	NIST's The Keyed-Hash Message Authentication Code (HMAC Standard)
PKCS #1 v2.2	RSA Laboratories' RSA Cryptography Standard, version 2.2
SEC1 v2.0	SECG's Elliptic Curve Cryptography, version 2.0

The intention is that these translations will be used to prove that any particular implementation matches the relevant standard. With that in mind, the overriding principle is to adhere as closely as possible, given the syntax of HOL, to the written standard. It should be obvious to any reader, regardless of their past experience with Isabelle, that these translations exactly match the standards. Thus we use the same function and variable names as in the written standards whenever possible and explain in the comments the few times when that is not possible.

We want the users of these theories to have faith that errors were not made in the translations. We do two things to achieve this. First, in addition to translating a standard, we provide a robust supporting theory that proves anything that one might wish to know about the primitives that the standard defines. For example, we prove that encryption and decryption are inverse operations. We prove when RSA keys are equivalent. We prove that if a message is signed, then that signature and message will pass the verification operation. Any fact that you may need in using these standards, we hope and believe we have already proved for you.

Second, we prove (by evaluation) that the test vectors provided by NIST, et al, check as intended (whether a passing or failing test case.) The test vectors may be found in theories named *_Test_Vectors.thy. These files can be large and take time for Isabelle to process. Thus, they are not imported by this main Crypto_Standards theory. Users may open those separately. As an aside, Isabelle must be told how to compute certain operations efficiently. For example, modular exponentiation or scalar multiplication of a point on an elliptic curve. The Efficient*.thy files are necessary to check the test vectors.

We attempt to be as agnostic to implementation details as possible. For example, we do not assume any particular data type has been used as input or output. Many standards operate on octets, or 8-bit values. For these theories, an octet string is modeled as a list of natural numbers. Then a nat x is a "valid octet" exactly when $x < 256$. Words.thy contains all the operations needed to convert natural number to a string of n-bit values and back to a natural number. There you will also find definitions for handling padding and bit manipulations that are found in the above standards. Again, we believe that we have proved anything you may need to apply these theories. We have erred on the side of including lemmas that may be of practical use as opposed to proving a minimal set of lemmas required.