(This document is being submitted electronically to AESFirstRound@nist.gov)

Date: 14 April 1999

From: Charles S. Williams
Chief Scientist
Cylink Corporation
910 Hermosa Court
PO Box 3759
Sunnyvale CA 94088-3759
Tel: (408) 328-5540

To: Information Technology Laboratory
Attn: AES Candidate Comments, Room 540
National Institute of Standards and Technology
100 Bureau Drive, STOP 8970
Gaithersburg, MD 20899-8970

Subj: Comments on Cylink's AES Candidate Algorithm **SAFER+**

Outline of Comments

1. Introduction

Since our original submission of SAFER+ as a candidate algorithm for the Advanced Encryption Standard (AES), Cylink Corporation has made substantial improvements in its implementations of this algorithm. This has resulted in a many fold speed increase in both software and hardware implementations. In Section 2, we describe the performance of our improved ANSI C implementation on the Pentium Pro (200 MHz) platform and on the Sun II (300 MHz) platform. Section 3 gives the performance of our improved Java implementation on a Microsoft Java Virtual Machine. Our improved hardware implementation is described in Section 4.

At the Second Advanced Encryption Standard (AES) Candidate Conference in Rome, Italy, March 22-23, 1999, J. Kelsey [1] described an "academic weakness" in the 192 and 256 bit key schedules for SAFER+ that allow a key search to be performed somewhat faster than by exhaustive search. In Section 5, we describe a modification of the SAFER+ key schedule that removes this "weakness", but does not affect the 128 bit key schedule. This modification was described by J. L. Massey in his presentation in the Algorithm Submitter Rebuttals and Discussion session at the Second Advanced Encryption Standard (AES) Candidate Conference.

Section 6 indicates how interested parties can obtain copies of our reports detailing the immunity of SAFER+ to differential cryptanalysis and linear cryptanalysis.

Section 7 describes the program files of Cylink's new software implementations that are being sent directly to NIST. Because of export considerations, these files are not made a part of this public comment.

2. ANSI C Implementation

We have made major improvements in our standard ANSI C implementation of SAFER+. The ANSI C programs described in Section 7 achieve the performances listed below on the indicated platforms with a 128 bit key. It is important to note that these figures are for single block encryption (as opposed to pipelined encryption of several blocks) and hence apply whether ECB mode or CBC mode is used for encryption. It should also be mentioned that a speed increase of at least 50% could be achieved on a 64-bit processor for an ANSI C implementation similar to that developed here for 32-bit processors.

2.1 Pentium Pro (200 MHz)

Number of clock cycles for encryption **= 859**
Encryption (or decryption) speed = **29 Mbit/s**

These figures should be compared to the 2095 clock cycles (or 12.2 Mbit/s) reported on Transparency 16 by NIST [2] and based on the implementation in our original submission.

[Remark: In our implementation using "extended ANSI C" that allows cyclic rotationcommands and is is accepted by the C compiler in the Microsoft Visual C++ 5.0 (or higher) system, the number of clock cycles reduces to **751** and the encryption speed increases to **33 Mbit/s**.]

2.2 Sun UltraSPARC II (300 MHz)

Number of clock cycles for encryption **= 810**
Encryption (or decryption) speed = **45 Mbit/s**

This speed figure should be compared to the 10.2 Mbit/s reported on Transparency 21 by NIST [2] and based on the ANSI C implementation in our original submission.

3. Microsoft Java Virtual Machine Implementation

We have made major improvements in our Java implementation of SAFER+. The Java programs described in Section 6 achieve the performance listed below for the Microsoft Java Virtual Machine on the Pentium Pro (200 MHz) platform with a 128 bit key.

Encryption (or decryption) speed = **11 Mbit/s**

This speed figure should be compared to the 2.6 Mbit/s reported on Transparency 29 by NIST [2] and based on the Java implementation in our original submission.

4. Hardware Implementation

We have made even greater improvements in our hardware implementation of SAFER+ than in our software implementations. For 0.25 micron CMOS cell based logic technology, our new hardware implementation with a system clock rate of 44 MHz requires only 181 nanoseconds to encrypt or decrypt a 128 bit block using a 128 bit key. This translates to an encryption / decryption rate of **704 Mbit/s** in either ECB or CBC mode. This figure should be compared with the 58.9 Mbit/s given in our submission document.

5. Key Schedules for 192 and 256 Bit Keys

Figures 1 and 2 of this comment diagram the new SAFER+ Unified Key Schedule that was described by J. L. Massey in his presentation in the Algorithm Submitter Rebuttals and Discussion session at the Second Advanced Encryption

Standard (AES) Candidate Conference. This key schedule removes the "weakness", which was mentioned in Section 1 above, in the 192 bit and 256 bit key schedules of our original submission. We use the description "unified" to reflect the fact that the 256 bit key schedule can also be used to perform either the 128 bit key schedule or the 192 bit key schedule simply by proper choice of the last 128 bits or last 64 bits, respectively, of the 256 bit user-selected key. Similarly, the 192 bit key schedule can also be used to perform the 128 bit key schedule simply by proper choice of the last 64 bits of the 192 bit user-selected key. The salient properties of this new schedule are the following:

- **Every byte of the user-selected key is used in every round.** (This is the property that removes the weakness described in Section 1.)

- **For the 128-bit key, the new key schedule gives exactly the same subkeys as the key schedule in our submission proposal.**

- For the 192-bit key, if bytes 17, 18, … 24 are chosen equal to bytes 1, 2, … 8, then the result reduces to the 128-bit key schedule.

- For the 256-bit key,
  if bytes 17, 18, … 32 are chosen equal to bytes 1, 2, … 16,
  then the result reduces to the 128-bit key schedule.

  If bytes 15, 26, … 32 are chosen equal to the byte-by-byte modulo-two sum of (1) bytes 1, 2, … 8; (2) bytes 9, 10, … 16; and (3) bytes 17, 18, … 24, then the result reduces to the 192-bit key schedule.

The design principles of the new SAFER+ Unified Key Schedule key schedule are identical to those used in the key schedule for SAFER SK-128, which has a 64 bit block size and 128 bit key, in the previous SAFER family of ciphers.

If SAFER+ is selected for the second round of the AES competition, we will submit the new SAFER+ Unified Key Schedule key schedule to NIST as a "tweak" on the algorithm.

6. Security against Differential and Linear Cryptanalysis

It was stated in our submission document that exhaustive cryptanalyses had shown that SAFER+ with 6 or more rounds is secure against differential cryptanalysis and that SAFER+ with 2.5 or more rounds is secure against linear cryptanalysis. Those interested in receiving copies of these cryptanalyses can obtain them by request to Dr. Gurgen Khachatrian (gurgenkh@forof.sci.am) or Prof. James L. Massey (101767.233@compuserve.com).

7. Files Sent Separately to NIST

C Program Package:
main.c
AES_SP.c
AES_SP.h
(ANSI C Programs:)
ANSI_SP.c
ANSI_SP.h
("Extended" ANSI C Programs:)
EXT_SP.c
EXT_SP.h

Java Package:
Safer_Algorithm
Safer_Properties
Safer_SecretKey
SaferPlus

References

[1] J. Kelsey, "Key Schedule Weaknesses in SAFER+", Second Advanced Encryption Standard (AES) Candidate Conference, Rome, Italy, March 22-23, 1999.

[2] "NIST's Efficiency Testing for Round1 AES Candidates", Second Advanced Encryption Standard (AES) Candidate Conference, Rome, Italy, March 22-23, 1999.

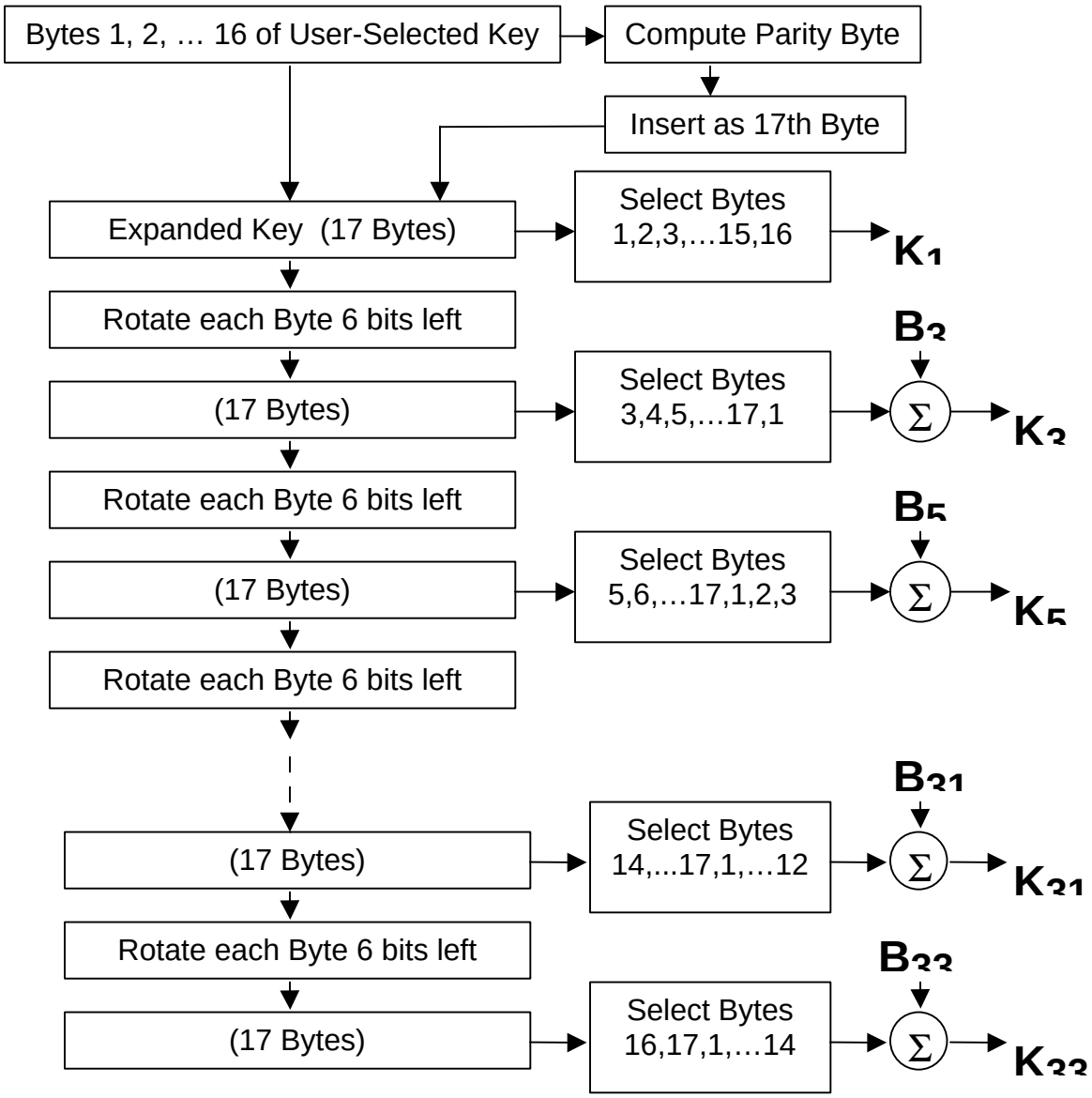

Fig. 1: SAFER+ Unified Key Schedule: Treatment of first 16 User-Selected Key Bytes
(Σ denotes bytewise mod 256 addition.)

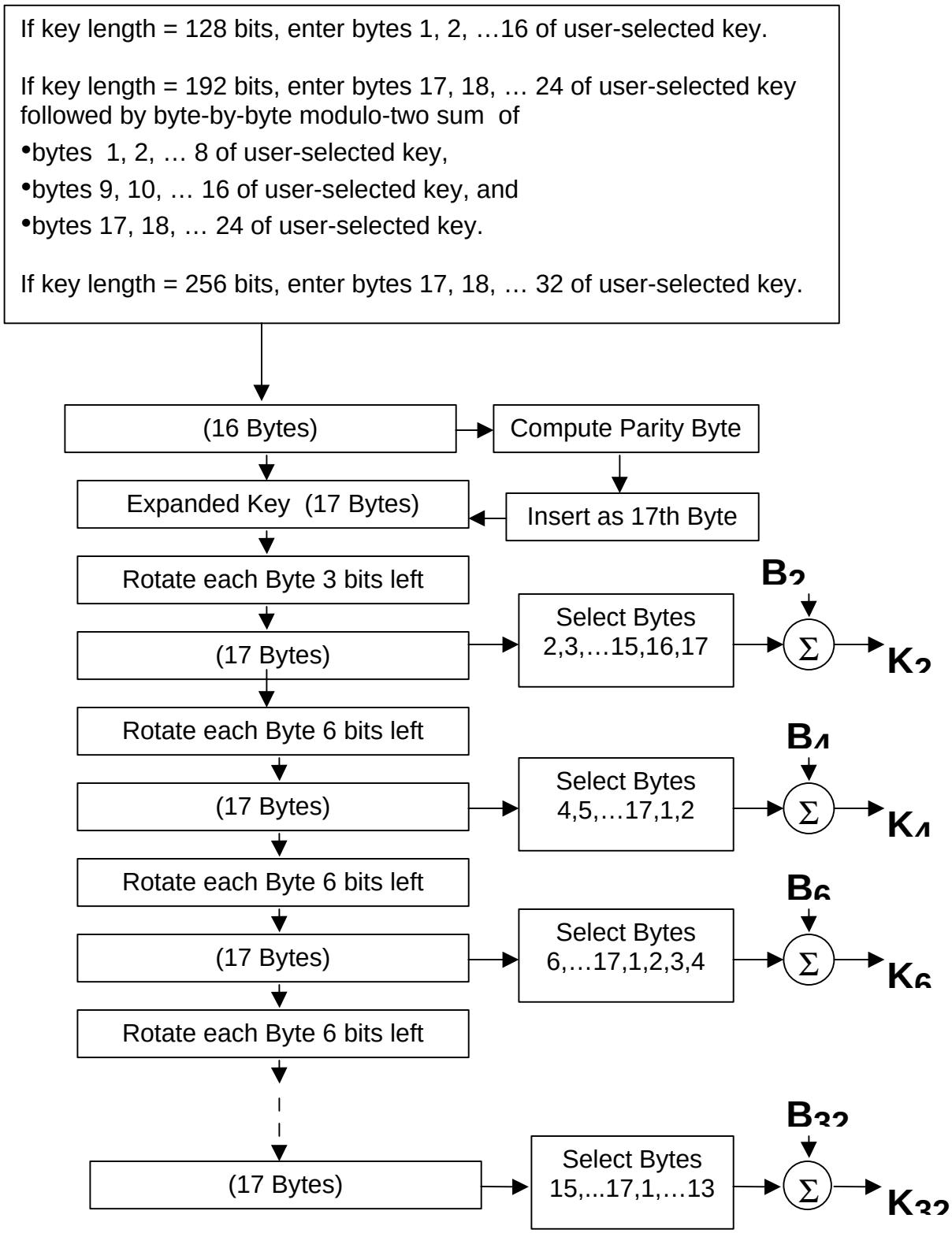

Fig. 2: SAFER+ Unified Key Schedule: Treatment of Remaining User-Selected Key Bytes
(Σ denotes bytewise mod 256 addition.)