ICS 35.030
CCS L 80

中华人民共和国国家标准

GB/T 15852.2—2024
代替 GB/T 15852.2—2012

网络安全技术　消息鉴别码
第2部分：采用专门设计的杂凑函数的机制

Cybersecurity technology—Message authentication codes (MACs)—Part 2: Mechanisms using a dedicated hash-function

2024-09-29 发布　　2025-04-01 实施

国家市场监督管理总局
国家标准化管理委员会　发布

目　　次

前　言

本文件按照GB/T 1.1—2020《标准化工作导则　第1部分：标准化文件的结构和起草规则》的规定起草。

本文件是GB/T 15852的第2部分。GB/T 15852已经发布了以下部分：

——信息技术　安全技术　消息鉴别码　第1部分：采用分组密码的机制；

——网络安全技术　消息鉴别码　第2部分：采用专门设计的杂凑函数的机制；

——信息技术　安全技术　消息鉴别码　第3部分：采用泛杂凑函数的机制。

本文件代替GB/T 15852.2—2012《信息技术　安全技术　消息鉴别码　第2部分：采用专用杂凑函数的机制》，与GB/T 15852.2—2012相比，除结构调整和编辑性改动外，主要技术变化如下：

a) 增加了术语“熵”“输入数据位串”“安全性强度”及定义，更改了术语“杂凑函数”“填充”“初始值”“轮函数”“分组”“字”的定义，删除了术语“抗碰撞杂凑函数”“消息比特串”（见第3章，2012年版的第3章）；

b) 删除了符号D'，更改了符号$\overline{h}$、K'、K_0、K_1、K_2、$\overline{K}$、$\overline{K_1}$、$\overline{K_2}$、R、S_0、S_1、S_2、T_0、T_1、T_2、U_0、U_1、U_2、ϕ'、$K_1[i]$、H的定义，增加了符号w、$\lceil\ \rceil$（见第4章，2012年版的第4章）；

c) 更改了可选的杂凑函数的范围，并更改了算法描述中采用专门设计的杂凑函数的说明和常数的计算（见第5章～第9章，2012年版的第5章～第9章）；

d) 增加了关于MAC值和输入数据串长度限制的说明（见第5章）；

e) 增加了MAC算法通则，增加了MAC算法密钥长度、输入数据位串长度的说明（见6.1、7.1、8.1）；

f) 增加了MAC算法描述的通则和步骤标注（见6.2、7.2、8.2）；

g) 增加了采用SM3密码杂凑算法的MAC算法1和MAC算法3的描述和相应的常数计算（见第6章～第9章）；删除了采用其他专门设计的杂凑函数的描述和相应的常数计算（见2012年版的第6章～第9章）

h) 更改了关于MAC算法2的安全证明的说明（见附录A，2012年版的附录B）。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别专利的责任。

本文件由全国网络安全标准化技术委员会（SAC/TC 260）提出并归口。

本文件起草单位：中国科学院软件研究所、中电科网络安全科技股份有限公司、中国科学院大学、国家密码管理局商用密码检测中心、桂林电子科技大学、广西网信信息技术有限公司、格尔软件股份有限公司、兴唐通信科技有限公司、郑州信大捷安信息技术股份有限公司、北京时代新威信息技术有限公司、北京时代亿信科技股份有限公司、长扬科技（北京）股份有限公司、中国电子科技集团公司第十五研究所、浙江大华技术股份有限公司、陕西省信息化工程研究院、华为技术有限公司。

本文件主要起草人：吴文玲、眭晗、张立廷、刘丽敏、孙思维、罗鹏、毛颖颖、张蕾、郑雅菲、韦永壮、韦博华、郑强、蔡子凡、刘为华、王连强、刘伟丰、赵华、李艳俊、魏东、赵晓荣、曾光。

本文件及其所代替文件的历次版本发布情况为：

——2012年首次发布为GB/T 15852.2—2012；

——本次为第一次修订。

引　言

消息鉴别码能够保护数据的完整性，也能够验证数据的来源。采用专门设计的杂凑函数的消息鉴别码是指：在设计过程中，以专门设计的杂凑函数（如 SM3 等）或其轮函数为主要部件，通过一定的迭代机制形成的消息鉴别码。

GB/T 15852 拟分为以下部分。

——第 1 部分：采用分组密码的机制。目的在于规定采用分组密码的消息鉴别码。

——第 2 部分：采用专门设计的杂凑函数的机制。目的在于规定采用专门设计的杂凑函数的消息鉴别码。

——第 3 部分：采用泛杂凑函数的机制。目的在于规定采用泛杂凑函数的消息鉴别码。

网络安全技术　消息鉴别码
第2部分：采用专门设计的杂凑函数的机制

1　范围

本文件规定了采用专门设计的杂凑函数的消息鉴别码(MAC)的用户使用要求，提供了3种采用专门设计的杂凑函数的消息鉴别码算法。

注1：这些消息鉴别码算法能用于数据完整性检验，检验数据是否被非授权地改变。

本文件适用于安全体系结构、过程及应用的安全服务。

注2：本文件定义的第一个MAC算法通常被称作MDx-MAC。它调用一次完整的杂凑函数，但对其中的轮函数做了细微的修改，把一个密钥加到了轮函数的附加常数上。第二个MAC算法通常被称作HMAC，它调用两次完整的杂凑函数。第三个MAC算法是MDx-MAC的一个变种，它限制输入长度不大于256位。在只处理较短输入的情况下，它有更好的性能。

2　规范性引用文件

下列文件中的内容通过文中的规范性引用而构成本文件必不可少的条款。其中，注日期的引用文件，仅该日期对应的版本适用于本文件；不注日期的引用文件，其最新版本(包括所有的修改单)适用于本文件。

GB/T 18238.3—2024　网络安全技术　杂凑函数　第3部分：专门设计的杂凑函数

GB/T 25069—2022　信息安全技术　术语

GB/T 32905—2016　信息安全技术　SM3密码杂凑算法

3　术语和定义

GB/T 25069—2022界定的以及下列术语和定义适用于本文件。

3.1

熵　entropy

封闭系统中无序性、随机性或可变性的度量。

注：随机变量X的熵是观察X所获得的信息量的量化度量。

[来源：ISO/IEC 18031：2011，3.11]

3.2

杂凑函数　hash-function

将任意长的位串映射为定长位串的函数，满足下列性质：

——给定一个输出位串，寻找一个输入位串来产生该输出位串，在计算上不可行；

——给定一个输入位串，寻找另一个不同的输入位串来产生相同的输出位串，在计算上不可行。

[来源：GB/T 25069—2022，3.505，有修改]

3.3

输入数据位串　input data string

输入MAC算法的位串。

3.4

填充　padding

向某一数据串附加额外位的操作。

[来源:GB/T 25069—2022,3.598]

3.5

初始值　initializing value

在密码变换中,为增强安全性或使密码设备同步而引入的用于数据变换的起始数据。

[来源:GB/T 25069—2022,3.80,有修改]

3.6

杂凑值　hash value

密码杂凑运算的结果。

[来源:GB/T 25069—2022,3.764]

3.7

轮函数　round-function

构成杂凑函数的主要部件之一,将两个特定长度的位串映射为一个定长位串的函数,被迭代地用于杂凑函数的计算过程。

注:在该领域的文献中,多个术语具有与轮函数相同或相似的含义。例如:压缩函数和迭代函数。

[来源:GB/T 18238.1—2024,3.8]

3.8

分组　block

作为一个单位记录或传输的元素序列。

注:这里的元素可为字符、字或记录。

[来源:GB/T 25069—2022,3.354]

3.9

字　word

为给定目的视为一个单位的字符串。

[来源:GB/T 25069—2022,3.812,有修改]

3.10

消息鉴别码　message authentication code,MAC

MAC 算法输出的位串。

注:一个 MAC 有时也称作一个密码校验值。

[来源:GB/T 15852.1—2020,3.10,有修改]

3.11

消息鉴别码算法　message authentication code algorithm

输入为密钥和消息,输出为一个固定长度的位串的算法,满足下面两个性质:

——对于任何密钥和消息,MAC 算法都能快速有效地计算;

——对于任何固定的密钥,攻击者在没有获得密钥信息的情况下,即使获得了一些(消息,MAC)对,对任何新的消息预测其 MAC 在计算上是不可行的。

注 1:MAC 算法有时也称作密码校验函数。

注 2:计算不可行性依赖于使用者具体的安全要求及其环境。

[来源:GB/T 15852.1—2020,3.11,有修改]

3.12

MAC 算法密钥　MAC algorithm key

用于控制消息鉴别码算法运算的密钥。

[来源:GB/T 15852.1—2020,3.12]

3.13

输出变换　output transformation

在算法末尾且截断操作之前所应用的函数。

[来源:GB/T 15852.1—2020,3.14]

3.14

安全性强度　security strength

与攻破密码算法或系统所需的工作量相关的数值。

注:以位为单位,s 位的安全强度表示需要的操作数是 2^s。

4　符号

下列符号适用于本文件。

C_i、C'_i:	轮函数中使用的常数字
D:	输入数据位串,即:将要被输入到 MAC 算法的数据位串
$\overline{D}$:	经过填充的数据位串
H:	杂凑值
H'、H'':	长度为 L_2 的位串,在 MAC 算法计算中被用来存储临时结果
h:	杂凑函数
h':	被修改了常数和初始值 IV 的杂凑函数 h
$\overline{h}$:	简化的杂凑函数 h,没有数据填充和长度附加,没有将轮函数输出(L_2 位)截断成其最左的 L_H 位。 注 1:被用来处理长度为 L_1 正整数倍的输入位串。 注 2:$\overline{h}$ 的输出是长度为 L_2 的位串,而不是长度为 L_H 的位串。
IV、IV'、IV_1、IV_2:	初始值
K:	MAC 算法的密钥
K'、K_0、K_1、K_2、$\overline{K}$、$\overline{K_1}$、$\overline{K_2}$:	被用于 MAC 算法的派生密钥
$K_1[i]$:	派生密钥 K_1 的第 i 个字
KT:	MAC 算法 1 的输出变换步骤中,函数 ϕ' 的第一个输入位串
k:	MAC 算法密钥的位长度
$\mathring{L}$:	MAC 算法 3 中表示消息长度的位串
L_X:	位串 X 的位长度
L_1:	输入到轮函数 ϕ 的两个位串中,第一个位串的位长度
L_2:	输入到轮函数 ϕ 的两个位串中,第二个位串的位长度;轮函数 ϕ 输出值的位长度;初始值 IV 的位长度
$MSB_j(X)$:	位串 X 最左侧 j 位位串
m:	MAC 值的位长度
$OPAD$、$IPAD$:	MAC 算法 2 中使用的常数位串
q:	经过填充和分割操作后,输入数据位串 D 的分组个数
R、S_0、S_1、S_2:	MAC 算法 1 和算法 3 的常数计算中使用的常数位串
T_0、T_1、T_2、U_0、U_1、U_2:	MAC 算法 1 和算法 3 的密钥派生中使用的常数位串
w:	字的位长度,取 32
$X \oplus Y$:	位串 X 和 Y 的异或值

$X \| Y$：	按顺序将位串 X 和 Y 连接所构成的位串
$:=$：	MAC 算法定义中使用的赋值符号
$\lceil \rceil$：	向上取整符号
ϕ：	轮函数，即：若 X 和 Y 分别表示长度为 L_1 和 L_2 的位串，则 $\phi(X,Y)$ 表示将 ϕ 作用到 X 和 Y 所得到的位串
ϕ'：	修改的轮函数，其常数与原轮函数中使用的不同
$+_w$：	模 2^w 加法操作，其中 w 是一个字的位数，即：若 A 和 B 是字，那么把 A 和 B 看作是整数的 2 进制表示，计算它们的和再模 2^w，所得到的结果在 0 和 2^w-1 之间，把它看作为字，记作 $A+_wB$。

5 用户使用要求

使用本文件中给出的 MAC 算法的用户需要选择：

——按照 GB/T 18238.3—2024 第 7 章规定的专门设计的杂凑函数；

——第 6 章、第 7 章、第 8 章中定义的一种 MAC 算法；

——MAC 的位长度 m，其中 m 不小于 32。

用户之间就这些选择达成协议对数据完整机制的运作至关重要。

MAC 算法中使用的密钥 K 的熵应满足或超过由 MAC 算法提供的安全性强度。

在任何情况下，MAC 算法密钥 K 的选择应使每个可能的密钥被选择的可能性近似相等。

对于 MAC 算法 1 和算法 2，MAC 的长度 m 应是一个正整数并且不大于杂凑值长度 L_H。对于 MAC 算法 2，MAC 值的长度 m 应不小于 32 位。对于 MAC 算法 3，MAC 的长度 m 是一个正整数并且不大于杂凑值长度的 1/2，即 $m \leqslant L_H/2$。输入数据串的位长度可能受专门设计的杂凑函数和/或 MAC 算法限制，需针对每个 MAC 算法进行讨论。

具体的 MAC 算法和 m 值的选择超出了本文件所规定的范围。

这些选择影响 MAC 算法的安全性级别。MAC 算法的安全性分析见附录 A。生成和验证 MAC 使用的密钥相同。如果输入数据串也被加密，那么计算 MAC 的密钥宜不同于用于加密的密钥。因为，密码实践宜为保密性和数据完整性使用相互独立的密钥。

附录 A 描述了本文件中定义的 MAC 算法的主要攻击和安全证明。

用符合附录 B 要求的对象标识符标识本文件规定的机制。

附录 C 提供了本文件中定义的 MAC 算法的测试向量，用于检查实现的正确性。

6 MAC 算法 1（MDx-MAC）

6.1 通则

本章描述了采用专门设计的杂凑函数的 MAC 算法 1，其中专门设计的杂凑函数符合 GB/T 18238.3—2024 第 7 章的规定。MAC 算法 1 又称 MDx-MAC。

MAC 算法 1 要求调用一次杂凑函数以计算 MAC 值，而且要求修改对应轮函数中的常数。MAC 算法 1 可最大容纳 128 位长度的密钥 K，因此可提供最高 128 位的安全性强度。对于 MAC 算法 1，采用 SM3 密码杂凑算法时的输入数据串 D 的位长度应不大于 $2^{64}-1$。

6.2 MAC 算法 1 的描述

6.2.1 通则

MAC 算法 1 要求如下 5 步操作：密钥扩展、修改常数和初始值、杂凑操作、输出变换和截断操作。

6.2.2 步骤 1(密钥扩展)

若 K 长度小于 128 位,则将 K 重复$\lceil 128/K \rceil$次,选取所得结果的最左侧 128 位作为 128 位密钥 K':

$$K' := MSB_{128}(K \| K \| K \ldots \| K)$$

若 K 长度不小于 128 位,$K' := MSB_{128}(K)$。

按照如下操作计算派生密钥 K_0、K_1和 K_2:

$$K_0 := \overline{h}(K' \| U_0 \| K'),$$

$$K_1 := MSB_{256}(\overline{h}(K' \| U_1 \| K'))$$

$$K_2 := MSB_{128}(\overline{h}(K' \| U_2 \| K'))$$

其中,$\overline{h}$表示简化的专门设计的杂凑函数 h,U_0、U_1和 U_2是第 9 章中定义的 768 位的常数。数据填充和长度附加可被省略,因为此时输入位串的长度是 L_1或 $2L_1$。

派生 K_0时,省略截断操作,且 K_0的长度总是 L_2位。

派生密钥 K_1被分割成 8 个字,记作 $K_1[i]$ $(0 \leqslant i \leqslant 7)$,即:

$$K_1 = K_1[0] \| K_1[1] \| K_1[2] \| K_1[3] \| K_1[4] \| K_1[5] \| K_1[6] \| K_1[7]$$

从位串到字的转换,需要规定字节排列顺序。在这里的转换中,采用 GB/T 32905—2016 中规定的字节排列顺序。

6.2.3 步骤 2(修改常数和初始值)

轮函数中采用的附加常数,被修改为它与 K_1中的一个字进行模 2^w加的结果,例如:

$$C_0 := C_0 +_w K_1[0]$$

每个常数具体与 K_1的哪个字相加取决于使用的杂凑函数,在第 9 章中规定。

用 $IV' := K_0$取代杂凑函数的初始值 IV,所得的杂凑函数记作 h',其中的轮函数记作 ϕ'。

6.2.4 步骤 3(杂凑操作)

将输入数据位串 D 输入被修改的杂凑函数 h',即:

$$H' := h'(D)$$

6.2.5 步骤 4(输出变换)

增加一次被修改的轮函数 ϕ'的应用,其中输入的第一个参数为 KT,第二个参数为 H'(步骤 3 的结果),即:

$$H'' := \phi'(KT, H')$$

其中,$KT = K_2 \| (K_2 \oplus T_0) \| (K_2 \oplus T_1) \| (K_2 \oplus T_2)$。这里 T_0、T_1和 T_2是长度为 128 的位串,在第 9 章中定义。

输出变换对应于处理在数据填充和长度附加操作之后由 K_2派生得到的一个额外的数据分组。

6.2.6 步骤 5(截断操作)

取位串 H''最左侧 m 位,作为 MAC 值,即:

$$\mathrm{MAC} := MSB_m(H'')$$

6.3 效率

如果填充后的输入数据位串包括 q 个分组(其中填充方法由杂凑函数决定),那么 MAC 算法 1 需要调用 $q+7$ 次轮函数。通过预计算 K_0、K_1和 K_2,并且在杂凑函数的应用中用 IV'取代 IV,MAC 算法 1 调用轮函数的次数可降低到 $q+1$ 次。宜对杂凑函数的代码的预计算和初始值修改与步骤 2 要求

的强制性修改共同使用。对于长的输入位串，MAC 算法 1 和相应杂凑函数的性能相当。

7 MAC 算法 2(HMAC)

7.1 通则

本章描述了采用专门设计的杂凑函数的 MAC 算法 2，其中专门设计的杂凑函数符合 GB/T 18238.3—2024 第 7 章规定。MAC 算法 2 又称 HMAC。

MAC 算法 2 计算 MAC 值要求调用两次杂凑函数。杂凑函数要求 L_1 是 8 的正整数倍。

密钥长度 k 不小于 L_2 位，其中 L_2 是杂凑值的位长度。若输入密钥的长度大于 L_1 位，其中 L_1 是轮函数的数据输入的位长度，则密钥的杂凑值被用作密钥，其长度是 L_2 位。因此，对于 MAC 算法 2，假设密钥长度不大于 L_1 位，即：$L_2 \leqslant k \leqslant L_1$。

对于 MAC 算法 2，采用 SM3 密码杂凑算法时的输入数据串 D 的位长度应不大于 $2^{64}-512$。

7.2 MAC 算法 2 的描述

7.2.1 通则

MAC 算法 2 应按如下 4 步操作：密钥扩展、杂凑操作、输出变换和截断操作。

7.2.2 步骤 1(密钥扩展)

在密钥 K 的右侧填充 L_1-k 个 0，所得的长度为 L_1 的位串记作 $\overline{K}$。

将 $\overline{K}$ 扩展为两个子密钥 $\overline{K_1}$ 和 $\overline{K_2}$。

- 将 16 进制的值“36”(二进制表示为“00110110”)重复 $L_1/8$ 次连接起来，所得位串记作 $IPAD$。然后将 $\overline{K}$ 和位串 $IPAD$ 相异或，记作 $\overline{K_1}$，即：

$$\overline{K_1} := \overline{K} \oplus IPAD$$

- 将 16 进制的值“5C”(二进制表示为“01011100”)重复 $L_1/8$ 次连接起来，所得位串记作 $OPAD$。然后将 $\overline{K}$ 和位串 $OPAD$ 相异或，记作 $\overline{K_2}$，即：

$$\overline{K_2} := \overline{K} \oplus OPAD$$

7.2.3 步骤 2(杂凑操作)

将 $\overline{K_1}$ 和 D 相连接，作为输入到杂凑函数的位串，即：

$$H' := h(\overline{K_1} \| D)$$

7.2.4 步骤 3(输出变换)

将 $\overline{K_2}$ 和 H' 相连接，作为输入到杂凑函数的位串，即：

$$H'' := h(\overline{K_2} \| H')$$

7.2.5 步骤 4(截断操作)

取位串 H'' 最左侧 m 位，作为 MAC 值，即：

$$\mathrm{MAC} := MSB_m(H'')$$

7.3 效率

如果填充后的输入数据位串包括 q 个分组(其中填充方法由杂凑函数决定)，那么 MAC 算法 2 调

用轮函数 $q+3$ 次。

通过修改杂凑函数代码，MAC 算法 2 调用轮函数的次数可降低到 $q+1$ 次。可预计算 $IV_1:=\phi(\overline{K_1},IV)$ 和 $IV_2:=\phi(\overline{K_2},IV)$，并且在第一次调用杂凑函数时用 IV_1 取代 IV，在输出变换中（第二次调用杂凑函数）用 IV_2 取代 IV。同时，这也要求对填充方法进行修改：事实上，对杂凑函数的实际输入少了 L_1 位。这意味着应把 L_1 的值加到 L_D 上。

8 MAC 算法 3（MDx-MAC 的变种）

8.1 通则

本章描述了 MAC 算法 1 的一个变种，该变种对短输入（不大于 256 位）做了优化。

对于 MAC 算法 3，输入数据串 D 的位长度应不大于 256。

密钥位长度 k 应不大于 128，MAC 值位长度 m 应不大于 $L_H/2$。

8.2 MAC 算法 3 的描述

8.2.1 通则

MAC 算法 3 应按如下五步操作：密钥扩展、修改常数和初始值、填充、应用轮函数和截断操作。

8.2.2 步骤 1（密钥扩展）

若 K 长度小于 128 位，则将 K 重复$\lceil 128/K \rceil$次，选取所得结果的最左侧 128 位作为 128 位密钥 K'：

$$K':=MSB_{128}(K \| K \| \cdots \| K)$$

若 K 长度不小于 128 位，$K':=MSB_{128}(K)$。

按照如下操作计算派生密钥 K_0、K_1 和 K_2：

$$K_0:=\overline{h}(K' \| U_0 \| K'),$$

$$K_1:=MSB_{256}(\overline{h}(K' \| U_1 \| K'))$$

$$K_2:=MSB_{128}(\overline{h}(K' \| U_2 \| K'))$$

其中，$\overline{h}$ 表示简化的专门设计的杂凑函数 h，U_0、U_1 和 U_2 是第 9 章中定义的 768 位的常数。数据填充和长度附加可被省略，因为此时输入位串的长度是 L_1 或 $2L_1$ 位。

派生 K_0 时，省略截断操作，且 K_0 的长度总是 L_2 位。

派生密钥 K_1 被分割成 8 个字，记作 $K_1[i]$ $(0\leqslant i\leqslant 7)$，即：

$$K_1=K_1[0] \| K_1[1] \| K_1[2] \| K_1[3] \| K_1[4] \| K_1[5] \| K_1[6] \| K_1[7]$$

从位串到字的转换，需要规定字节排列顺序。在这里的转换中，采用 GB/T 32905—2016 中规定的字节排列顺序。

8.2.3 步骤 2（修改常数和初始值）

轮函数中采用的附加常数，被修改为它与 K_1 4 个字中的一个进行模 2^w 加的结果，例如：

$$C_0:=C_0+_w K_1[0]。$$

每个常数具体与 K_1 的哪个字相加取决于使用的杂凑函数，在第 9 章中规定。

用 $IV':=K_0$ 取代杂凑函数的初始值 IV，所得的杂凑函数记作 h'，其中的轮函数记作 ϕ'。

8.2.4 步骤 3（填充）

对原始消息填充的位串只用来计算 MAC。所以，这些填充位串（如果有）不必随原始消息存储或发送。MAC 的验证者应知道填充位串是否已经被存储或发送。对要输入 MAC 算法的数据位串 D，在

其右侧填充尽可能少(可能没有)的“0”位以获得长度是256位的数据位串$\overline{D}$。

如果输入数据位串D是空串,那么规定填充后的位串$\overline{D}$为256个“0”位。

8.2.5 步骤4(应用轮函数)

计算数据位串D的位长度L_D的二进制表示,并在左侧填充尽可能少的“0”位以获得长度为128的位串,记作$\widetilde{L}$。将K_2、$\overline{D}$和K_2与$\widetilde{L}$的异或值相连接,作为(修改常数的)轮函数ϕ'的输入。

$$H' := \phi'(K_2 \| \overline{D} \| (K_2 \oplus \widetilde{L}), IV')$$

8.2.6 步骤5(截断操作)

取位串H'最左侧m位,作为MAC值,即:

$$\mathrm{MAC} := MSB_m(H')$$

8.3 效率

MAC算法3需要调用7次简化的轮函数。通过预计算K_0、K_1和K_2,可降低到仅调用1次。

9 常数的计算

9.1 概述

第9章规定的常数被用在MAC算法1和MAC算法3中。

位串T_i和U_i是MAC算法中固定的元素。它们通过杂凑函数计算得到(只计算一次),不同杂凑函数中的常数不同。

128位的常数T_i和768位的常数U_i按照如下的方法定义。T_i的定义使用到496位的常数R=“$ab...yzAB...YZ01...89$”和16位的常数S_0、S_1、S_2,其中S_i通过重复两次数字i的16进制ASCII编码得到(比如,S_1的16进制表示为3131)。R和S_i都采用ASCII编码,等同于采用GB/T 1988—1998进行编码:

$$T_i := MSB_{128}(\overline{h}(S_i \| R)),\ i=0,1,2$$

$$U_i := T_i \| T_{i+1} \| T_{i+2} \| T_i \| T_{i+1} \| T_{i+2},\quad i=0,1,2$$

其中,T_i的下标采用模3加。对于所有的常数C_i、C'_i和所有的字$K_1[i]$,最高位对应于最左侧的位。常数C_i和C'_i采用16进制表示。

9.2 SM3密码杂凑函数

SM3密码杂凑函数中的128位常数T_i定义如下(以16进制表示):

$$T_0 = 52EA0B36B5A4FA8C8D9403894A7421BF$$

$$T_1 = 457E3B1FCE828A8E1442AA01AC83E2BE$$

$$T_2 = 740B7A08B7CCB27F54B31B160EF57302$$

SM3密码杂凑函数的轮函数中用到的常数字序列$C_0, C_1, \cdots, C_{63}$定义如下:

$$C_i = K_1[i \bmod 8] +_{32} C'_i \quad (0 \leqslant i \leqslant 63),$$

其中,序列$C'_0, \cdots, C'_{63}$用16进制表示定义如下:

$$C'_i = 79CC4519 \quad (0 \leqslant i \leqslant 15),$$

$$C'_i = 7A879D8A \quad (16 \leqslant i \leqslant 63)$$

附 录 A
（资料性）
MAC 算法的安全性分析

本附录讨论了本文件中 MAC 算法的安全强度。它的目标是协助本文件的使用者选择合适的 MAC 算法和参数值。

本附录中，$MAC_K(D)$表示用密钥为 K 的 MAC 算法对消息 D 进行计算所得到的 MAC。

为了确定 MAC 算法的安全强度，本附录考虑了如下两类攻击。

——**伪造攻击**：此类攻击是在没有密钥 K 的情况下，对消息 D 预测 $MAC_K(D)$。如果攻击者能对一个消息成功预测其 MAC，那么称它有能力“伪造”。实际攻击经常要求伪造是可验证的，也就是说，以接近 1 的概率确认伪造的 MAC 是正确的。在许多应用中消息有特定的格式，这就意味着对消息 D 有额外限制。

——**密钥恢复攻击**：此类攻击根据大量的（消息，MAC）对找到 MAC 算法的密钥 K。密钥恢复攻击比伪造攻击更强大，因为它一旦成功就可进行任意伪造。

一个攻击的可行性依赖于攻击者已知和选择的（消息，MAC）对数目以及离线加密的次数。

对 MAC 算法可能的攻击描述如下，但是这里并不保证列举了所有的攻击。前两种攻击是一般性的，它们对任何 MAC 算法都有效。后两种适用于迭代的 MAC 算法（更多信息请参阅参考文献[4]）。

——**猜测 MAC**：这种伪造是不可验证的，成功概率为 $\max(1/2^m, 1/2^k)$。这种攻击适用于所有的 MAC 算法，只有合适地选择 m 和 k 才能够抵抗这种攻击。

——**密钥穷搜索**：这种攻击需要运行平均 2^{k-1} 次 MAC 算法，并且需要 k/m 对（消息，MAC）以唯一确定密钥。同样这种攻击适用于所有 MAC 算法，合适地选择 k 能够抵抗这种攻击。另外，MAC 算法使用者也可阻止攻击者获得 k/m 对（消息，MAC）以抵抗这种攻击。例如，如果 $k=128$ 且 $m=64$，大约有 2^{64} 个密钥与给定的（消息，MAC）对相对应。如果计算每个 MAC 都使用不同的密钥，那么密钥穷搜索攻击并不比猜测 MAC 攻击更有效。

——**生日攻击**[4]：如果攻击者获得足够数目的（消息，MAC）对，将有很高的概率找到消息 D 和 D'，使得：$MAC_K(D)=MAC_K(D')$，并且两次输出变换的输入值在两次 MAC 计算中是相等的；这被称为内部碰撞。如果消息 D 和 D' 构成内部碰撞，那么对任意的位串 Y 都有 $MAC_K(D \| Y)=MAC_K(D' \| Y)$。这就构成了一种伪造，在攻击者得到 DP Y 的 MAC 后，能够预测 $D' \| Y$ 的 MAC。这种伪造依赖于消息的特殊格式，可能对许多应用没有威胁。但是，这种攻击的扩展版本在消息格式方面有更大的灵活性。这种攻击需要一个选择消息、大约 $2^{n/2}$ 个已知消息和 2^{n-m} 个选择消息。

通过以下方式可避免生日攻击：在要处理的消息前面加上一个序列号消息块，使得 MAC 算法是带状态的。这要求在 MAC 算法实现中要保证每个序列号在密钥生命周期内，MAC 计算过程中只用一次。这种要求并不是在所有环境下都可行。

——**捷径密钥恢复**：基于内部碰撞的密钥恢复攻击适用于某些 MAC 算法。目前还没有关于本文件中 MAC 算法的捷径密钥恢复攻击。

安全证明

若如下假设成立，那么 MAC 算法 1 被证明是安全的：

使用密钥为初始值 IV 和附加常数的轮函数 ϕ 是一个伪随机函数。

注 1：伪随机函数是一种带密钥的函数，对于不知道密钥的敌手，它的表现和一个随机函数相仿（也就是说，伪随机函数和随机函数很难区别）。

若如下假设成立，那么 MAC 算法 2 被证明是安全的：

使用密钥为初始值 IV 的轮函数 ϕ 是一个强 MAC 算法(也就是说,很难预测它的输出)且迭代杂凑函数是抗第二原像的。

若如下假设成立,那么 MAC 算法 2 被证明是量子安全的:

使用密钥为初始值 IV 的轮函数 ϕ 是一个量子伪随机函数。

注 2:量子伪随机函数是一种带密钥的函数,对于不知道密钥的量子敌手,它的表现和一个随机函数相仿。

附 录 B
（规范性）
对象标识符

本附录列出了本文件规定的 MAC 机制的对象标识符。

```
MechanismsUsingADedicatedHashFunction {
    iso(1) standard(0) message-authentication-codes(9797) part(2)
        asn1-module(0) mechanisms-using-a-dedicated-hash-function(0) version2(2)}

DEFINITIONS AUTOMATIC TAGS ::= BEGIN

EXPORTS ALL;

IMPORTS
    OID,ALGORITHM,HashFunctions,HashFunctionAlgs
        FROM DedicatedHashFunctions {iso(1) standard(0)
            hash-functions(10118) part(3)
            asn1-module(1) dedicated-hash-functions(0)};

-- OID assignments
-- =========

is9797-2 OID ::= {iso standard message-authentication-codes(9797) part(2)}

id-mac-1 OID ::= {is9797-2 macAlgorithm-1(1)}
id-mac-2 OID ::= {is9797-2 macAlgorithm-2(2)}
id-mac-3 OID ::= {is9797-2 macAlgorithm-3(3)}

-- MAC algorithm identifier type and the set of recognized MAC algorithms
-- ================================================================
AlgorithmIdentifier {ALGORITHM:IOSet} ::= SEQUENCE {
    algorithm ALGORITHM.&id({IOSet}),
    parameters ALGORITHM.&Type({IOSet}{@algorithm}) OPTIONAL
}

MessageAuthenticationCode ::= AlgorithmIdentifier{{MacAlgorithms}}

MacAlgorithms ALGORITHM ::= {
    {OID id-mac-1 PARMS MacParameters} |
    {OID id-mac-2 PARMS MacParameters} |
    {OID id-mac-3 PARMS MacParameters} ,
```

```
    ... -- additional algorithms expected --
}

-- MAC parameter type definitions
-- ==========================

-- The optional parameters may be agreed upon by other means

MacParameters ::= SEQUENCE {
    dhfAlgo HashFunctions OPTIONAL,
    m INTEGER (1..MAX)
}

END -- MechanismsUsingADedicatedHashFunction --


END - MessageAuthenticationCodes --
```

附 录 C
（资料性）
测试向量

C.1 概述

本附录提供了 MAC 算法 1、算法 2 和算法 3 的计算示例，采用 GB/T 32905—2016 的 SM3 密码杂凑算法。表 C.1 包含了序号为 1～9 的输入位串。对于 MAC 算法 1 和算法 2，本附录给出了计算 MAC 值的 9 个示例；对于 MAC 算法 3，只给出了表中的前 5 个示例。

在整个附录中，采用 GB/T 1988—1998 规定的 7 位编码字符对输入数据位串进行编码。

表 C.1 用于测试的输入位串

序号	输入位串
1	“”（空位串）
2	“a”
3	“abc”
4	“message digest”
5	“abcdefghi jklmnopqrstuvwxyz”
6	“abcdbcdecdefdefgefghfghighijhijkijkljklmklmnlmnomnopnopq”
7	“ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789”
8	“1234567890”重复 8 次所得的长为 80 的字符串
9	“a”重复 1,000,000 次得到 1 兆长的字符串

两个 128 位的密钥如下：

密钥 1＝00112233445566778899AABBCCDDEEFF

密钥 2＝0123456789ABCDEFFEDCBA9876543210

对于本附录中的示例，选取 $m=L_H/2$，其中 L_H 取可选最大值。对于采用 SM3 密码杂凑函数的 MAC 算法，即：$m=128$。

C.2 MAC 算法 1（MDx-MAC）

对于采用 SM3 密码杂凑函数的 MAC 算法 1，测试向量见表 C.2 和表 C.3。

表 C.2 MAC 算法 1 的测试向量（密钥 1）

密钥 1：00112233445566778899AABBCCDDEEFF	
序号	128 位 MAC 值：下列位串的最左侧的 128 位
1	4AECB47B01AF2E0B420FE3C9B24A0FD8C5EF6D82F41C8009B9133E398BD4A8B5
2	AC5FE9BA773DAF4A9235C858BFC03E6863052E3C58B0C125AED70A999AB775E7
3	F321D3C152400A44CB98D8096084823ADFDDBB57A3B2E947A4024B581020E404
4	CFF373D995C6112F16A585184595E0A99FAD55AF4A55494E37D47B6766C28A05

表 C.2　MAC 算法 1 的测试向量(密钥 1)(续)

密钥 1:00112233445566778899AABBCCDDEEFF	
序号	128 位 MAC 值:下列位串的最左侧的 128 位
5	2A139ABD14AFD36C4C483816BA188A2444B3B5B08D95E2B64FA946A1604128AE
6	389066AB696C54D77BBD168A1B5D1D177DB6233751E3994B0804C3F1CC378075
7	D0DAD6CB29EE0D393547D1F716BF83111E00A6513FA213DB522F9E0E1AFF96B4
8	9C1C16CEE95C7BBFEE01C054A72169A0F99A1472B9060074AA5366CC18EBB0D4
9	ED73BAE5E7FA51284FC2704C98DA12FB20992564BF1D8976970E9E07586D1783

表 C.3　MAC 算法 1 的测试向量(密钥 2)

密钥 2:0123456789ABCDEFFEDCBA9876543210	
序号	128 位 MAC 值:下列位串的最左侧的 128 位
1	80DFB76DBF659EDBCA4E703FF92D522B00BECE4B8D698BF644837F2F88EEC613
2	8F319C8AC77E38786B6DDA88CEE90388597CC2FE7772FA3BCF14DA16F1D11C50
3	01B474B0BA28ECBA6EF0A7A1339C96A734351D59D4BB5EC192EDC59BBF4FBC4B
4	9841877F2A1E0E04813B4258ED16FDD080476CC7D6B2C51D7BD2EEBD7F93FA5D
5	57CD60810A8A106EFE2B4B16F3D16F4EF89CB49E0EC5D9C07300BC127603DD35
6	1B3BEF507A9D57B95D2B41E8A56EE88BDC8539D04B224D6BEDA2CE1098E4CBB2
7	271AF8DC5A4DA1AF692306CC198EFE2BC9923BD6D382FCE5C1284D19E6A2721C
8	BFDD8620B52EB6FF3FCF74EF0EE93652161C816DEE86DDF5C822867A2901ADE4
9	883827DE1F419CC0E8502F962A052DF5913212A9AD2DA3BFF00DA80A7CA8DFE4

C.3　MAC 算法 2(HMAC)

对于采用 SM3 密码杂凑函数的 MAC 算法 2,测试向量见表 C.4 和表 C.5。

本附录的示例使用 C.1 给出的 128 位密钥 1 和密钥 2 生成。实际应用中,密钥长度不小于 L_2 位。

表 C.4　MAC 算法 2 的测试向量(密钥 1)

密钥 1:00112233445566778899AABBCCDDEEFF	
序号	128 位 MAC 值:下列位串的最左侧的 128 位
1	C8E4E95012EB3D449B5DD0691947986E469E08A3506BB55CCB94A96EBFADA654
2	5FD9F7568A24C438F14B7A22E799B0689FE053ABB76D316202E3C9D10E9EEBE2
3	0933617A88D312F6F9FB4B5F200E31A64D655E92F7FA2A43F55DFEEB8AB6788D
4	9C9A22E8B5797B82CFF9BABA56893CC1D75811C334D198F3AF43401740B824F7
5	A51CE58C52AE29EDD66A53E6AAF0745BF4FEDBDE899973B2D817290E646DF87E
6	DC813339153491AD81477754EB3DF00DBB3CC3E6A69F9CACCE737DB7E61342FF
7	BCA6FA751AECAC5BA3AC49963F6A58F7C2293C6E6923802BC52117A741A49FEE

表 C.4　MAC 算法 2 的测试向量(密钥 1)(续)

密钥 1:00112233445566778899AABBCCDDEEFF	
序号	128 位 MAC 值:下列位串的最左侧的 128 位
8	25E034DF9A3AC81599C233440CA6F68F38CA5166438BFA620210EC2F59880C0D
9	34DB1B0452359EA54DA16932E42A662BE88C19C5AD4FE9073867C05A92752024

表 C.5　MAC 算法 2 的测试向量(密钥 2)

密钥 2:0123456789ABCDEFFEDCBA9876543210	
序号	128 位 MAC 值:下列位串的最左侧的 128 位
1	F14B797B559216B73D3816ADFB790250AF3F21198A1AE867123762BB63A00945
2	5BD1836B97C74F88A77BC309E77A269481F53BE9D5C4CE1E40B1C50FE574762E
3	28D8A61BE67D8BF7652C4EDA7092B612F88BE62184F55005C57DDF076E764199
4	E0ACCC4DA77E77D135F17F5CA1EE3E600DAB444FC23ADD6F7E6A54E1B34B26BC
5	429D9030B1D992AD8198E01C13141C2859A913D69DE00CCE9E4A60F00BF276CB
6	AAB294F80562AB234E6226BF7FC3B03F839C7759E60F69735B7E99E50EB94A24
7	08F457B37E5E062AFAFB24DE8D48B92246F1788BAAD4D7B3D11E5F627E33A0D3
8	9F85C779D718A33BDEC2D6E0C1F280FE6A8C12FF2521530A44D168DD4080BC14
9	ED3057AB0DB1E826240FCF8E8760C3DB9338E9AABDAD8B11BB0C040D73E74441

C.4　MAC 算法 3(MDx-MAC 的变种)

对于采用 SM3 密码杂凑函数的 MAC 算法 3,测试向量见表 C.6 和表 C.7。

表 C.6　MAC 算法 3 的测试向量(密钥 1)

密钥 1:00112233445566778899AABBCCDDEEFF	
序号	128 位 MAC 值:下列位串的最左侧的 128 位
1	B7DA221372494909407DAE9BAA89EC6F97B4B5FF44453C1C634B0CBB642612F2
2	C9333F511344032C70CA41DAB3E335A2AB28DF5B933344E90B4EFC91BA90DC6B
3	336202E1213B63AF5A141FDFCD2B2213C6EDC56CEE7EC12B8A8878182C530FB3
4	E6A3BC8769F5DC27131F4799AD710C8B933347FA65D3A7EFDF354472B81F0238
5	4602CD8E9DA630CB863A14362B0D7DD025D2F5855ABC283DFEEFB1E720B32467

表 C.7　MAC 算法 3 的测试向量(密钥 2)

密钥 2:0123456789ABCDEFFEDCBA9876543210	
序号	128 位 MAC 值:下列位串的最左侧的 128 位
1	94160AE33F288BFB23235E507CB9C9E46C4E074E91E7F4962F1F09C341BE7A53
2	D2D2A779931DFC33B5231422A243C78F012264BA933B1DF1D34D6D6AF2FD34A9
3	1EF7683A13569F6F6596C31E16811B54A51073848527573DB6F685F3AE8D0BC7
4	C60D7367FC8F6715D563A52C50D1DD9FA1E2EDBAB832F4C2D8873A8EED77264B
5	605BF340130372153AA4BBDC9C10E7783D0143C6B7E1D186072FF062D04D8DDD

参 考 文 献

[1] GB/T 1988—1998 信息技术 信息交换用七位编码字符集

[2] GB/T 15852.1—2020 信息技术 安全技术 消息鉴别码 第1部分:采用分组密码的机制

[3] GB/T 18238.1—2024 网络安全技术 杂凑函数 第1部分:总则

[4] ISO/IEC 18031:2011 Information technology—Security techniques—Random bit generation

[5] BPRENEEL, PCVAN OORSCHOT.'MDx-MAC and building fast MACs from hash functions', Advances in Cryptology, CRYPTO′95, LNCS 963, D. Coppersmith, Ed., Springer-Verlag, 1995, pp.1-14.

[6] MBELLARE, RCANETTI, HKRAWCZYK, 'Pseudorandom functions revisited: The cascade construction and its concrete security', Proc.37th Annual Symposium on the Foundations of Computer Science, IEEE, 1996:514-523.

[7] MBELLARE, RCANETTI, HKRAWCZYK.'Keying hash functions for message authentication', Advances in Cryptology CRYPTO′96, LNCS 1109, N. Koblitz, Ed., Springer-Verlag, 1996: 1-15.

[8] FSONG AYUN.'Quantum security of NMAC and related constructions', Advances in Cryptology, CRYPTO'17, LNCS 10402, J.Katz, Ed., Springer-Verlag, 2017:283-309.

[9] AHOSOYAMADA TIWATA.'On Tight Quantum Security of HMAC and NMAC in the Quantum Random Oracle Model', Advances in Cryptology, CRYPTO′21, LNCS 12825, T. Malkin, Ed., Springer-Verlag, 2021:585-615.

购买者:豪密科技
时　间:2025-04-08
定　价:43元

GB/T 15852.2-2024

中　华　人　民　共　和　国

国　家　标　准

网络安全技术　消息鉴别码

第 2 部分:采用专门设计的杂凑函数的机制

GB/T 15852.2—2024

*

中国标准出版社出版发行

北京市朝阳区和平里西街甲 2 号(100029)

北京市西城区三里河北街 16 号(100045)

网址:www.spc.net.cn

服务热线:400-168-0010

2024 年 9 月第一版

*

书号:155066·1-77665