AES & FLAM®

(Frankenstein-Limes-Access-Method)

Der Advanced Encryption Standard (AES) zur Verschlüsselung
der Advanced Data Compression (ADC) ab FLAM® V4.0

Inhaltsverzeichnis

Abbildungsverzeichnis

1 AES-Kryptographie ab FLAM® V4.0

1.1 Einleitung

Der 'Advanced Encryption Standard' (AES) löst den in die Jahre gekommenen 'Data Encryption Standard' (DES) ab. Dieser moderne symmetrische Blockalgorithmus bildet die Basis für die kryptographische Absicherung einer FLAMFILE® ab FLAM® 4.0. Er ist gegenüber DES wesentlich sicherer und benötigt gleichzeitig nur ein Zehntel der Rechenzeit. Dies - in Verbindung mit der ADC-Komprimierung - macht es möglich, starke Kryptographie auf große Datenmengen anzuwenden.

Anmerkung: Aus Sicherheitsgründen wurde DES schon seit Jahren nur noch in der Dreifachkombination *Triple DES* angewendet, was den ohnehin schon hohen Verbrauch an CPU-Zeit verdreifacht hatte.

In FLAM® wird AES mit einer Block- und Schlüssellänge von jeweils 128 Bits (16 Bytes) eingesetzt. Die folgende Abbildung zeigt die beiden Basisoperationen (AES-Schedule und AES-Encryption), welche immer wieder in Kombination zum Einsatz kommen.

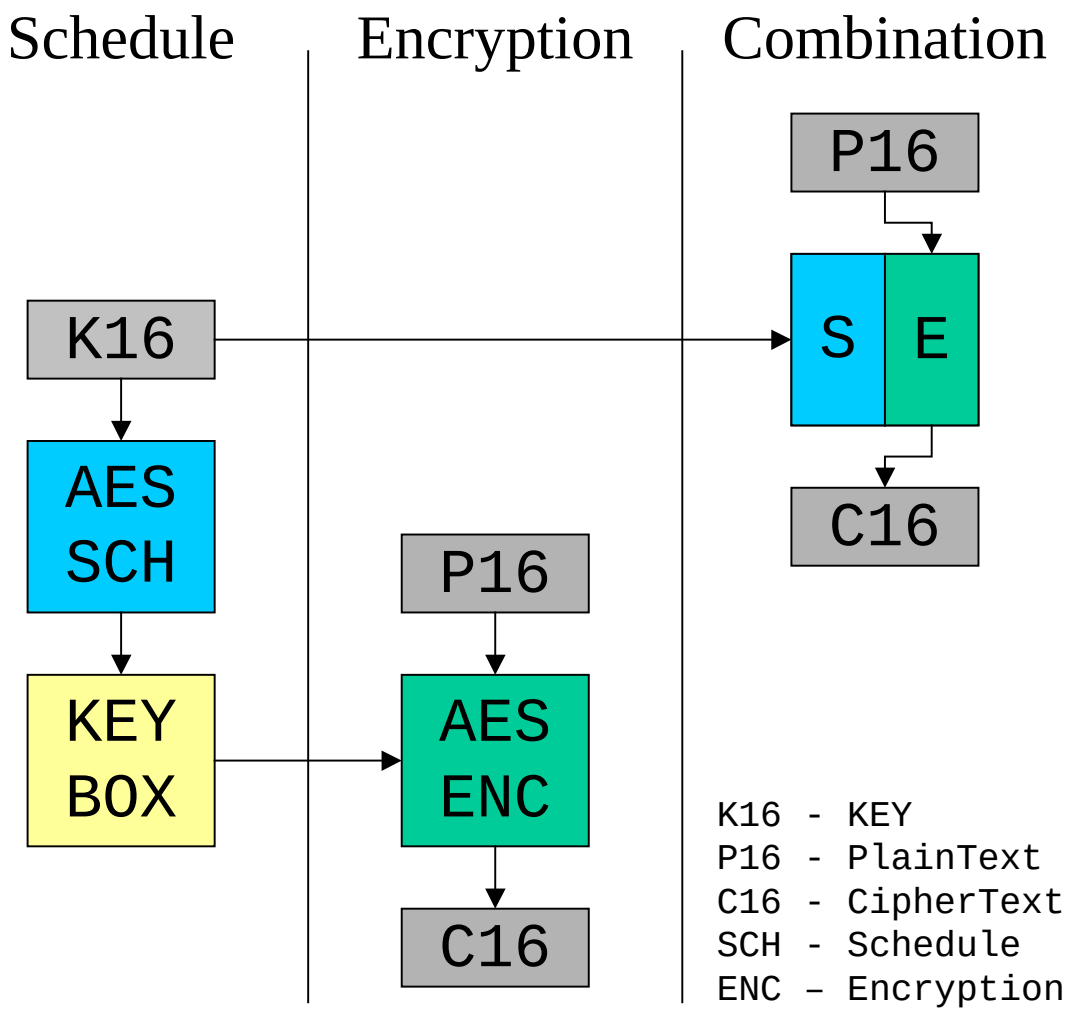

Abbildung 1 Der Basisalgorithmus

Die Entschlüsselungsoperation (Decryption (DEC)) wird in FLAM® nicht benötigt, da die Daten im CFB-Mode (Cipher Feedback) durch eine Stromchiffrierung ver- resp. entschlüsselt werden (vgl. Bruce Schneier, Angewandte Kryptographie).

Die Verschlüsselung mit AES wird von FLAM® nur im MODE=ADC® (Advanced Data Compression) oder im MODE=NDC (No Data Compression) – einer Unterfunktion der ADC-Algorithmik – unterstützt. Mit NDC werden die reinen Nettodaten nur 1:1 kopiert. Damit kann auch jede FLAMFILE® im "Nachhinein" ohne Performance-Verluste (2-Schritt-Verfahren) verschlüsselt werden. Auf diese Weise kann sogar eine "leere" Datei so verschlüsselt werden, dass "leer" nicht mehr erkennbar ist.

Wie die Vertraulichkeit und die Integrität der Daten in einer FLAMFILE® geschützt werden, wird in den nächsten Abschnitten beschrieben.

Bei diesem Schutz handelt es sich um reine Software-Kryptographie, was bedeutet, dass die verwendeten Schlüssel - wenn auch nur kurzzeitig - in klarer Form auf dem Rechner, wo die FLAMFILE® erzeugt wird, vorkommen. Da aber zu diesem Zeitpunkt auch die Originaldaten auf diesem Rechner existieren, kann ein Angreifer, der Zugriff auf den Rechner erlangt hat, gleich die klaren Daten ausspähen. Der verwendete Schlüssel nutzt ihm nur etwas, wenn dieser erneut zur Anwendung kommt und der Angreifer dann keinen Zugriff mehr auf das System hat.

Die maximale Sicherheit, die FLAM® mit AES bieten kann, ist abhängig von der *Sicherheit der Rechner*, auf denen die FLAMFILE® geschrieben bzw. gelesen wird. FLAM® stellt mit AES kryptographisch sicher, dass auf dem Übertragungsweg niemand ohne die Kenntnis des Schlüssels Daten manipulieren oder ausspähen kann. Man kann diese Sicherheit noch verbessern, indem man die verschlüsselte FLAMFILE® zwischen Servern austauscht, auf denen weder FLAM® noch die Originaldaten verfügbar sind. Dies ist eine einfache organisatorische Maßnahme, die die Sicherheit wesentlich erhöht. Diese organisatorische Lösung mit FLAM® ist auch wesentlich sicherer als eine Kombination aus File Transfer und integrierte Kryptographie in direkter Verbindung zwischen Sende- und Empfangssystem.

Kryptographie allein - ohne ein angepasstes organisatorisches Umfeld - ist kein Garant für Sicherheit.

Eine in Verbindung mit Kryptographie organisatorisch interessante Lösung, die FLAM® V4.0 bietet, ist das Parallel-Splitting. Durch die gleichmäßige Verteilung der verschlüsselten FLAMFILE® in Einheiten von nur 4 Bytes parallel auf mehrere Teildateien, kann man nur decodieren, wenn man den Schlüssel **und** alle zusammengehörenden Teildateien gleichzeitig an FLAM® übergibt. Damit kann u.U. das Problem der Synchronisation des Schlüssels gelöst werden (z.B. in der Langzeit-Archivierung durch Verteilung auf verschiedene Standorte).

Es gibt in FLAM® V4.0 ein Feature, mit dem man eine FLAMFILE® - ob verschlüsselt oder nicht - auf ihre technische Integrität prüfen kann (Checksummen auf der Basis von CRC-Routinen). Solche Techniken sind z.B. in Verbindung mit File-Transfer international allgemeiner Standard. *Sie schützen nicht vor Manipulation.*

Unabhängig davon kann man eine mit FLAM® V4.0 und AES verschlüsselte FLAMFILE® - *ohne zu dekomprimieren* - auf ihre Integrität gemäß den Anforderungen der Kryptographie prüfen. Dazu muss man allerdings den Schlüssel benutzen, mit dem diese FLAMFILE® erzeugt worden ist.

1.2 Übersicht

FLAM® nimmt für die Verschlüsselung ein max. 64 Bytes langes Passphrase als Schlüsselinformation entgegen. Dieses kann beim Aufruf von FLAM® auf unterschiedliche Art und Weise übergeben werden (***Verweis innerhalb des Handbuches***). Es ist die Basis für die Verschlüsselung der FLAMFILE®. Es handelt sich dabei um dieselbe Eingangsschnittstelle wie in FLAM® V3.0, wo ebenfalls variabel bis zu 64 Bytes (64 x 8 = 512 Bits) lange Schlüssel zur Verschlüsselung der FLAMFILE® übergeben wurden.

Die alte hochperformante Verschlüsselung basierte nicht auf kryptographisch anerkannten Verfahren, sondern auf einem Zusatz zum Komprimierungsalgorithmus ADC (Advanced Data Compression) und war eine Eigenentwicklung der limes datentechnik® gmbh. Da in vielen Bereichen der Informationssicherheit anerkannte kryptographische Verfahren notwendig sind, um die Sicherheit nachweisen zu können, wird mit FLAM® 4.0 zusätzlich der Advanced Encryption Standard (AES) zur Verschlüsselung angeboten. Wie die Absicherung der FLAMFILE® mit AES erfolgt, wird nachfolgend beschrieben.

FLAM® ist ein Komprimierungswerkzeug, das mehrere Dateien aufnehmen kann. Die Dateien in FLAM® werden als Member bezeichnet. Diese Member werden in autarken Segmenten von maximal 64 KByte komprimiert. Alle komprimierten Member werden als eine FLAMFILE® zusammengefasst. Damit ergibt sich ein Baum mit 3 Ebenen (File, Member, Segment mit Header/Trailer). Diese Hierarchie muss das kryptographische System widerspiegeln. Eine Übersicht über das Datenformat in einer FLAMFILE® kann der nächsten Abbildung entnommen werden.

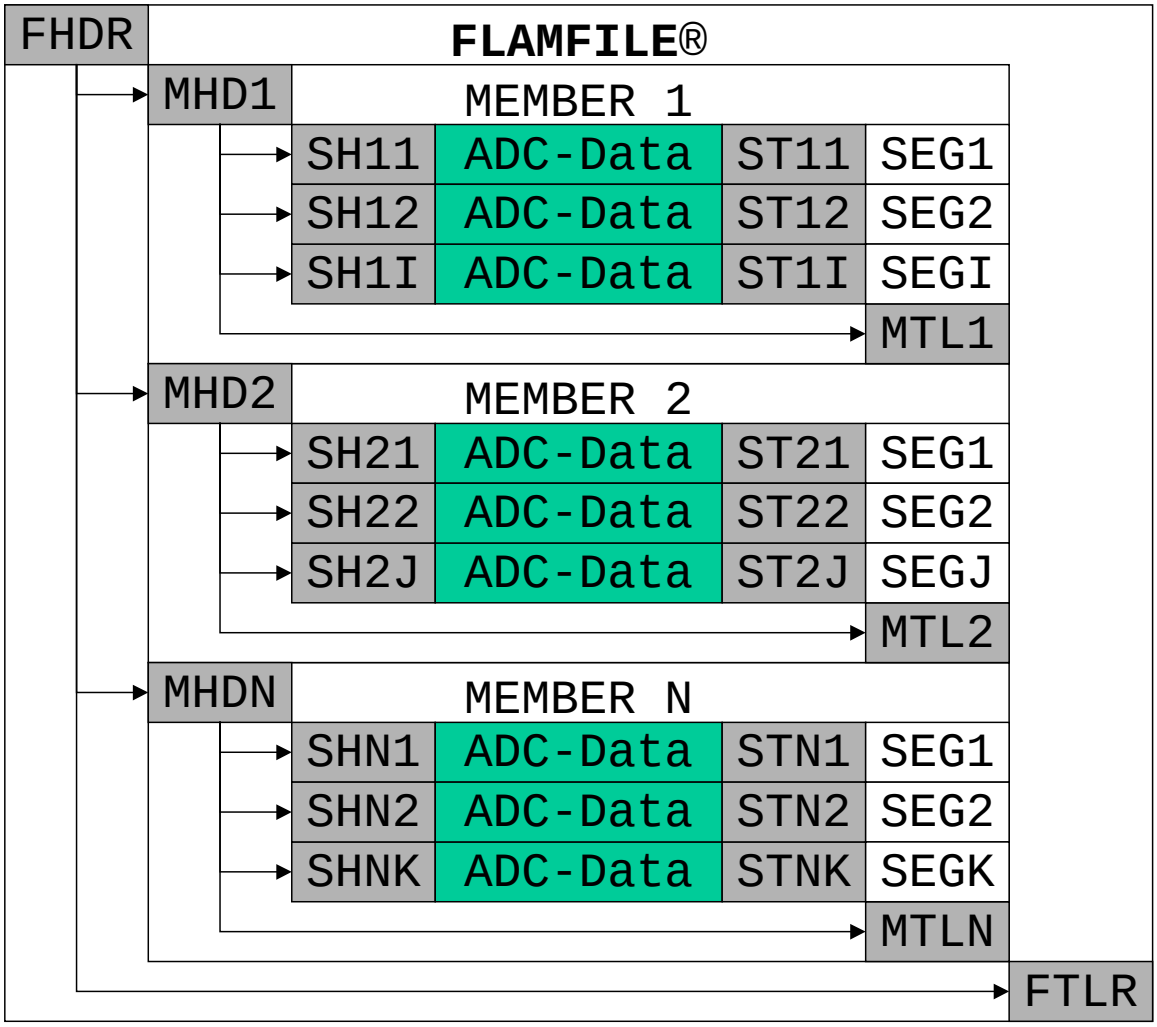

Abbildung 2 Das Datenformat einer FLAMFILE®

Die Besonderheit von FLAM® ist der direkte Zugriff auf einzelne Segmente aus logischen Einheiten (Records mit variabler Länge). Dies ermöglicht es, dass nur die Teile der FLAMFILE® dekomprimiert werden, die gebraucht werden. FLAM® ist hierfür auch als Subsystem für den Dateizugriff erhältlich. Im Fall von Fehlern oder Sabotage kann man Segmente retten, die korrekt sind. Die Möglichkeit eines direkten Zugriffs auf Segmente bedingt, dass diese autark kryptographisch behandelt werden müssen. Hierzu gehört der Schutz der Vertraulichkeit und der Integrität des Segmentkomprimats. Die Bildung des Segmentkomprimats in FLAM® wird in der folgenden Abbildung veranschaulicht.

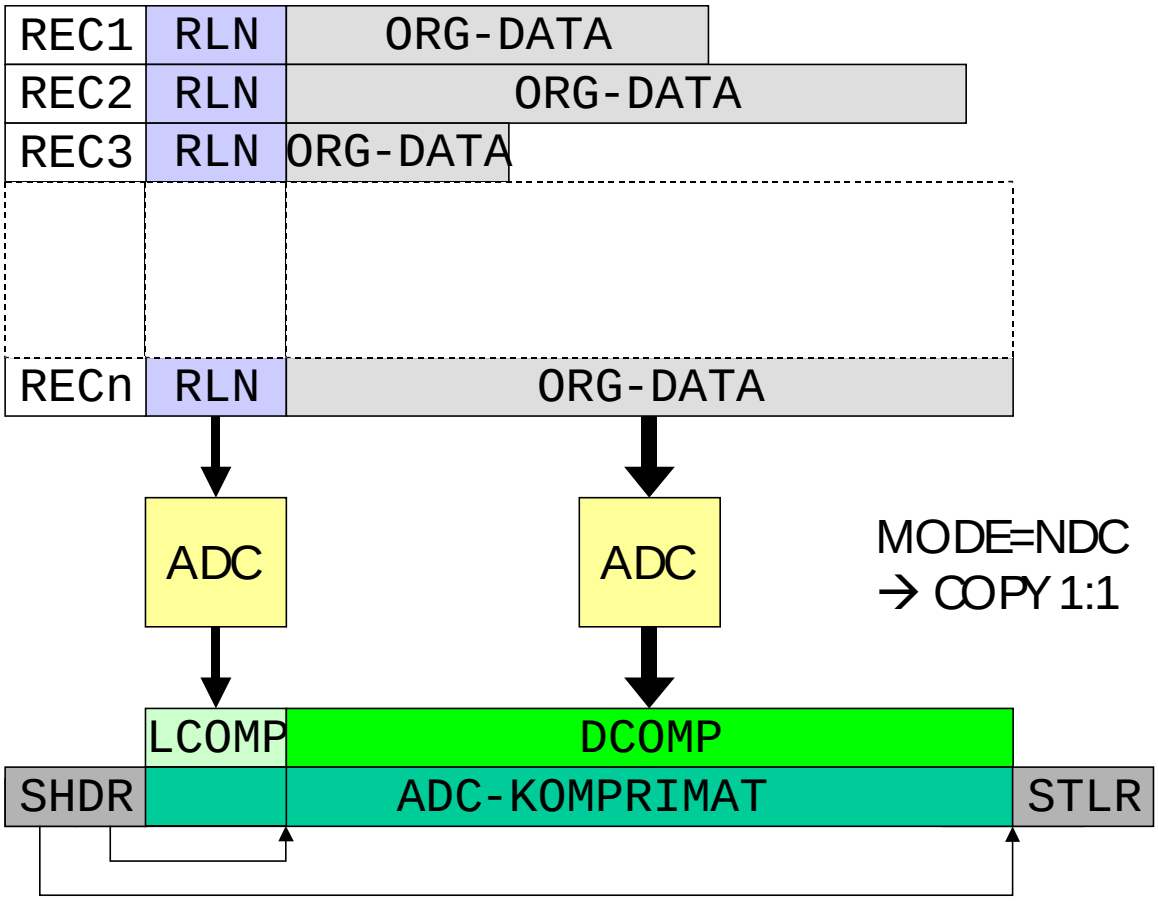

Abbildung 3 Die Bildung eines FLAM®-Segmentkomprimats

In FLAM® werden nur die grün (dunkel) dargestellten komprimierten Daten (Komprimat der Satzattribute **plus** das Komprimat der Nettodaten resp. beim NDC die Kopie davon) verschlüsselt. Der gesamte Rest, d.h. Meta-Informationen in Headern und Trailern einer FLAMFILE® wird mit Hilfe von kryptographischen Checksummen, den sogenannten MACs (Message Authentification Codes), vor Manipulationen geschützt.

Die folgende Abbildung macht deutlich, dass alle Bestandteile einer FLAMFILE® kryptographisch geschützt werden.

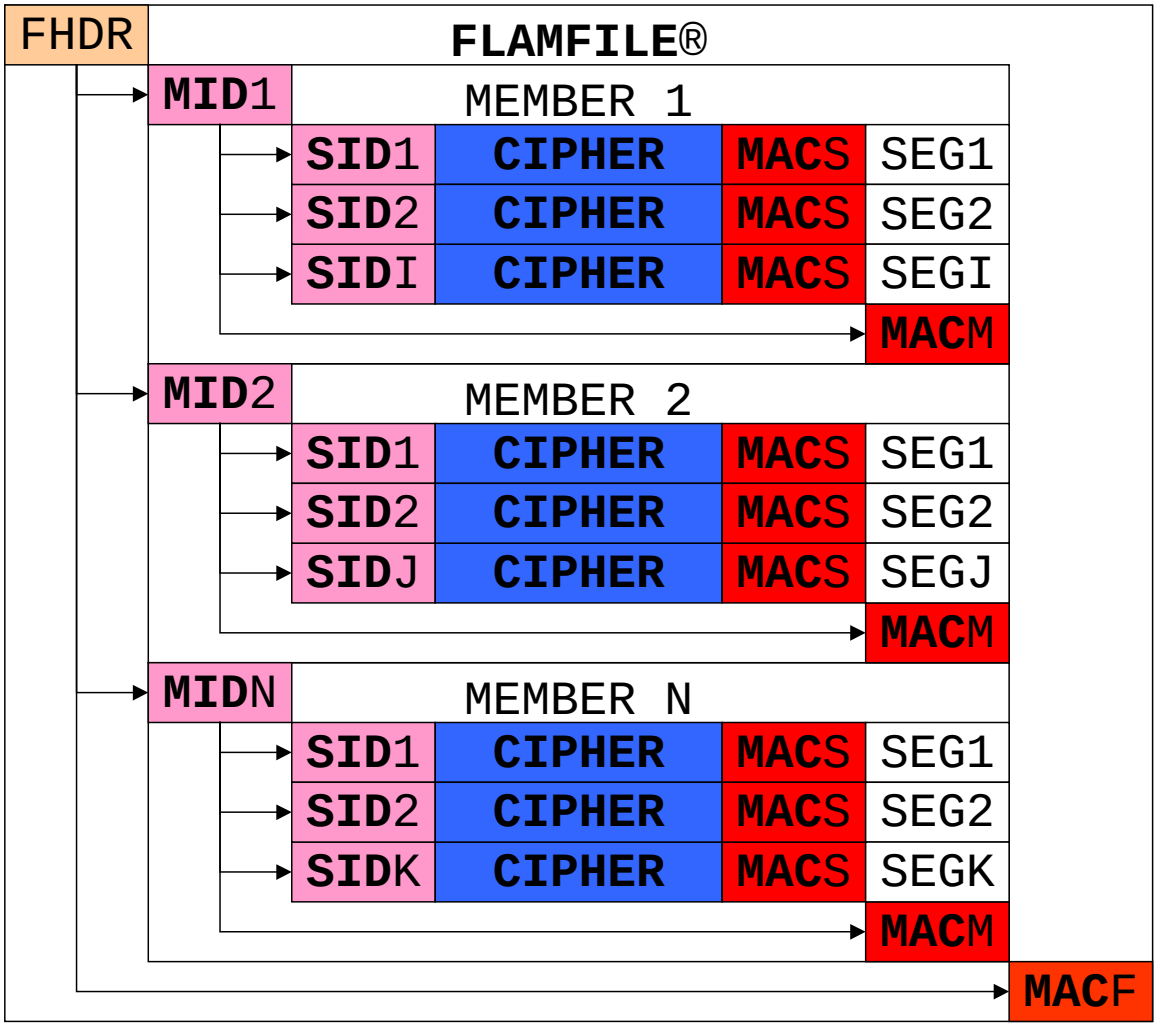

Abbildung 4 Die Kryptographie in einer FLAMFILE®

Durch die Autarkie der Member/Segmente wäre es nun möglich, ganze gültige Member/Segmente aus einer FLAMFILE® zu entfernen, auszutauschen bzw. hinzuzufügen, was einer Manipulation entsprechen würde. Dies bedingt, dass die Reihenfolge und die Vollständigkeit der Member/Segmente geprüft werden muss. Zusammenfassend lassen sich folgende kryptographische Schutzmaßnahmen für ein Segment definieren:

- Verschlüsselung des Segmentkomprimats
- MAC-Bildung über das verschlüsselte Segmentkomprimat (MAC1)
- MAC-Bildung zur kryptographischen Verkettung der Segmente (MAC2SS)
- MAC-Bildung über den Segmentheader und den Segmenttrailer, wozu auch der MAC1 und der MAC2SS gehören (MAC3).

Damit sind die Segment- und Member-Identifikation (MID, SID) kryptographisch mit AES abgesicherte Daten.

Dieses Vorgehen ist durch die bewährte interne Programmstruktur von FLAM® bestimmt, die auch in der Version 4.0 beibehalten wird.

Die folgende Abbildung macht den Integritätsschutz über ein Segment deutlich.

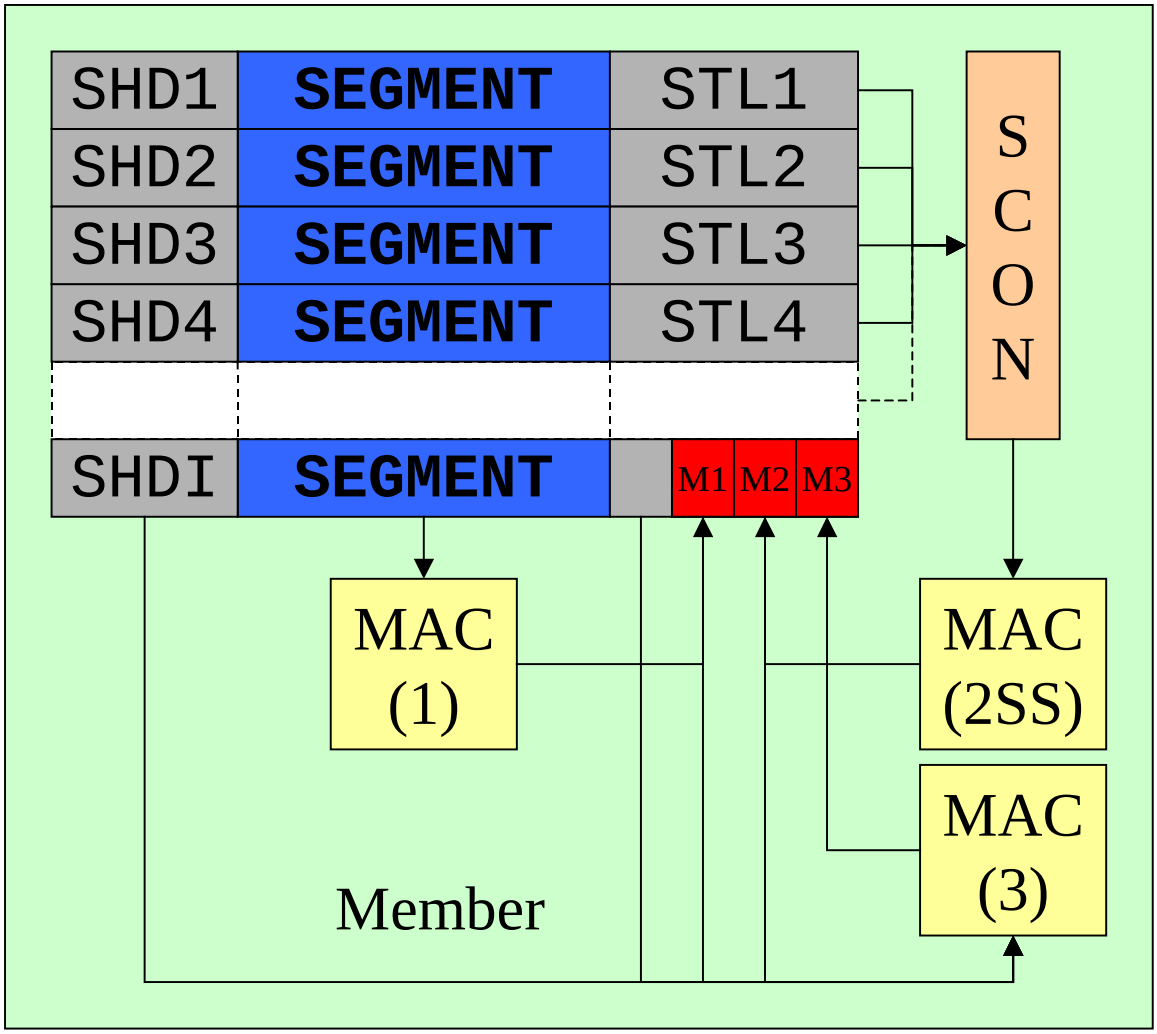

Abbildung 5 Der Integritätsschutz auf Segmentlevel

Auf dem Memberlevel muss die Reihenfolge und Vollständigkeit ganzer Member sichergestellt werden. Die Segment-Connectoren (SCON) mit 128 Bits sorgen für eine AES-basierte Verkettung signifikanter Informationen zwischen Segment-Trailern in einem Member.

Hierzu gehört der Übergang vom letzten Segment zum Member-Trailer (MAC2SM) und der Übergang von Member zu Member (MAC2MM). Diese beiden MACs werden zusammen mit dem Memberheader und den zusätzlichen Meta-Informationen im Membertrailer durch einen MAC3 geschützt.

Die folgende Abbildung macht den Integritätsschutz auf Memberlevel deutlich.

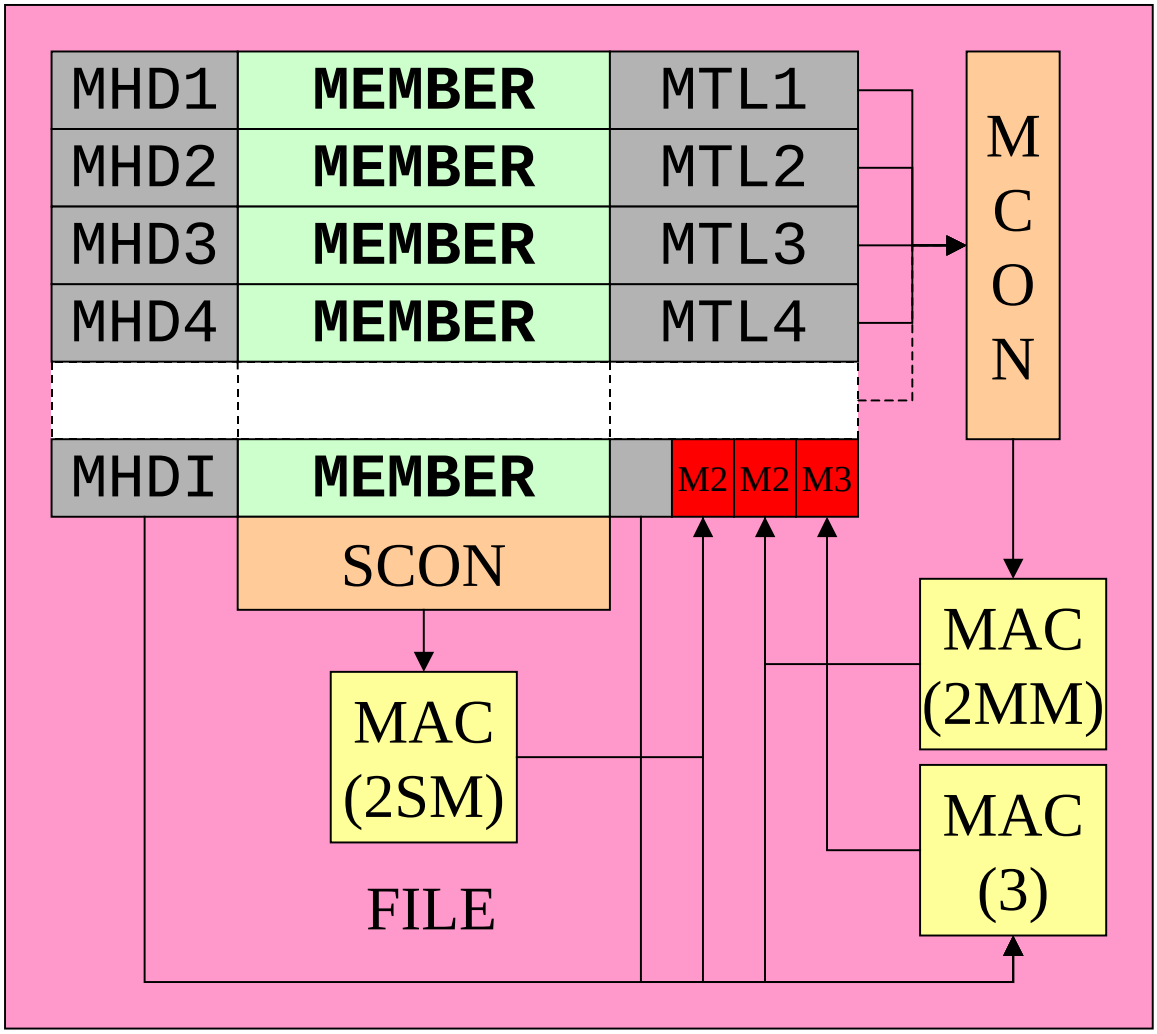

Abbildung 6 Der Integritätsschutz auf Memberlevel

Die Member-Connectoren (MCON) mit 128 Bits sorgen für eine AES-basierte Verkettung signifikanter Informationen zwischen Membertrailern.

Auf Filelevel muss zum Schluss noch die Verkettung des letzten Members zum FLAMFILE®-Trailer (MAC2MF) sichergestellt werden, bevor der MAC3 über den Fileheader und den Filetrailer inkl. MAC2MF berechnet werden kann.

Die folgende Abbildung veranschaulicht auch diesen Vorgang.

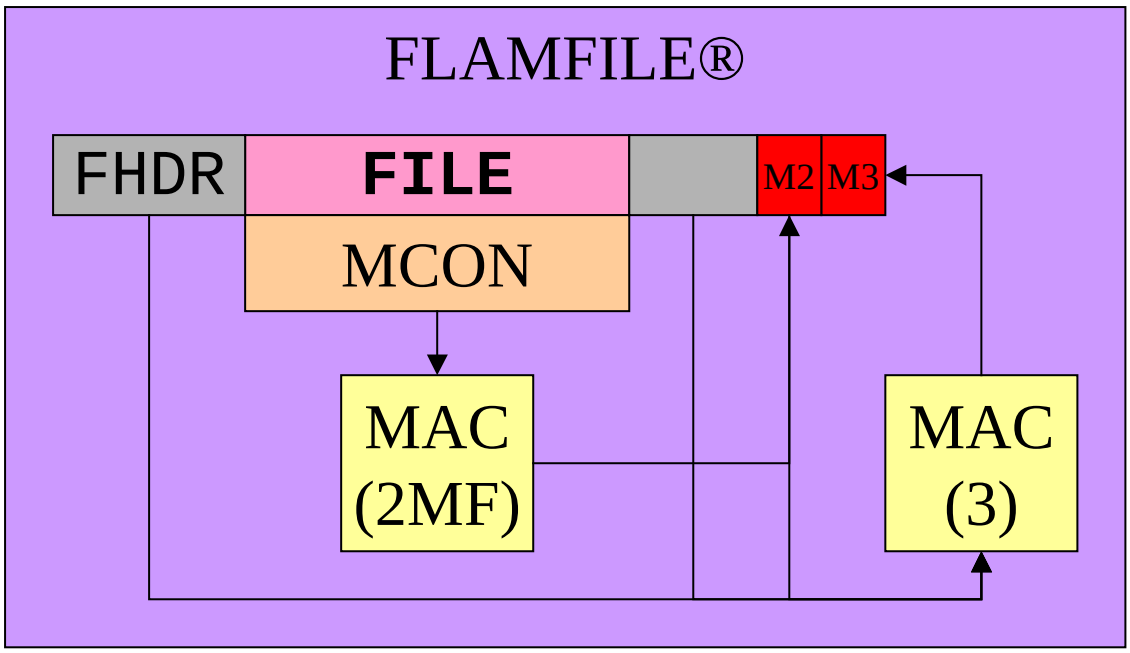

Abbildung 7 Der Integritätsschutz auf Filelevel

Durch den dezidierten Integritätsschutz über alle Level kann optional direkt auf Member/Segmente zugegriffen oder sequentiell die ganze Datei durchgearbeitet werden, wobei sichergestellt ist, dass niemand die Daten, die sich im Zugriff befinden, manipuliert hat.

Da nur das reine Segmentkomprimat verschlüsselt wird, hat ein Angreifer kein einziges Byte bekannten Klartext für eine Attacke zur Verfügung. Bei einem ADC-Komprimat handelt es sich im Idealfall um ein redundanzfreies "weißes Rauschen". Dadurch wird ein möglicher Angriff noch erschwert. Auch bei MODE=NDC (No Data Compression) werden die Daten vor der Verschlüsselung so permutiert (Scrambling), dass sie von einem normalen Komprimat nicht zu unterscheiden sind.

Geringste Bit-Fehler in den entschlüsselten Daten machen eine Rekonstruktion der Originaldaten durch Unscrambling/Dekomprimierung unmöglich. Fehler in den MACs signalisieren, dass eine Weiterverarbeitung der entschlüsselten Daten unsinnig wäre. Die Ursachen hierfür können Fehler in den Daten oder die Benutzung eines falschen Schlüssels sein. Datenfehler wegen rein technischer Probleme sind heutzutage äußerst selten.

Will man mit FLAM® "probieren", ob ein Input-Schlüssel falsch oder richtig ist, muss FLAM® das erste Segment komplett entschlüsseln. Erst die MAC-Prüfungen auf Segmentlevel geben darüber Auskunft, ob der Schlüssel korrekt oder falsch war. Da der Aufwand hierfür - im Sinne einfachen Probierens - extrem groß und eine Einwegfunktion ist, kann FLAM® selbst nicht als Angriffstool für eine Brute-Force-Attacke verwendet werden. Für den berechtigten Benutzer hat das bzgl. Performance und auch sonst überhaupt keine Nachteile. Wegen der Komplexität des Schichtenmodells in FLAM® (vgl. oben) ist es nicht möglich, diese Vorgehensweise durch eine eigene Lösung zu unterlaufen, um schneller probieren zu können.

1.3 Die Verfahren

Alle kryptographisch relevanten Verfahren in FLAM® basieren ausschließlich auf dem Basisalgorithmus AES, welcher - im Fall der Fälle - auf einfache Weise durch einen anderen Algorithmus ersetzt werden kann (transparente Ersetzung). Damit ist das FLAM®-Konzept gegenüber solchen Änderungen invariant und zukunftssicher.

1.3.1 Die Schlüsselableitung

Auf Segmentlevel werden 4 kryptographische Operationen durchgeführt. Eine Grundregel der Kryptographie lautet:

- Nimm für jede Operation einen eigenen Schlüssel.

Das bedeutet, dass auf Segmentlevel 4 Schlüssel benötigt werden. Eine weitere Grundregel der Kryptographie kann wie folgt formuliert werden:

- Für jedes Element muss ein eindeutiger, einmaliger Schlüssel verwendet werden.

Hieraus ergibt sich, dass pro Segment 4, pro Member 3 und pro FLAMFILE® 2 verschiedene eindeutige Schlüssel erzeugt werden müssen.

FLAM® kann mit Hilfe von eindeutigen Identifikationsdaten in den Segmentheadern direkt auf einzelne Segmente zugreifen (eindeutiges Adressierungsschema). Diese Informationen werden als Ableitungsdaten für die segmentspezifischen Schlüssel verwendet, was zu eindeutigen Schlüsselwerten führt. Gleiches gilt auch auf Memberlevel, wo 3 verschiedene Schlüssel benötigt werden. Wenn nun auf Filelevel aus dem maximal 64 Byte langen Passphrase vier 16 Bytes (128 Bits) lange Schlüssel abgeleitet werden, die dazu dienen, die MACs auf Filelevel zu berechnen und die Schlüssel auf Memberlevel abzuleiten, stehen auf Memberlevel wiederum 4 verschiedene memberspezifische Schlüssel zur MAC-Berechnung und zur Ableitung der 4 verschiedenen segmentspezifischen Schlüssel zur Verfügung.

Da für die Berechnung von MACs und die Verschlüsselung von Daten jeweils ein dazugehöriger Initialwert gebraucht wird, entstehen somit 4 Säulen, die auf jeder Ebene jeweils 4 verschiedene spezifische Schlüssel und 4 verschiedene spezifische Initialwerte zur Verfügung stellen.

Die folgende Abbildung macht dies deutlich.

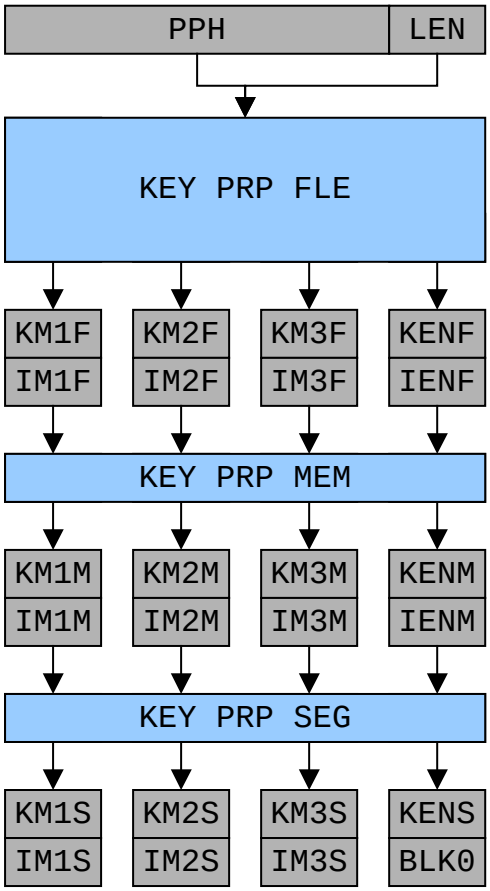

Abbildung 8 Die 4 Säulen zur Schlüsselableitung

Die zur Ableitung verwendeten Verfahren sind auf AES basierende kryptographische Einwegfunktionen, was ein Zurückrechnen von "unten nach oben" unmöglich macht und einen Pseudozufallsprozess darstellt. Nähere Informationen hierzu können der Detailspezifikation entnommen werden.

Da die aktuelle Uhrzeit, die Rechner-ID und andere kontextspezifische Informationen in die Schlüsselableitung einfließen, ist die FLAMFILE® als Chiffrat auch bei der Verwendung des gleichen Inputschlüssel und der gleichen Ausgangsdaten immer grundlegend verschieden (Unikat). Diese Eigenschaft erhöht die Sicherheit und muss bei der Wiedereinreichungskontrolle beachtet werden.

Die Daten in den Headern, die für diese Unikat-Eigenschaft verantwortlich sind, werden über MACs mit AES kryptographisch abgesichert.

1.3.2 Die Verschlüsselung des Segmentkomprimates

Die Verschlüsselung des Segmentkomprimates erfolgt wie schon erwähnt im Cipher-Feedback-Modus (CFB-Mode), um auf das Auffüllen auf volle 16 Byte-Blöcke (am Ende der Segmentkomprimate) verzichten zu können. Hierbei werden zusätzlich über einen Pre- und einen Post-XOR mit weiteren vorberechneten segmentspezifischen Pseudozufallsfolgen (S256/Sk, MS256/Mk) einfache Known-Plain-Text-Attacken gegen den segmentspezifischen Schlüssel wirkungsvoll verhindert. Für 16 Plaintext-Blöcke sieht dies wie folgt aus:

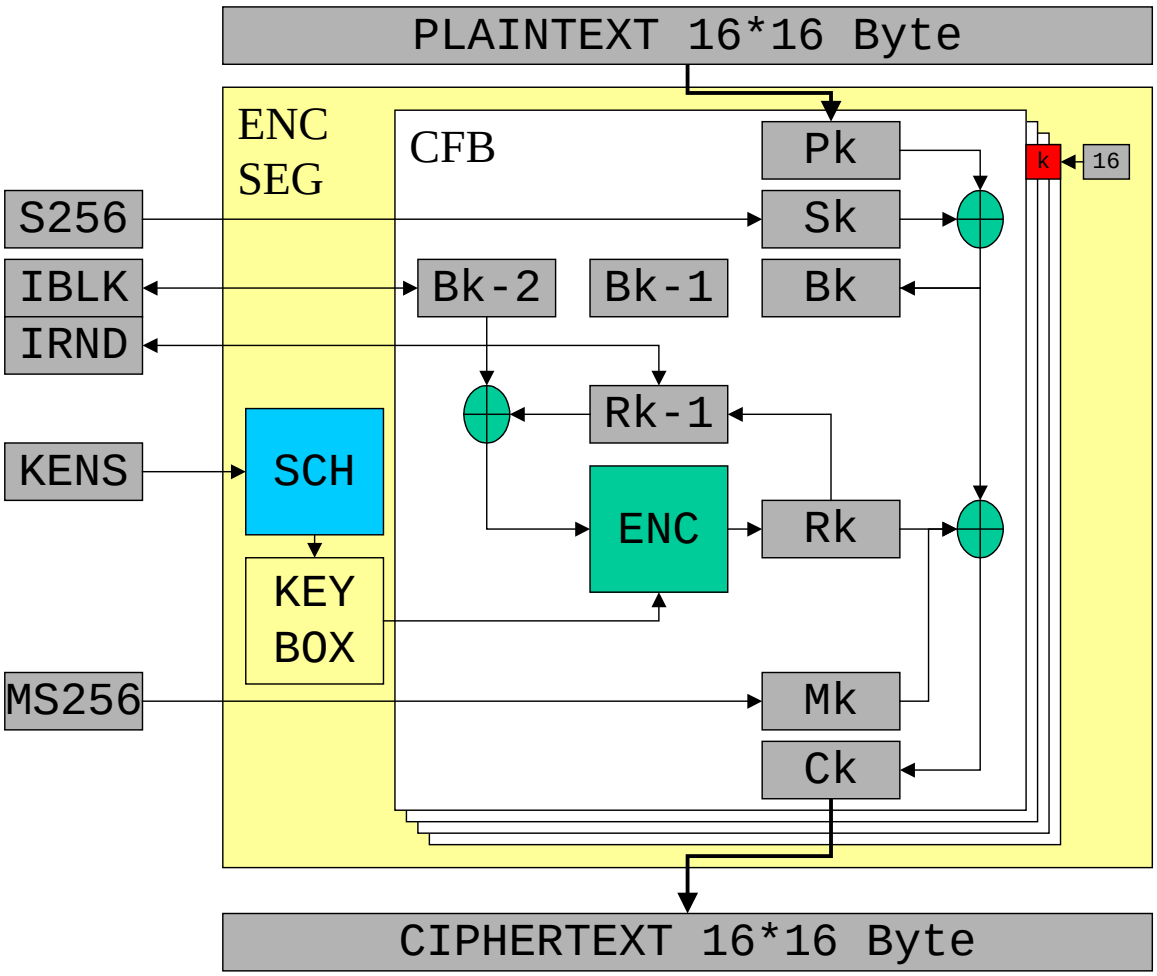

Abbildung 9 Die Verschlüsselung des Segmentkomprimates

Der Random-Wert (Rk) zur Verschlüsselung des Blocks (Bk) wird aus dem Random-Wert (Rk-1) und dem Block (Bk-2) gebildet. Das XOR-Produkt aus (Rk-1) mit (Bk-2) wird mit der Key-Box (KENS) verschlüsselt. Das ergibt den Random-Wert (Rk). IBLK und IRND sind Initialwerte.

Nähere Informationen hierzu können der Detailspezifikation entnommen werden.

1.3.3 Die MAC-Berechnung über das verschlüsselte Segmentkomprimat

Die MAC-Berechnung über das zuvor verschlüsselte Segmentkomprimat erfolgt im Cipher-Block-Chaining-Modus (CBC-Mode). Damit entspricht dieser MAC dem ANSI 9.9-MAC-Verfahren, wobei DES durch AES ersetzt wurde. Die folgende Abbildung macht dies deutlich.

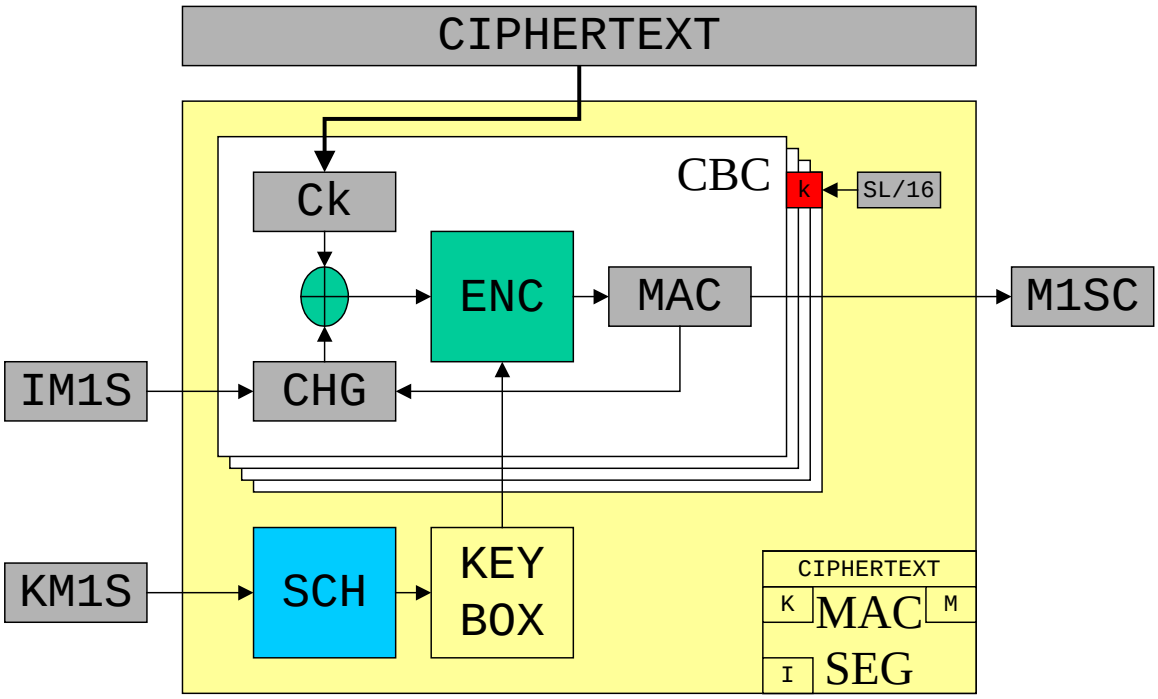

Abbildung 10 Die Bildung des MAC über das verschlüsselte Segmentkomprimat

Das Zwischenergebnis (CHG) wird per XOR mit dem Cipher-Block (Ck) verknüpft. Das XOR-Produkt wird mit der Key-Box (KM1S) verschlüsselt. Das ergibt das neue Zwischenergebnis (CHG). IM1S ist der Initialwert zu dieser Folge (CHG). Das Endergebnis dieser Folge ist der MAC (M1SC).

Nähere Informationen hierzu können der Detailspezifikation entnommen werden.

1.3.4 Die restlichen MAC-Berechnungen

Für die restlichen MAC-Berechnungen wird ein HASH-MAC-Verfahren verwendet. Hierbei wird AES zur Berechnung eines kryptographischen 128 Bit langen HASH-Wertes verwendet. Das Verfahren hierzu ist das einfachste der als sicher geltenden Verfahren zur Berechnung eines kryptographischen HASH-Wertes auf Basis eines symmetrischen Blockalgorithmus, wenn die Blocklänge und die Schlüssellänge gleich der HASH-Länge ist.

- $HSHi=ENC_{HSHi-1}(BLKi)$ XOR BLKi (vgl. Bruce Schneier, Angewandte Kryptographie)

Dieser HASH-Wert wird im Anschluss mit Hilfe des jeweiligen Schlüssels mit AES verschlüsselt, was den MAC ergibt. Nähere Informationen hierzu können der Detailspezifikation entnommen werden.

1.4 Zusammenfassung

Der Aufbau der FLAMFILE®, die Eigenschaften des Produktes und die interne Programmstruktur bedingen das kryptographische System, das sich im Detail zwangsläufig durch eine gewisse Komplexität auszeichnet. Des weiteren war es das Bestreben der limes datentechnik® gmbh, jegliche Angriffsmöglichkeit gegen Schlüssel - egal in welcher Ebene - auszuschließen, was zu weitreichenden zusätzlichen Schutzmaßnahmen geführt hat. All diese zusätzlichen Schutzmaßnahmen können der Detailspezifikation entnommen werden, da sie den Rahmen dieser Dokumentation sprengen würden. Die limes datentechnik® gmbh ist grundsätzlich daran interessiert, alle Sicherheitsmaßnahmen zu veröffentlichen.

Anmerkung: AES mit 128 Bits Schlüssellänge ist in den USA seit Ende Mai 2002 Behörden-Standard.

Eine umfassende Internet-Dokumentation zu AES finden Sie z.Zt. unter folgender Linkadresse:

http://csrc.nist.gov/archive/aes/

Leider ändert sich dieser Link immer mal wieder. Dann müssen Sie über eine Suchmaschine (z.B. www.google.de) mit den Suchbegriffen AES in Verbindung mit NIST die neue Linkadresse suchen.

NIST ist die US-Behörde, die AES in den USA zum Standard erklärt hat.