Kernel Crypto API Interface Specification

Introduction

The kernel crypto API offers a rich set of cryptographic ciphers as well as other data transformation mechanisms and methods to invoke these. This document contains a description of the API and provides example code.

To understand and properly use the kernel crypto API a brief explanation of its structure is given. Based on the architecture, the API can be separated into different components. Following the architecture specification, hints to developers of ciphers are provided. Pointers to the API function call documentation are given at the end.

The kernel crypto API refers to all algorithms as "transformations". Therefore, a cipher handle variable usually has the name "tfm". Besides cryptographic operations, the kernel crypto API also knows compression transformations and handles them the same way as ciphers.

The kernel crypto API serves the following entity types:

- consumers requesting cryptographic services
- data transformation implementations (typically ciphers) that can be called by consumers using the kernel crypto API

This specification is intended for consumers of the kernel crypto API as well as for developers implementing ciphers. This API specification, however, does not discuss all API calls available to data transformation implementations (i.e. implementations of ciphers and other transformations (such as CRC or even compression algorithms) that can register with the kernel crypto API).

Note: The terms "transformation" and cipher algorithm are used interchangeably.

Terminology

The transformation implementation is an actual code or interface to hardware which implements a certain transformation with precisely defined behavior.

The transformation object (TFM) is an instance of a transformation implementation. There can be multiple transformation objects associated with a single transformation implementation. Each of those transformation objects is held by a crypto API consumer or another transformation. Transformation object is allocated when a crypto API consumer requests a transformation implementation. The consumer is then provided with a structure, which contains a transformation object (TFM).

The structure that contains transformation objects may also be referred to as a "cipher handle". Such a cipher handle is always subject to the following phases that are reflected in the API calls applicable to such a cipher handle:

1. Initialization of a cipher handle.
2. Execution of all intended cipher operations applicable for the handle where the cipher handle must be furnished to every API call.
3. Destruction of a cipher handle.

When using the initialization API calls, a cipher handle is created and returned to the consumer. Therefore, please refer to all initialization API calls that refer to the data structure type a consumer is expected to receive and subsequently to use. The initialization API calls have all the same naming conventions of crypto_alloc*.

The transformation context is private data associated with the transformation object.