1 Introduction

In 1972 and 1974, the National Bureau of Standards (now the National Institute of Standards and Technology, or NIST) issued the first public request for an encryption standard. The result was DES [NBS77], arguably the most widely used and successful encryption algorithm in the world.

Despite its popularity, DES has been plagued with controversy. Some cryptographers objected to the "closed-door" design process of the algorithm. The debate about whether DES' key is too short for acceptable commercial security has raged for many years [DH79], but recent advances in distributed key search techniques have left no doubt in anyone's mind that its key is simply too short for today's security applications [Wie94, BDR+96]. Triple-DES has emerged as an interim solution in many high-security applications, such as banking, but it is too slow for some uses. More fundamentally, the 64-bit block length shared by DES and most other well-known ciphers opens it up to attacks when large amounts of data are encrypted under the same key.

In response to a growing desire to replace DES, NIST announced the Advanced Encryption Standard (AES) program in 1997 [NIST97a]. NIST solicited comments from the public on the proposed standard, and eventually issued a call for algorithms to satisfy the standard [NIST97b]. The intention is for NIST to make all submissions public and eventually, through a process of public review and comment, choose a new encryption standard to replace DES.

NIST's call requested a block cipher. Block ciphers can be used to design stream ciphers with a variety of synchronization and error extension properties, one-way hash functions, message authentication codes, and pseudo-random number generators. Because of this flexibility, they are the workhorse of modern cryptography.

NIST specified several other design criteria: a longer key length, larger block size, faster speed, and greater flexibility. While no single algorithm can be optimized for all needs, NIST intends AES to become the standard symmetric algorithm of the next decade.

Twofish is our submission to the AES selection process. It meets all the required NIST criteria—128-bit block; 128-, 192-, and 256-bit key; efficient on various platforms; etc.—and some strenuous design requirements, performance as well as cryptographic, of our own.

Twofish can:

- Encrypt data at 285 clock cycles per block on a Pentium Pro, after a 12700 clock-cycle key setup.
- Encrypt data at 860 clock cycles per block on a Pentium Pro, after a 1250 clock-cycle key setup.
- Encrypt data at 26500 clock cycles per block on a 6805 smart card, after a 1750 clock-cycle key setup.

This paper is organized as follows: Section 2 discusses our design goals for Twofish. Section 3 describes the building blocks and general design of the cipher. Section 4 defines the cipher. Section 5 discusses the performance of Twofish. Section 6 talks about the design philosophy that we used. In Section 7 we describe the design process, and why the various choices were made. Section 8 contains our best cryptanalysis of Twofish. In Section 9 we discuss the possibility of trapdoors in the cipher. Section 10 compares Twofish with some other ciphers. Section 11 discusses various modes of using Twofish, including a family-key variant. Section 12 contains historical remarks, and Section 13 our conclusions and directions for future analysis.

2 Twofish Design Goals

Twofish was designed to meet NIST's design criteria for AES [NIST97b]. Specifically, they are:

- A 128-bit symmetric block cipher.
- Key lengths of 128 bits, 192 bits, and 256 bits.
- No weak keys.
- Efficiency, both on the Intel Pentium Pro and other software and hardware platforms.
- Flexible design: e.g., accept additional key lengths; be implementable on a wide variety of platforms and applications; and be suitable for a stream cipher, hash function, and MAC.
- Simple design, both to facilitate ease of analysis and ease of implementation.

Additionally, we imposed the following performance criteria on our design:

- Accept any key length up to 256 bits.

- Encrypt data in less than 500 clock cycles per block on an Intel Pentium, Pentium Pro, and Pentium II, for a fully optimized version of the algorithm.

- Be capable of setting up a 128-bit key (for optimal encryption speed) in less than the time required to encrypt 32 blocks on a Pentium, Pentium Pro, and Pentium II.

- Encrypt data in less than 5000 clock cycles per block on a Pentium, Pentium Pro, and Pentium II with no key setup time.

- Not contain any operations that make it inefficient on other 32-bit microprocessors.

- Not contain any operations that make it inefficient on 8-bit and 16-bit microprocessors.

- Not contain any operations that reduce its efficiency on proposed 64-bit microprocessors; e.g., Merced.

- Not include any elements that make it inefficient in hardware.

- Have a variety of performance tradeoffs with respect to the key schedule.

- Encrypt data in less than less than 10 milliseconds on a commodity 8-bit microprocessor.

- Be implementable on a 8-bit microprocessor with only 64 bytes of RAM.

- Be implementable in hardware using less than 20,000 gates.

Our cryptographic goals were as follows:

- 16-round Twofish (without whitening) should have no chosen-plaintext attack requiring fewer than 2^{80} chosen plaintexts and less than 2^N time, where N is the key length.

- 12-round Twofish (without whitening) should have no related-key attack requiring fewer than 2^{64} chosen plaintexts, and less than $2^{N/2}$ time, where N is the key length.

Finally, we imposed the following flexibility goals:

- Have variants with a variable number of rounds.

- Have a key schedule that can be precomputed for maximum speed, or computed on-the-fly for maximum agility and minimum memory requirements. Additionally, it should be suitable for dedicated hardware applications: e.g., no large tables.

- Be suitable as a stream cipher, one-way hash function, MAC, and pseudo-random number generator, using well-understood construction methods.

- Have a family-key variant to allow for different, non-interoperable, versions of the cipher.

We feel we have met all of these goals in the design of Twofish.

3 Twofish Building Blocks

3.1 Feistel Networks

A *Feistel network* is a general method of transforming any function (usually called the F function) into a permutation. It was invented by Horst Feistel [FNS75] in his design of Lucifer [Fei73], and popularized by DES [NBS77]. It is the basis of most block ciphers published since then, including FEAL [SM88], GOST [GOST89], Khufu and Khafre [Mer91], LOKI [BPS90, BKPS93], CAST-128 [Ada97a], Blowfish [Sch94], and RC5 [Riv95].

The fundamental building block of a Feistel network is the F function: a key-dependent mapping of an input string onto an output string. An F function is always non-linear and possibly non-surjective[1]:

$$F : \{0,1\}^{n/2} \times \{0,1\}^N \mapsto \{0,1\}^{n/2}$$

where n is the block size of the Feistel Network, and F is a function taking $n/2$ bits of the block and N bits of a key as input, and producing an output of length $n/2$ bits. In each round, the "source block" is the input to F, and the output of F is XORed with the "target block," after which these two blocks swap places for the next round. The idea here is to take an F function, which may be a weak encryption algorithm when taken by itself, and repeatedly iterate it to create a strong encryption algorithm.

Two rounds of a Feistel network is called a "cycle" [SK96]. In one cycle, every bit of the text block has been modified once.[2]

[1] A non-surjective F function is one in which not all outputs in the output space can occur.

[2] The notion of a cycle allows Feistel networks to be compared with unbalanced Feistel networks [SK96, ZMI90] such as MacGuffin [BS95] (cryptanalyzed in [RP95a]) and Bear/Lion [AB96b], and with SP-networks (also called uniform transformation structures [Fei73]) such as IDEA, SAFER, and Shark [RDP+96] (see also [YTH96]). Thus, 8-cycle (8-round) IDEA is comparable to 8-cycle (16-round) DES and 8-cycle (32-round) Skipjack.

Twofish is a 16-round Feistel network with a bijective F function.

3.2 S-boxes

An S-box is a table-driven non-linear substitution operation used in most block ciphers. S-boxes vary in both input size and output size, and can be created either randomly or algorithmically. S-boxes were first used in Lucifer, then DES, and afterwards in most encryption algorithms.

Twofish uses four different, bijective, key-dependent, 8-by-8-bit S-boxes. These S-boxes are built using two fixed 8-by-8-bit permutations and key material.

3.3 MDS Matrices

A maximum distance separable (MDS) code over a field is a linear mapping from a field elements to b field elements, producing a composite vector of $a+b$ elements, with the property that the minimum number of non-zero elements in any non-zero vector is at least $b+1$ [MS77]. Put another way, the "distance" (i.e., the number of elements that differ) between any two distinct vectors produced by the MDS mapping is at least $b+1$. It can easily be shown that no mapping can have a larger minimum distance between two distinct vectors, hence the term maximum distance separable. MDS mappings can be represented by an MDS matrix consisting of $a \times b$ elements. Reed-Solomon (RS) error-correcting codes are known to be MDS. A necessary and sufficient condition for an $a \times b$ matrix to be MDS is that all possible square submatrices, obtained by discarding rows or columns, are non-singular.

Serge Vaudenay first proposed MDS matrices as a cipher design element [Vau95]. Shark [RDP+96] and Square [DKR97] use MDS matrices (see also [YMT97]), although we first saw the construction used in the unpublished cipher Manta[3] [Fer96]. Twofish uses a single 4-by-4 MDS matrix over GF(2^8).

3.4 Pseudo-Hadamard Transforms

A pseudo-Hadamard transform (PHT) is a simple mixing operation that runs quickly in software. Given two inputs, a and b, the 32-bit PHT is defined as:

$$\begin{aligned} a' &= a + b \bmod 2^{32} \\ b' &= a + 2b \bmod 2^{32} \end{aligned}$$

SAFER [Mas94] uses 8-bit PHTs extensively for diffusion. Twofish uses a 32-bit PHT to mix the outputs from its two parallel 32-bit g functions. This PHT can be executed in two opcodes on most modern microprocessors, including the Pentium family.

3.5 Whitening

Whitening, the technique of XORing key material before the first round and after the last round, was used by Merkle in Khufu/Khafre, and independently invented by Rivest for DES-X [KR96]. In [KR96], it was shown that whitening substantially increases the difficulty of keysearch attacks against the remainder of the cipher. In our attacks on reduced-round Twofish variants, we discovered that whitening substantially increased the difficulty of attacking the cipher, by hiding from an attacker the specific inputs to the first and last rounds' F functions.

Twofish XORs 128 bits of subkey before the first Feistel round, and another 128 bits after the last Feistel round. These subkeys are calculated in the same manner as the round subkeys, but are not used anywhere else in the cipher.

3.6 Key Schedule

The key schedule is the means by which the key bits are turned into round keys that the cipher can use. Twofish needs a lot of key material, and has a complicated key schedule. To facilitate analysis, the key schedule uses the same primitives as the round function.

4 Twofish

Figure 1 shows an overview of the Twofish block cipher. Twofish uses a 16-round Feistel-like structure with additional whitening of the input and output. The only non-Feistel elements are the 1-bit rotates. The rotations can be moved into the F function to create a pure Feistel structure, but this requires an additional rotation of the words just before the output whitening step.

The plaintext is split into four 32-bit words. In the input whitening step, these are XORed with four key words. This is followed by sixteen rounds. In each

[3] Manta is a block cipher with a large block size and an emphasis on long-term security rather than speed. It uses an SP-like network with DES as the S-boxes and MDS matrices for the permutations.