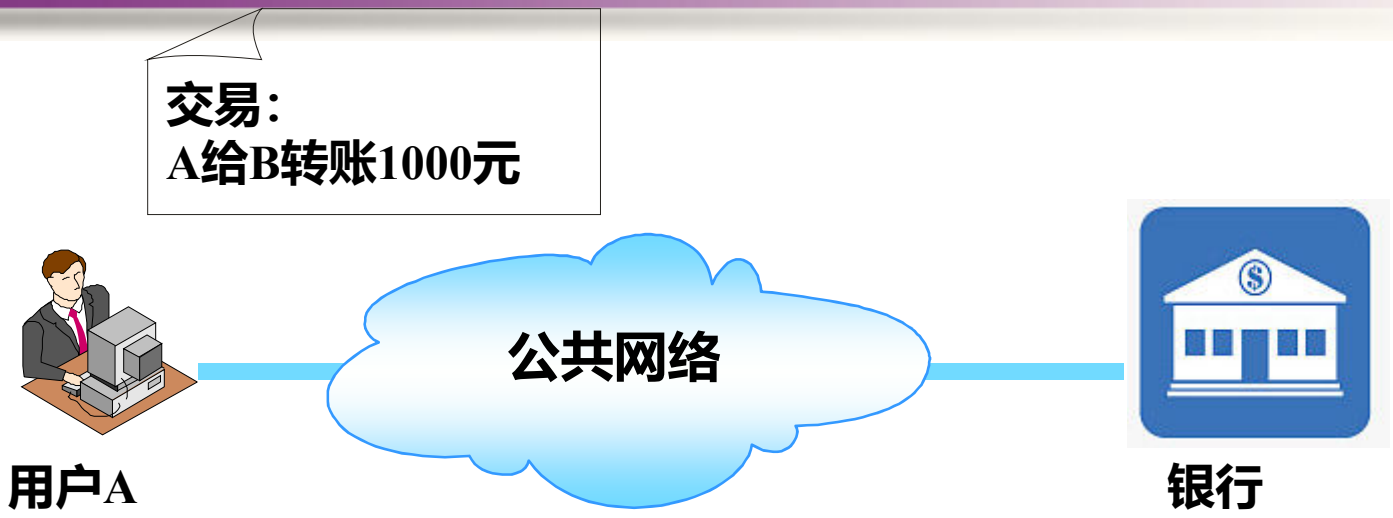
消息认证码及认证加密算法

清华大学计算机系
于红波
2023年4月26日

引言：网络通信威胁



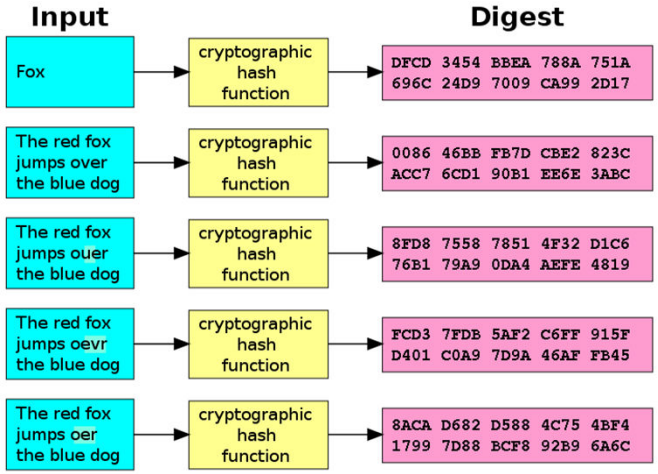
银行收到交易信息，需要考虑下面两个问题：

1. 信息在传输中有没有被修改过？
2. 交易是否可靠，交易信息真的是由用户A发出的吗？

需要对消息完整性和消息源进行认证

内容回顾 –密码Hash函数

密码学Hash函数: 将任意长度的消息压缩成固定长度的摘要



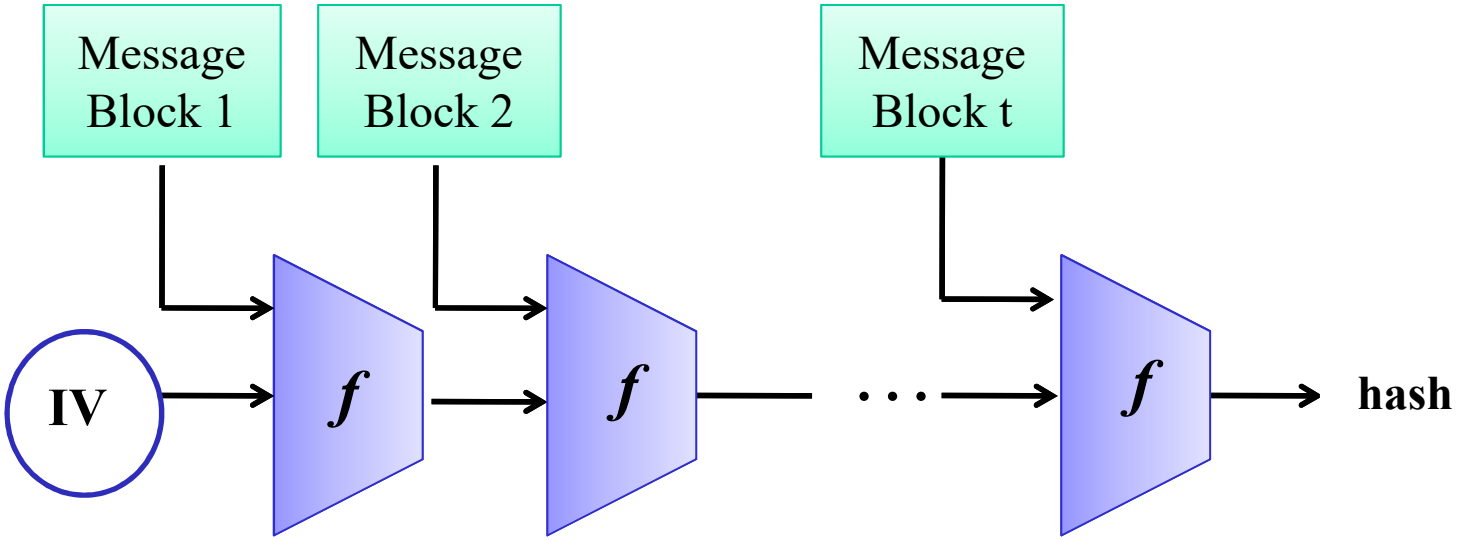
三个安全属性:

1. 抗原像攻击：对任意给定的Hash值h, 找到满足H(y)=h的y在计算上不可行
2. 抗第二原像攻击：对任意给定的消息x, 找到满足$y \neq x$且H(x)=H(y)的y在计算上是不可行的
3. 抗碰撞攻击：找到任何满足H(x)=H(y)的消息对(x, y)在计算上是不可行的

内容回顾-MD迭代结构

Merkle-Damgård(MD)迭代结构： 设消息分组x= $x_1x_2,\ldots x_t$

h_0=IV; $h_i=f(h_{i-1},x_i)$, $1\le i\le t$; $hash(x)=h_t$

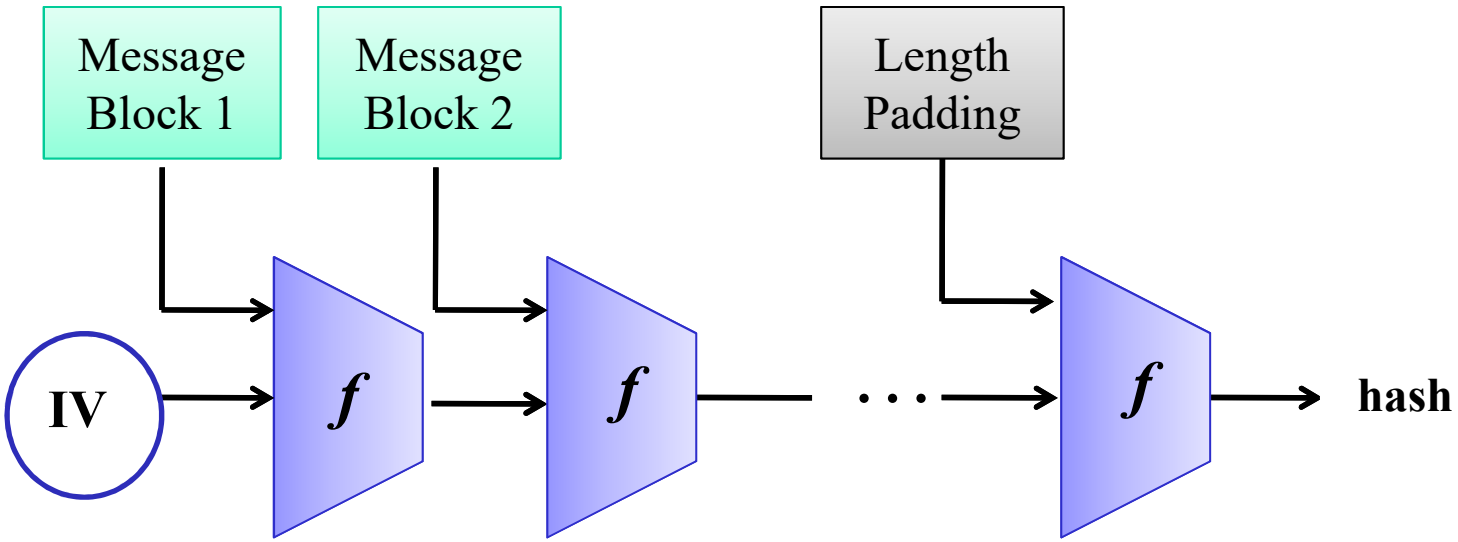


IV是固定值；压缩函数*f*为单向函数

内容回顾-MD增强结构

MD增强结构： 设消息分组x= $x_1x_2,\ldots x_t$； 长度填充分组x_{t+1}

h_0=IV； $h_i=f(h_{i-1},x_i)$, $1\le i\le t+1$； $hash(x)=h_{t+1}$



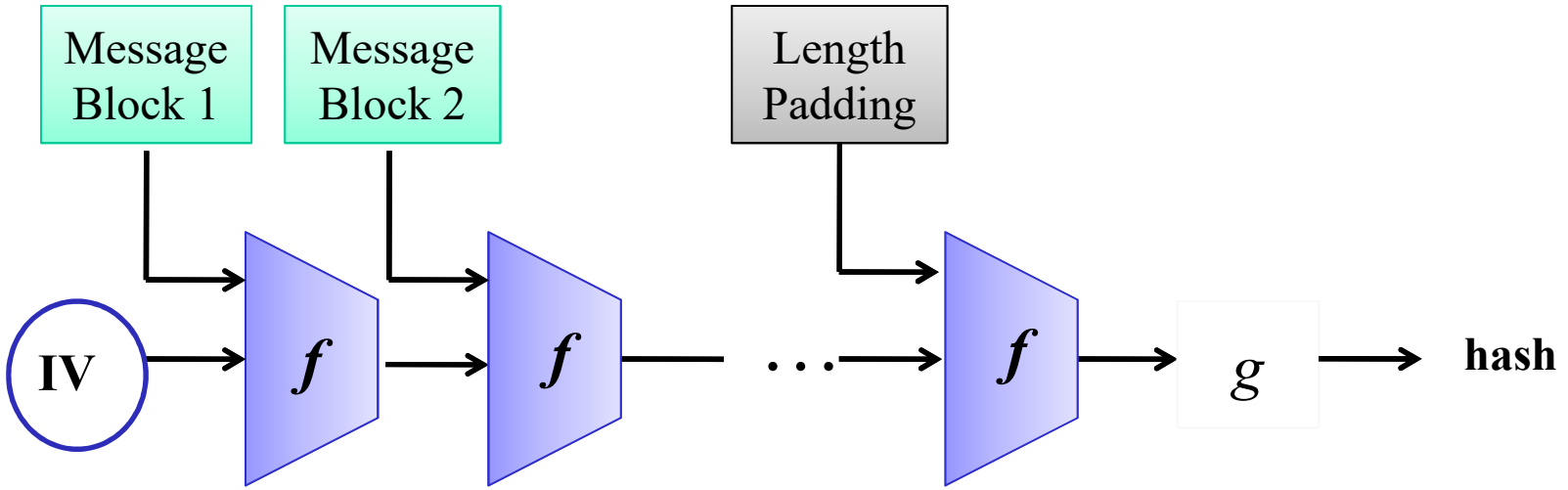
大多数通用的Hash函数标准都使用MD增强结构：

国际标准MD5,SHA-1, SHA-2； 我国标准SM3

内容回顾-完全MD结构

完全MD结构： 设消息分组x= $x_1x_2,\ldots x_t$； 长度填充分组x_{t+1}

h_0=IV； $h_i=f(h_{i-1},x_i)$, $1\le i\le t+1$； $hash(x)=g(h_{t+1})$



g为一置换函数或者压缩函数

教学内容

1. 认证的功能
2. 消息认证码的构造
3. 认证加密算法

认证的功能

- 用来验证消息的完整性(data integrity)
  - 未被篡改、插入、删除
- 用来验证消息的来源（真实性,data origin authentication）
  - 的确是由它所声称的实体发出的
- 用来验证消息的顺序性和时间性(uniqueness and timeliness)
  - 未重拍、重放、延迟 （Transaction authentication）
- 用来验证消息的不可否认性(non-repudiation)
  - 防止通讯双方中的某一方对所传输消息的否认

消息完整性认证

- 加密算法能否提供数据来源保护?

  A、B双发共享一个密钥K，若B收到密文，用 K解密，解密后的明文如果是“有意义”的， 则证明消息来源于A?

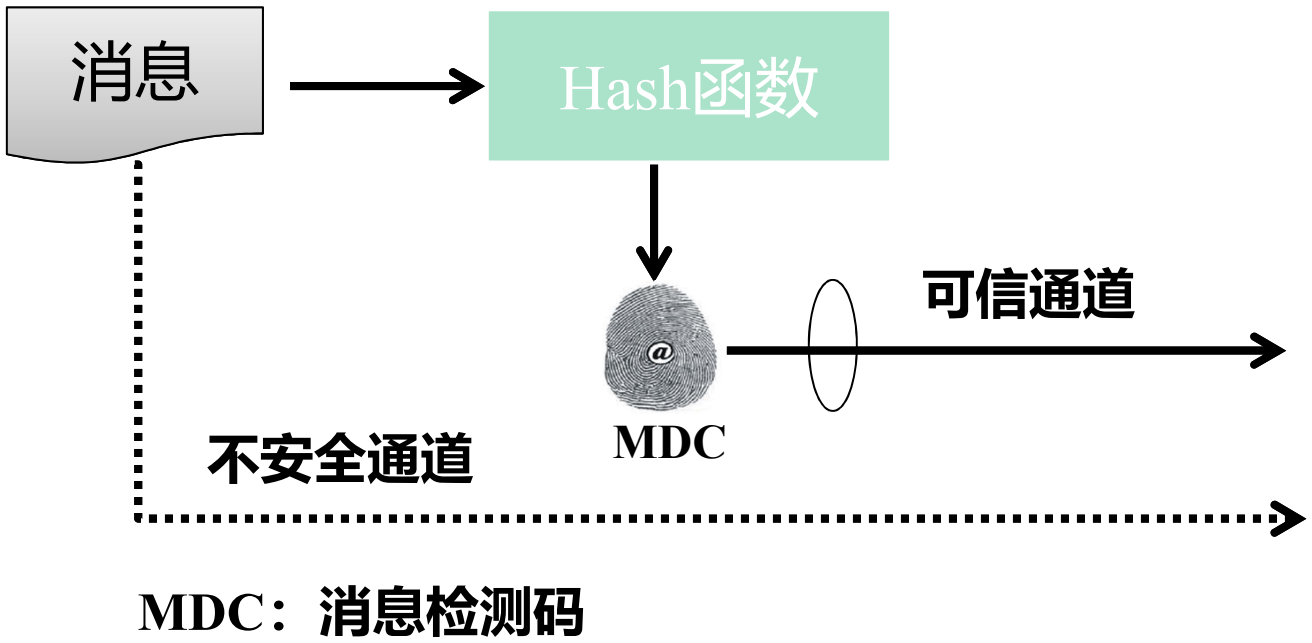
- 加密算法不能提供消息完整性认证
  - ECB模式密文重排
  - 加密随机数据
  - 使用序列密码或分组密码的OFB，CFB模式加密

消息完整性认证

- 加密算法不能提供消息完整性认证
  - ECB模式密文重排
  - 加密随机数据
  - 使用序列密码或分组密码的OFB，CFB模式加密
- 如何实现消息的完整性认证
  - 方法1：Hash函数+可信通道
  - 方法2：Hash函数+加密算法
  - 方法3：消息认证码 (Message authentication Codes)

消息完整性认证

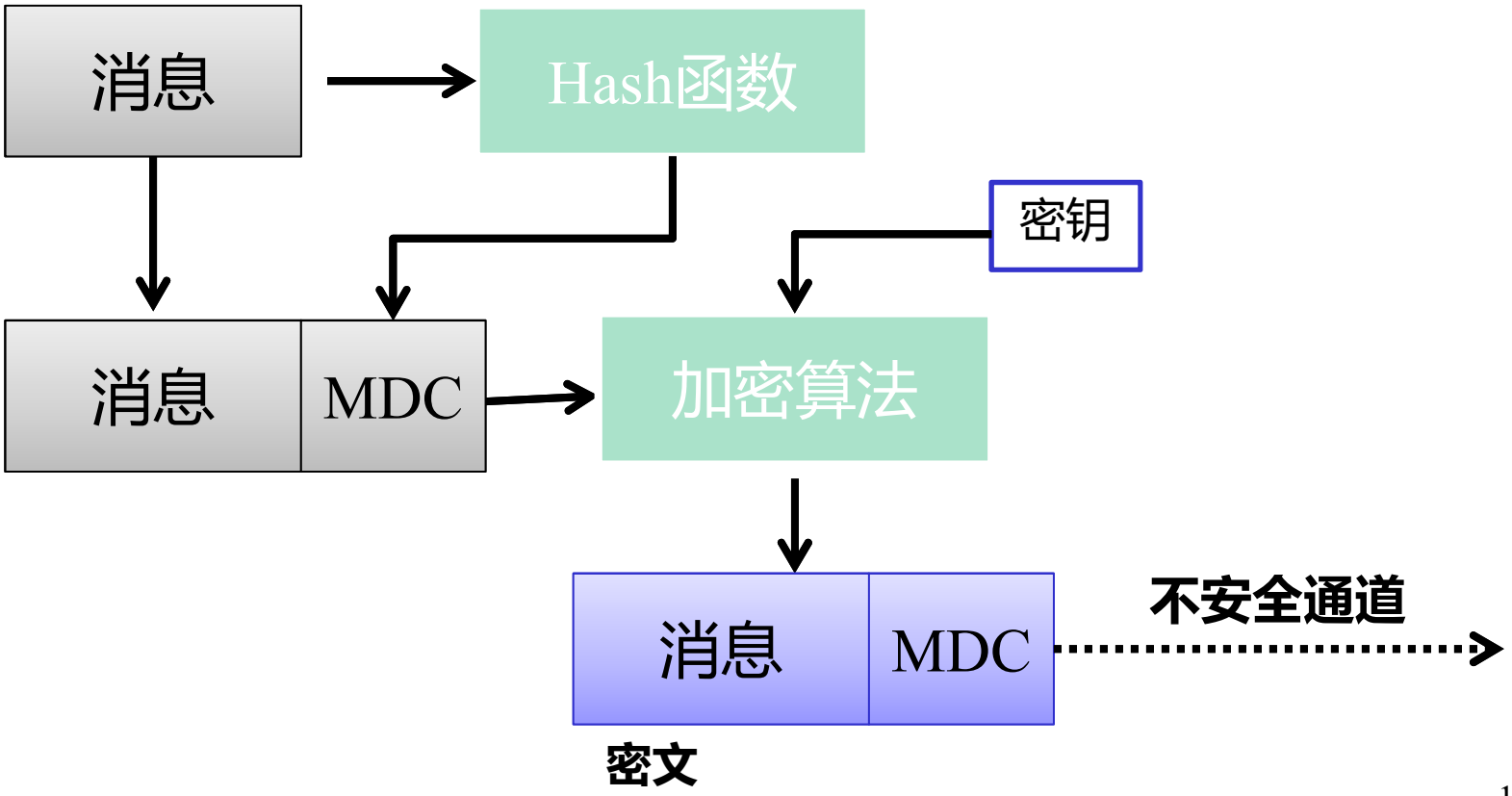
- 方法1：Hash函数+可信通道



  - 可信通道：电话、数据载体（优盘、纸张、传真）、公共媒体

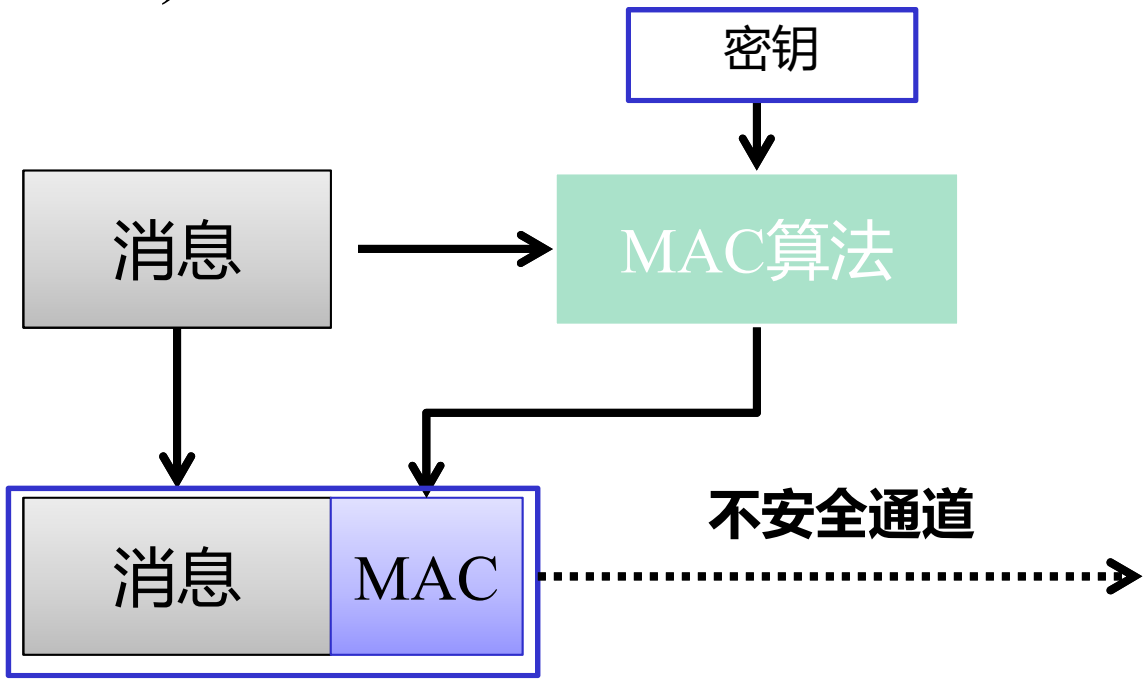
1.1 消息完整性认证

- 方法2：Hash函数+加密算法



消息完整性认证

- 方法3：消息认证码 (Message Authentication Codes)

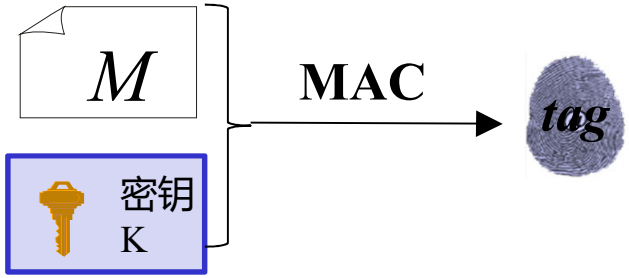


消息真实性认证

- 提供消息真实性认证的方法
  - 消息认证码（message authentication codes)
  - 数字签名(digital signature schemes)
  - 在加密之前附加一个秘密的认证码（认证加密）

消息认证码MAC

- 消息认证码（MAC）是在密钥的控制下将任意长的消息映射为一个固定长度的短数据分组

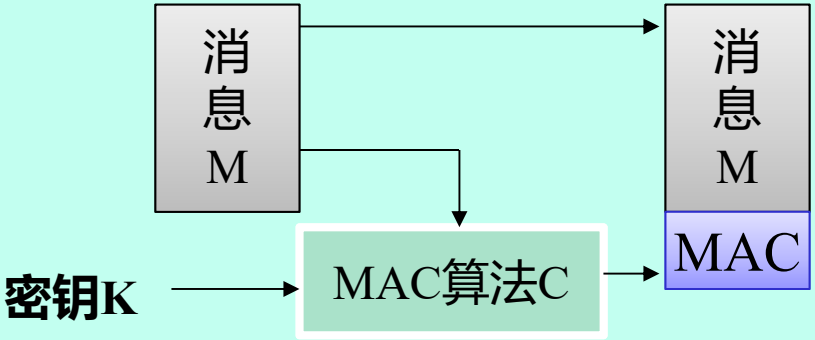


- MAC是消息和密钥的函数：$MAC=C_K(M)$
- 功能：完整性认证、消息源认证
- 应用：
  - 网络协议：SSL, TLS, SSH, SMNP, IPsec等

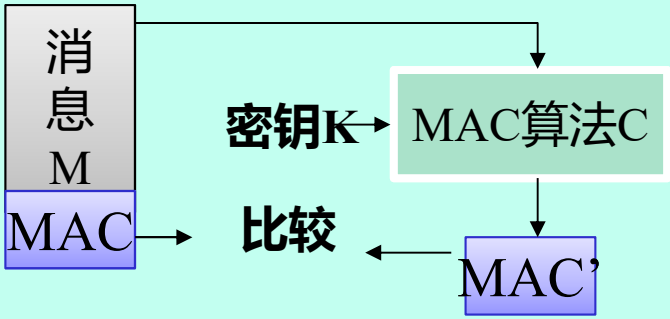
消息认证码MAC——认证过程

- 通信双方共享密钥K

对消息M，计算$MAC=C_k(M)$



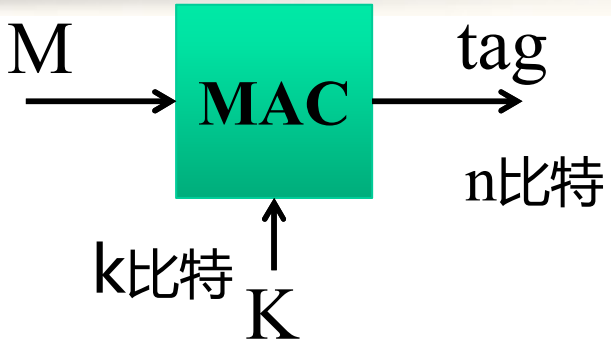
对消息M，计算$MAC'=C_k(M)$，比较MAC= MAC'?



MAC提供数据完整性以及消息来源认证、但不具有不可否认性

安全的MAC

安全参数：密钥K的长度为k, 输出标签长度为n



伪造攻击：产生一个新的消息-标签对(m, tag)，满足$tag=MAC_k(m)$

安全界

$k \le n$: 猜2^k密钥，k/m消息-标签对进行验证

$K>n$: 直接选择一个消息，猜测其对应的标签；成功概率2^{-n}

教学内容

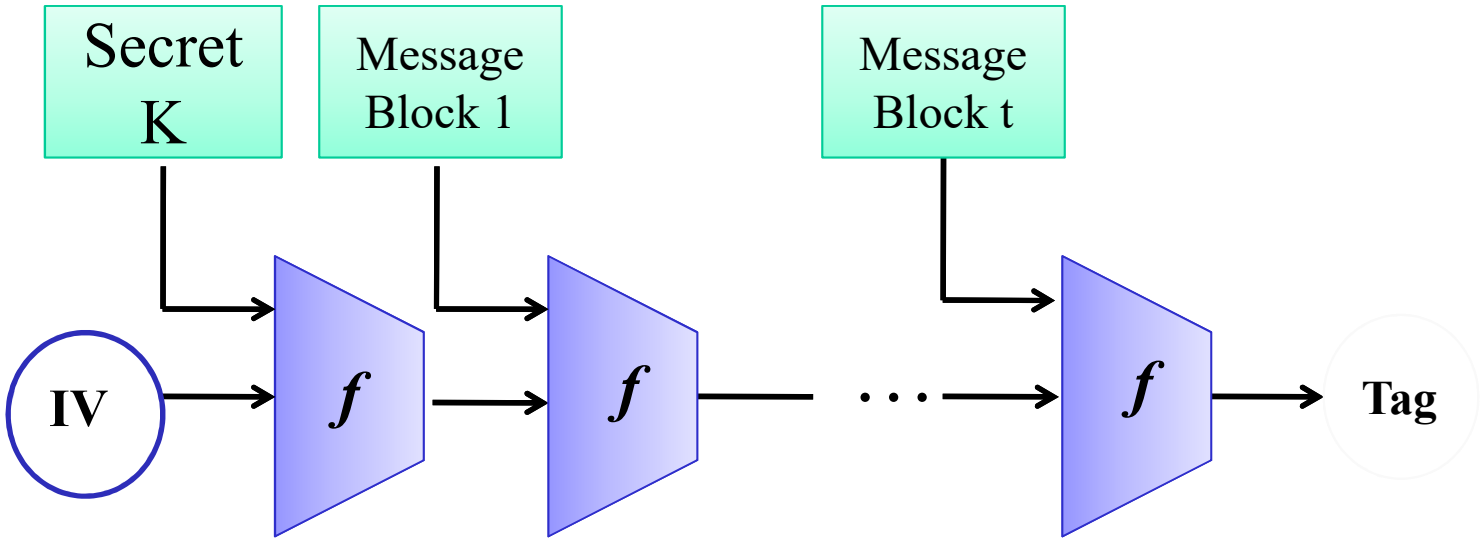
1. 认证的功能
2. 消息认证码的构造
3. 认证加密算法

MAC分类

- 基于分组算法
  - 串行的：CBC-MAC, XCBC, OMAC, CMAC等
  - 并行的：PMAC, XOR-MAC 等
  - 认证加密的：OCB, CCM等
- 直接设计的MAC：MAA
- 基于杂凑算法
  - Secret-Prefix method, Secret-suffix method, secret-envelope method, MDx-MAC, HMAC, NMAC
- 基于Universal Hashing
  - UMAC, MMH, Poly1305-AES, GCM/GMAC 等

密钥前缀方案：基于MD迭代结构

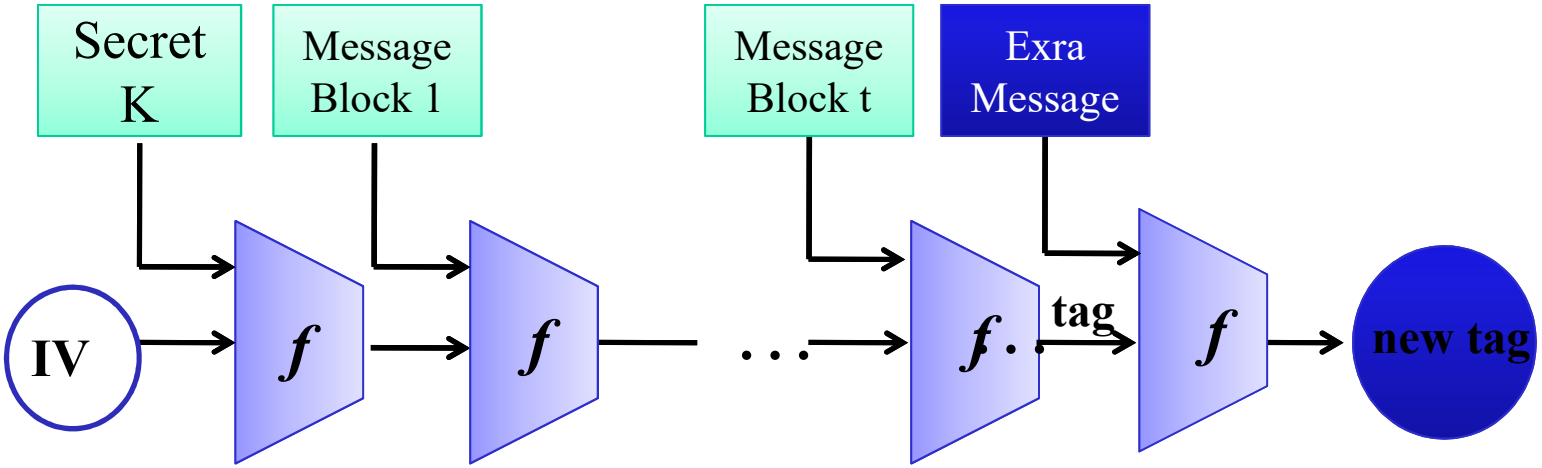
密钥前缀： $MAC_k(M)=H(K||M)$



假设密钥作为第一个分组，该方案是否安全？

密钥前缀方案：基于MD迭代结构

密钥前缀： $MAC_k(M)=H(K||M)$

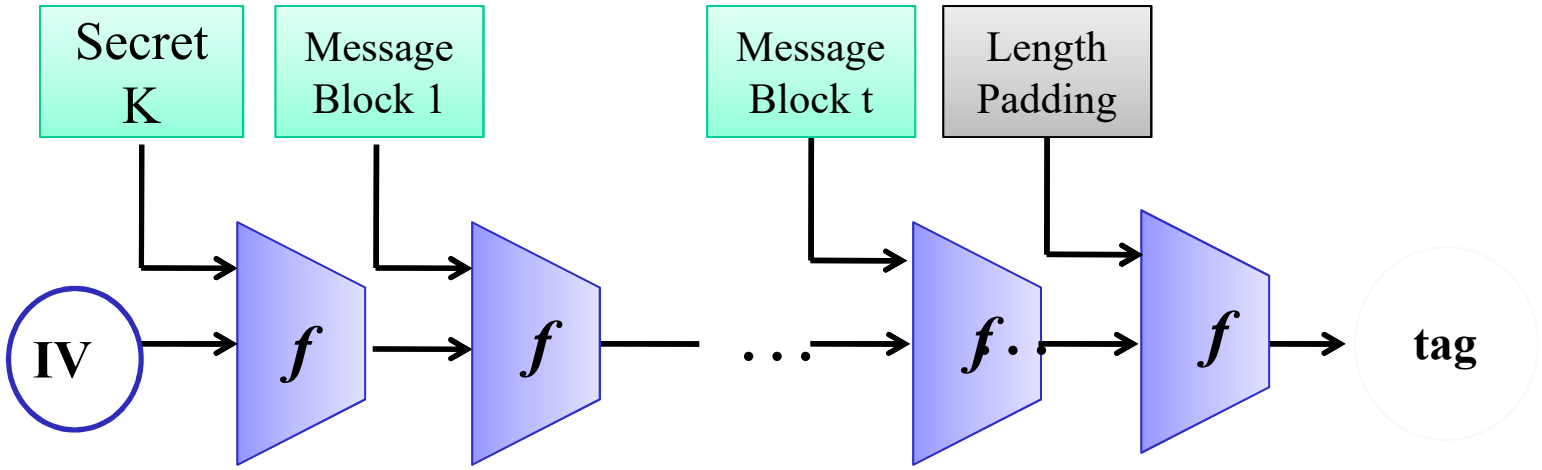


缺点：长度扩展攻击

攻击者可以在消息的尾部增加一个新的消息分组，伪造一个新的MAC: (M||S, newtag)

密钥前缀方案：基于MD增强结构Hash

密钥前缀： $MAC_k(M)=H(K||M)$

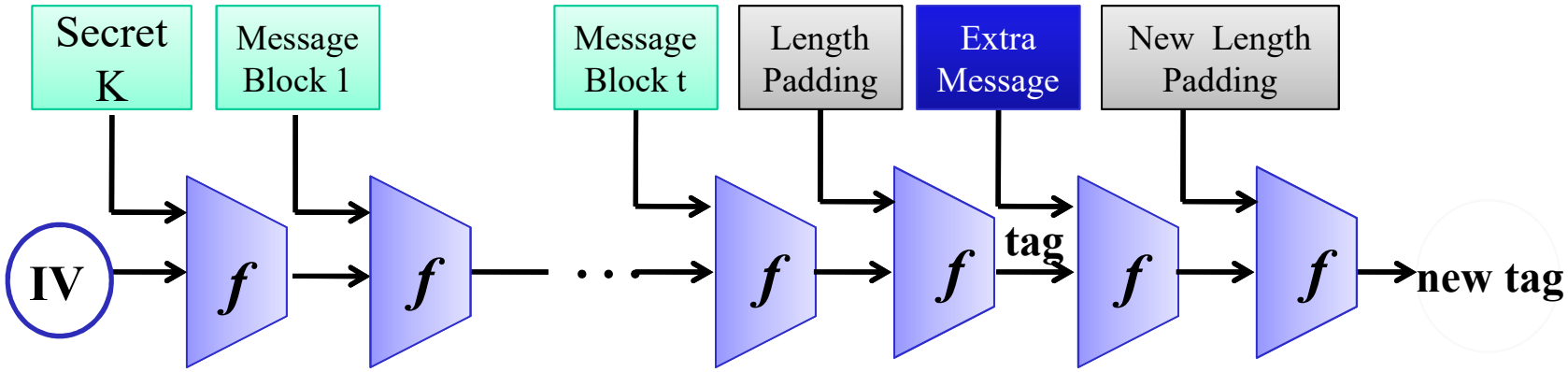


问题：

对基于带长度填充的MD迭代结构，该方案是否安全？

密钥前缀方案：基于MD增强结构Hash

密钥前缀： $MAC_k(M)=H(K||M)$

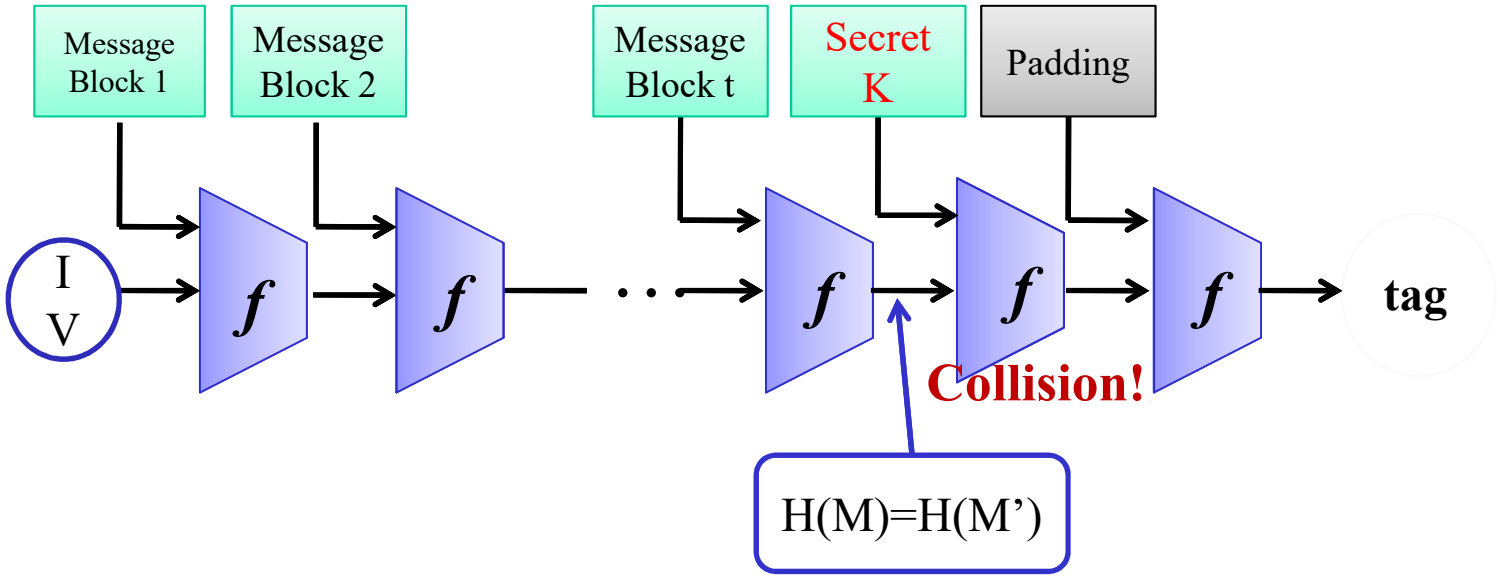


长度扩展攻击

攻击者仍然可以在消息的尾部增加一个新的消息分组，伪造一个新的MAC: (M||pad||S, newtag)

密钥后缀方案

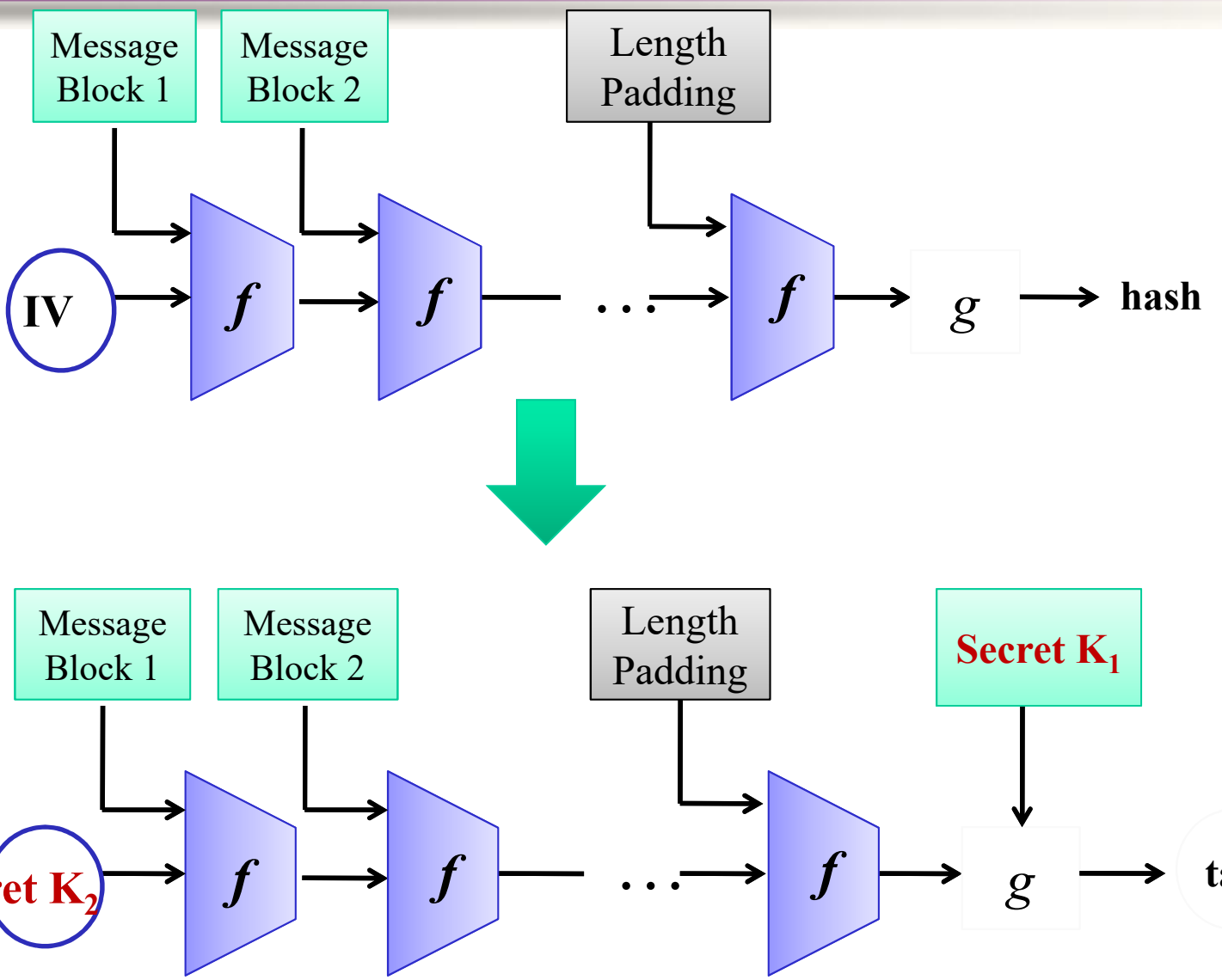
密钥后缀： $MAC_k(M)=H(M||K)$



攻击：使用Hash函数的碰撞，可以伪造密钥后缀的MAC

- 通过生日攻击算法寻找一对等长消息的碰撞的消息(M, M')：复杂度$2^{n/2}$次离线运算
- 询问M的MAC值tag，伪造新的消息-MAC对（M', tag）

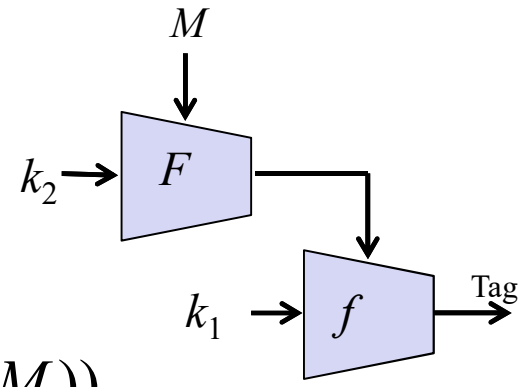
基于完全MD迭代结构MAC



NMAC 和HMAC

NMAC & HMAC 1996年由Bellare等人提出，并给出了安全性证明; 2006年对安全性证明进行更新

$$NMAC_{(k_1,k_2)}(M) = f_{k_1}(F_{k_2}(M))$$



HMAC （Hash based MAC）：

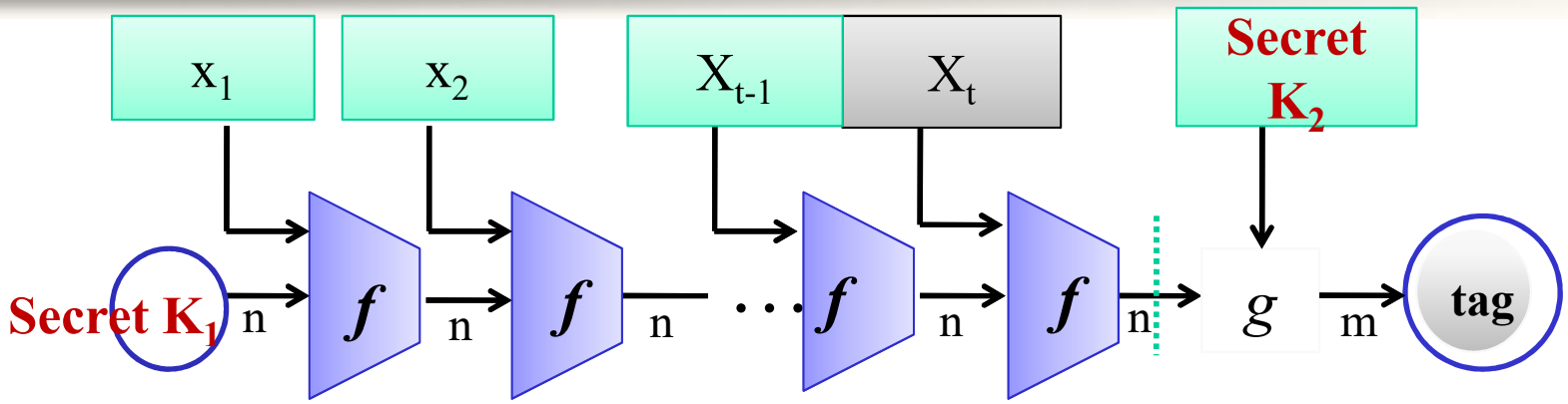
$$H_K(M) = H(\overline{K} \oplus opad \,||\, H(\overline{K} \oplus ipad \,||\, M))$$

- 其中 $\overline{K} = K \,||\, 0...0$ 为一完整消息分组
- opad和ipad为常数

HMAC Win!

HMAC是国际标准：ANSI, NIST, IETF, ISO
广泛应用于网络安全协议：SSL, IPSec, TLS, POP3, IMAP

对基于MD迭代结构的MAC的通用伪造攻击



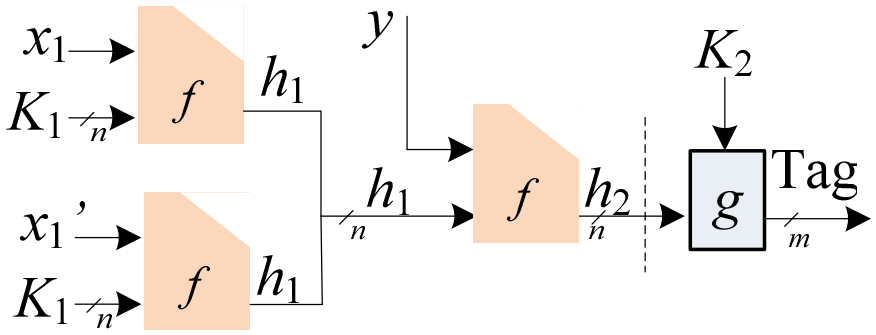
$$M = x_1 \| \ldots \| x_t, K = K_1 \| K_2$$

$$h_0 = K_1; h_i = f(h_{i-1}, x_i), 1 \le i \le t; MAC(M) = g(K_2, h_t)$$

- 内部碰撞： g函数之前的碰撞
- 外部碰撞： 输出Tag碰撞， g函数之后的碰撞

对基于MD迭代结构的MAC的通用伪造攻击

命题1：一个内部碰撞能够用来伪造MAC，仅需要一个选择消息和一次MAC询问



证明：设x_1和x_1'是一对内部碰撞，即$f(K_1, x_1)= f(K_1, x_1')$

- 询问：$x_1 \| y$的MAC，其值为tag
- 伪造：输出伪造消息-MAC对（$x_1' \| y$, tag）

对基于MD迭代结构的MAC的通用伪造攻击

命题2： 设h是一个基于MD迭代结构的MAC算法。需要u个已知消息的消息询问和v个选择消息的询问能够找到h的一对内部碰撞。

u和v的期望值如下：

$$u = \sqrt{2}\,2^{n/2}$$

当 g是置换函数（n=m）, 外部碰即是内部碰撞, v=0;

当g是压缩函数（m<n）, v近似于

$$2 \cdot 2^{n-m} + 2\lceil n/m \rceil$$

对基于MD迭代结构的MAC的通用伪造攻击

命题2的证明：

$u = \sqrt{2}2^{n/2}$ **个已知消息-MAC对产生1个内部碰撞和2^{n-m}个外部碰撞（生日攻击算法）：(x_i, x_i')**

如何将内部碰撞从外部碰撞中区分开？

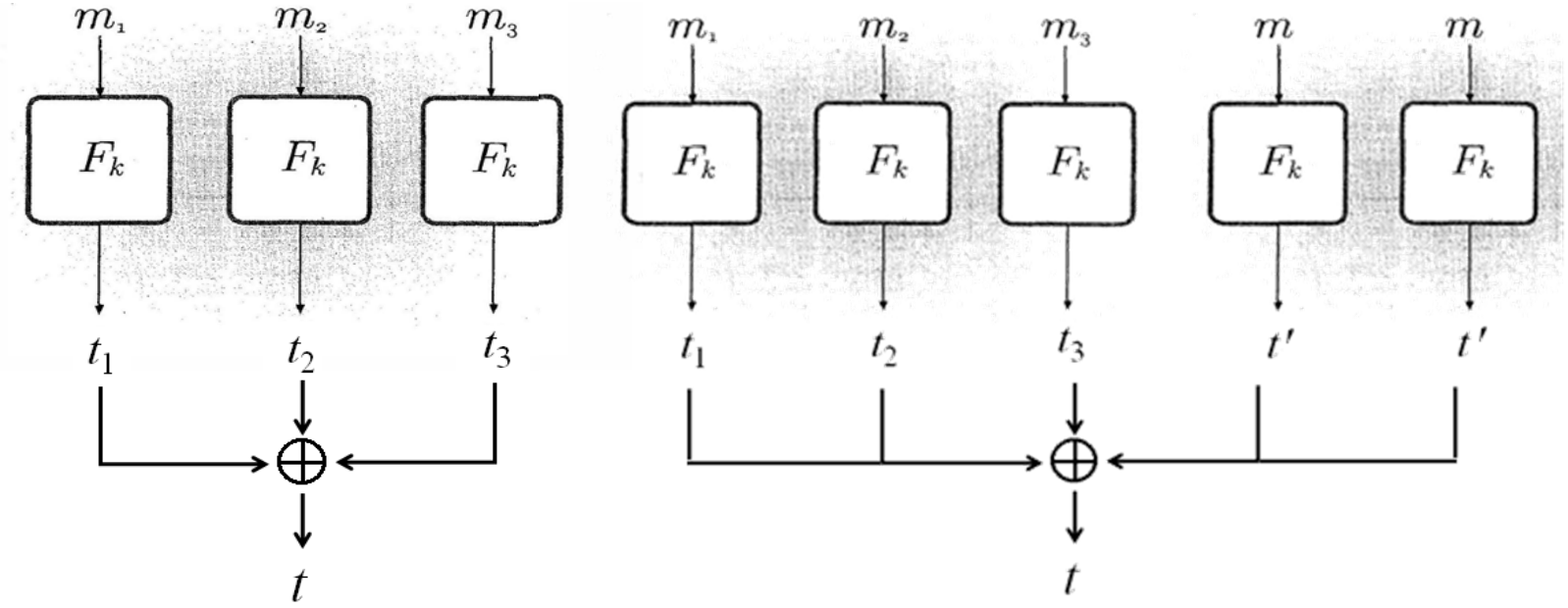
- ✓ 当 g是置换函数（n=m），外部碰即是内部碰, v=0;
- ✓ 当g是压缩函数（m<n）

区分器: 询问消息对$(x_i \| y, x_i' \| y)$的MAC值，看其是否还是碰撞：

1. 内部碰撞通过概率为1
2. 外部碰撞通过概率为$2^{1/(m)}$

基于分组密码的消息认证码

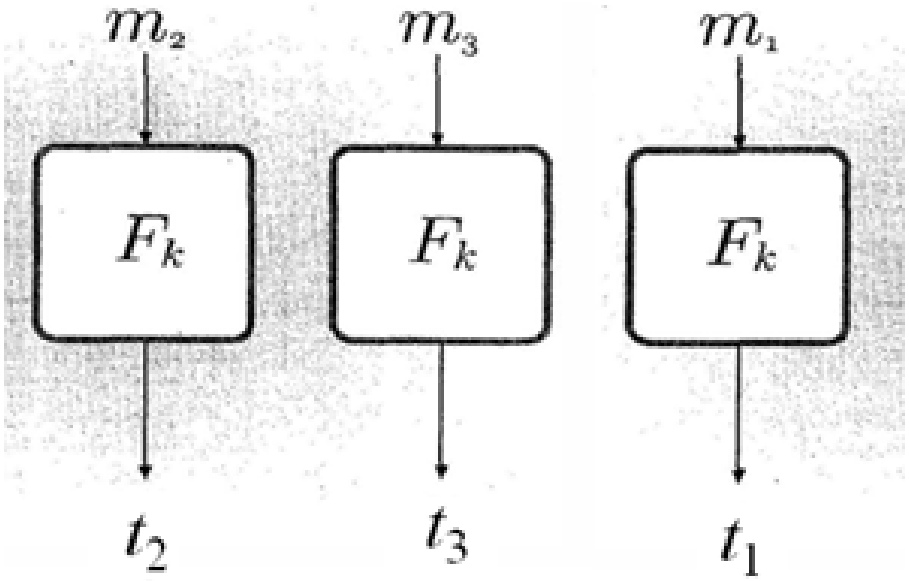
- Π = (Gen,Mac,Vrfy)是一个变长的消息认证码



- 是否安全？伪造如何找到？

由定长消息认证码构造变长消息认证码

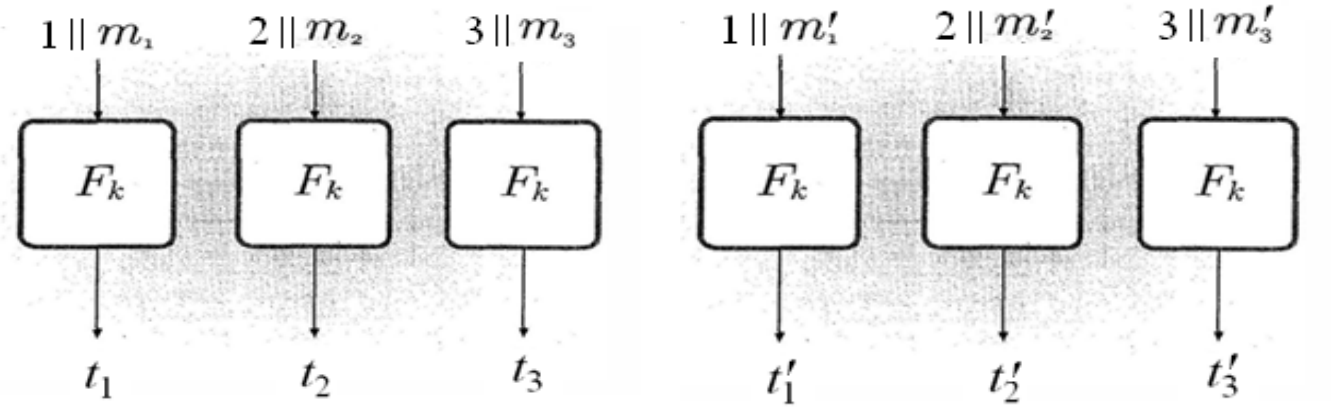
- Π = (Gen,Mac，Vrfy)是一个定长的认证码

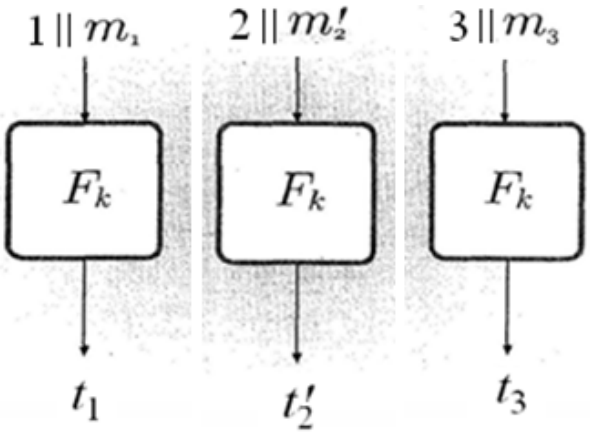


- 是否安全？伪造如何找到？

定长消息认证码

- Π = (Gen,Mac，Vrfy)是一个定长的鉴别码

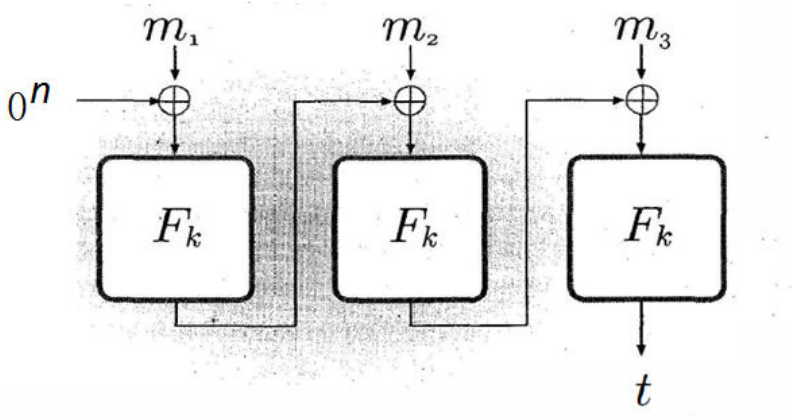




变长消息认证码的构造

- 构造方法
  - Π = (Gen,Mac，Vrfy)是一个定长的认证码
  - Gen：随机选择 $k \leftarrow \{0,1\}^n$
  - MAC：对任意长度小于$2^{n/4}$的消息 m，将 m 分成长度为 $n/4$ 的分组 $m_1, \ldots, m_d$，随机选择一个标识符 $r \leftarrow \{0,1\}^{n/4}$ 对 $i = 1, \ldots, d$，计算 $t_i \leftarrow \mathrm{MAC}_k(r\|l\|i\|m_i)$，$i$，$l$ 长度$n/4$ 输出标记 $t = < r, t_1, \ldots, t_d >$
  - Vrfy：输入 k 和 m 以及对应的 $t = < r, t_1, \ldots, t_d >$，将 m 分成长度为$n/4$ 的分组 $m_1, \ldots, m_d$，对对 $i = 1, \ldots, d$，$\mathrm{Vrfy}_k(r\|l\|i\|m_i) = 1$，输出1

CBC-MAC



CBC-MAC和CBC加密的比较：

CBC 使用随机的初始变量 IV，CBC-MAC 使用全 0 的初始值
CBC 加密每个分组都输出密文，CBC-MAC 只能最后一个分组输出标记

CBC-MAC算法

- 算法：消息

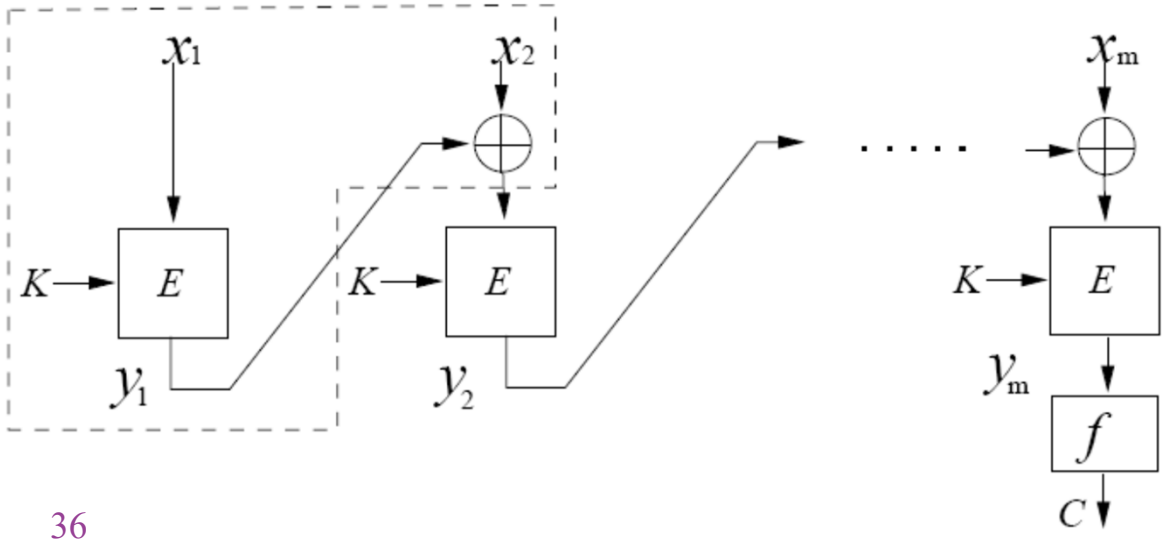
$$M = x_1 \| x_2 \| \cdots \| x_m, |x_i| = n$$

$$y_0 = 0$$

$$y_i = E_K(y_{i-1} \oplus x_i), 1 \le i \le m$$

$$CBC_K(M) = f(y_m)$$

- 结构图：



对CBC-MAC攻击

当 $f(x)=x$

已知 $C=CBC_K(M)$

$\rightarrow C=CBC_K(M \| C \oplus M)$

已知 $C=CBC_K(M), C'=CBC_K(M')$

$\rightarrow CBC_K(M' \| C' \oplus C \oplus N)=CBC_K(M \| N)$

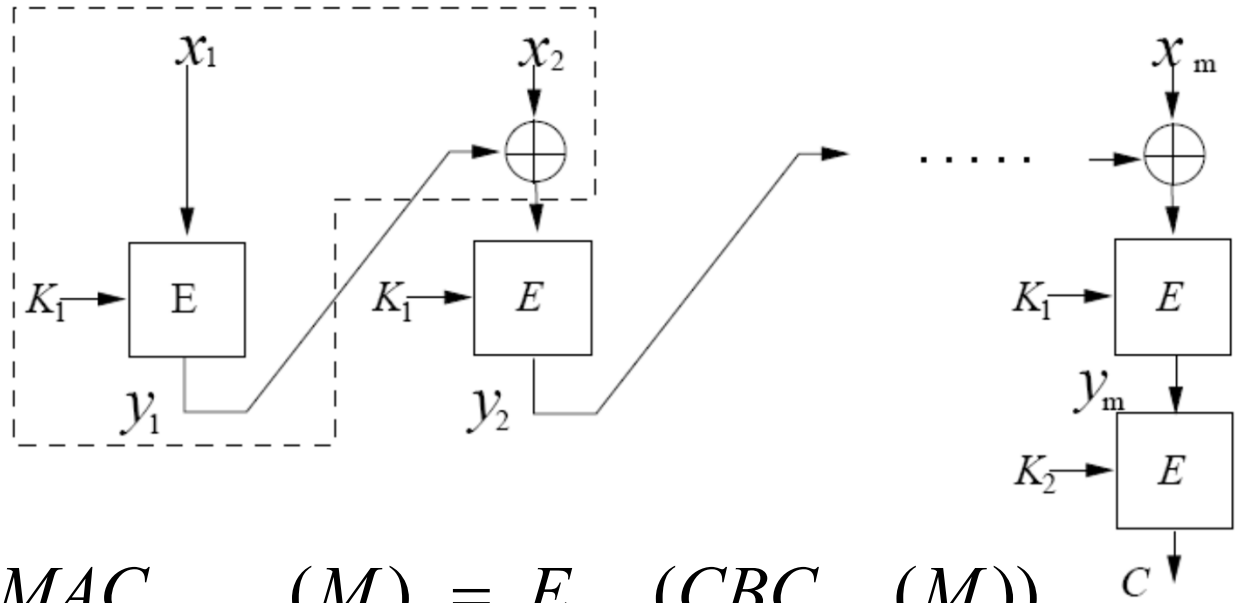
当 $f(x) \neq x$ 时，使用生日攻击得到一对碰撞

$$\left.\begin{array}{l} x=(M_1 \| M_2 \| M_3 \| \cdots \| M_q) \\ y=(M_1' \| M_2' \| M_3 \| \cdots \| M_q) \end{array}\right\} E_k(M_1 \oplus M_2)=E_k(M_1' \oplus M_2')$$

$$MAC(M_1 \| M_2 \oplus \delta \| M_3 \| \cdots \| M_q)=MAC(M_1' \| M_2' \oplus \delta \| M_3 \| \cdots \| M_q)$$

EMAC（Encrypted CBC MAC)

- B. den Boer 等 “RIPE-RACE 1040”上提出
- 2个密钥，$m+1$次加密

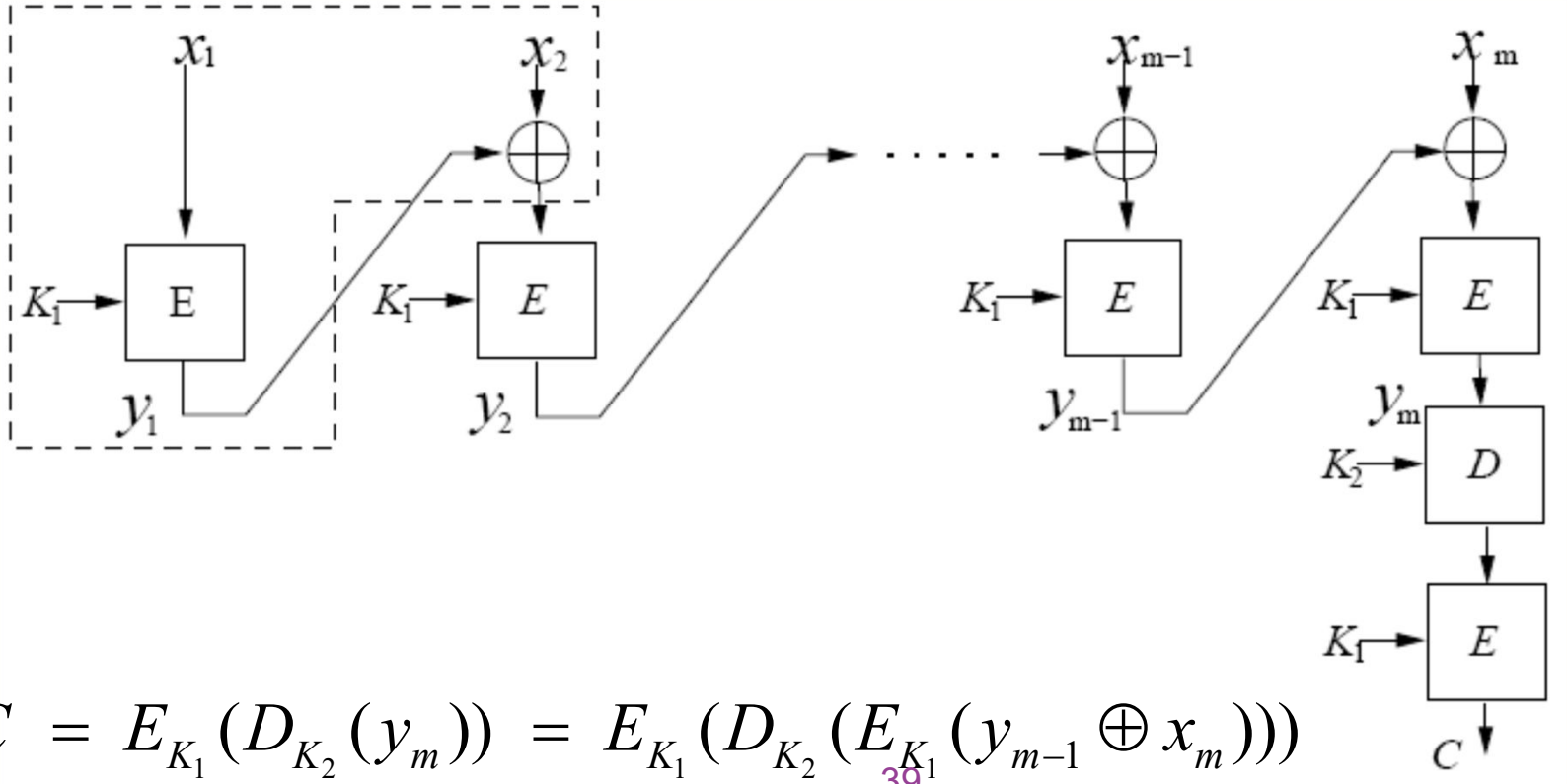


$$EMAC_{K_1,K_2}(M) = E_{K_2}(CBC_{K_1}(M)),$$

- E. Petrank 和 C. Rackoff 给出了一个安全证明

Retail MAC

- ANSI X9.19, ISO/IEC 9797
- 2个密钥，m+2次加密



$$C = E_{K_1}(D_{K_2}(y_m)) = E_{K_1}(D_{K_2}(E_{K_1}(y_{m-1} \oplus x_m)))$$

密钥恢复攻击（基于DES算法的Retail MAC）

- 96年，B. Preneel 和 P. C. van Oorschot给出了一种密钥恢复攻击：(K_1, K_2) 共112比特密钥
  - $2^{32.5}$ 已知的 text-MAC 对
  - $3 \cdot 2^{56}$ 离线DES计算
- 攻击方法：
  - 利用已知的text-MAC 对找一个碰撞
  - 穷搜密钥K_1，利用碰撞对识别，得到K_1
  - 穷搜密钥K_2，$D_{K_1}(C) = D_{K_2}(y_m)$

对消息认证码的攻击

- 重放攻击（replay attack)： 攻击者没有破解消息认证码，而是将事先保存的MAC值进行重放来发动攻击
- 防止重放攻击的办法：
  - 序号：每次对发送的消息赋予一个递增的编号
  - 时间戳：在发送的消息中包含当前的时间
  - Nonce: 在通信之前，接收者先向发送者发送一个一次性的随机数

教学内容

1. 认证的功能
2. 消息认证码的构造
3. 认证加密算法

获得私密性和消息鉴别（认证加密）

- 加密和认证(Encrypt and authenticate)：加密和鉴别独立进行。给定消息 m，计算 $c \leftarrow \mathrm{Enc}_{k1}(m)$ 和 $t \leftarrow \mathrm{Mac}_{k2}(m)$，输出 $< c, t >$
- 鉴别后加密(Authenticate then Encrypt)：给定消息 m，首先计算$t \leftarrow \mathrm{Mac}_{k2}(m)$，再计算 $c \leftarrow \mathrm{Enc}_{k1}(m \| t)$，只将 c 输出
- 加密后鉴别(Encrypt then Authenticate)：给定消息 m，首先计算$c \leftarrow \mathrm{Enc}_{k1}(m)$, 再计算 $t \leftarrow \mathrm{Mac}_{k2}(c)$, 输出$< c, t >$
- 用户需求
  - 仅仅是加密？还是既需要加密又需要鉴别？

认证加密（AE）

- Encrypt and authenticate
  - MAC_k不需要保证私密性，可能泄露消息信息
  - 确定性的CBC-MAC，敌手可以验证是否一个消息发送两次
- Authenticate then Encrypt: IPSEC,
  - Padding Oricle攻击 结合timing Attack
- Encrypt then Authenticate ?

CAESAR竞赛

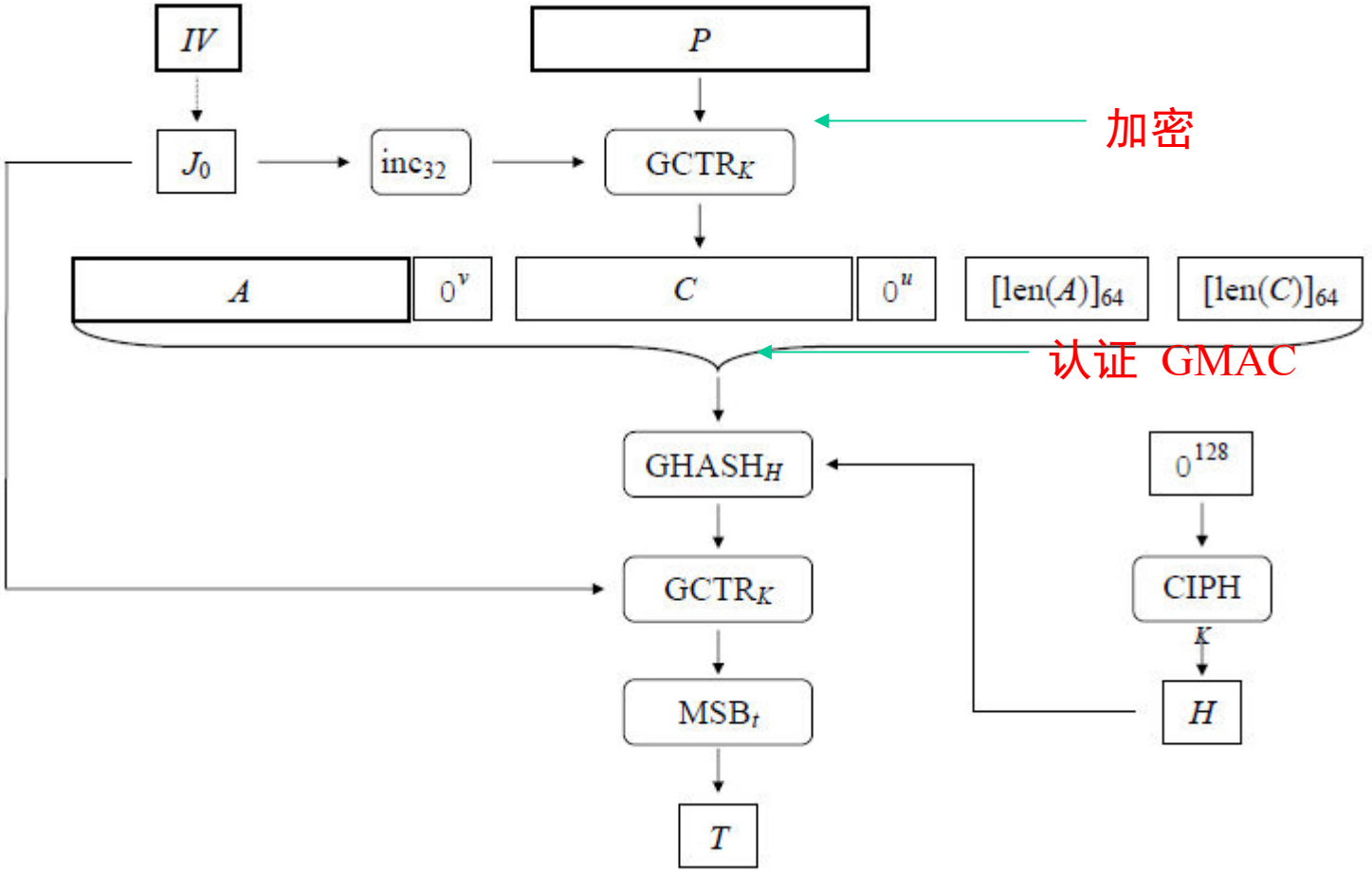
- CAESAR竞赛
  - 2013 年1 月开启，旨在选拔安全高效的认证加密算法
  - 2014 年3 月公布征集到的57个算法
    - 基于分组密码的认证加密算法，12个
    - 基于杂凑函数的认证加密算法，11个
    - 基于流密码的认证加密算法，6个
  - 2015 年7 月，公布了进入第二轮评估的29 个候选算法
  - 2016 年8 月，公布了进入第三轮评估的15 个候选算法
  - 2018年3月，公布Finalists的7个算法
  - 2019年2月，公布了最后的6个Winners

Finalists

candidate	designers
ACORN for use case 1: v1 v2 **v3**	Hongjun Wu
AEGIS for use case 2: v1 **v1.1**	Hongjun Wu, Bart Preneel
Ascon for use case 1: home v1 v1.1 **v1.2**	Christoph Dobraunig, Maria Eichlseder, Florian Mendel, Martin Schläffer
COLM for use case 3: v1pre **v1**; superseding AES-COPA: v1 v2; and superseding ELmD: v1clarification v2.0 v2.1	Elena Andreeva, Andrey Bogdanov, Nilanjan Datta, Atul Luykx, Bart Mennink, Mridul Nandi, Elmar Tischhauser, Kan Yasuda
Deoxys-II for use case 3 (Deoxys-I was only in third round): home v1 ordering addendum v1.3v1 .4 **v1.41**	Jérémy Jean, Ivica Nikolić, Thomas Peyrin, Yannick Seurin
MORUS for use case 2: v1 figure1-corrected v1.1 **v2**	Hongjun Wu, Tao Huang
OCB for use case 2: v1 **v1.1**	Ted Krovetz, Phillip Rogaway

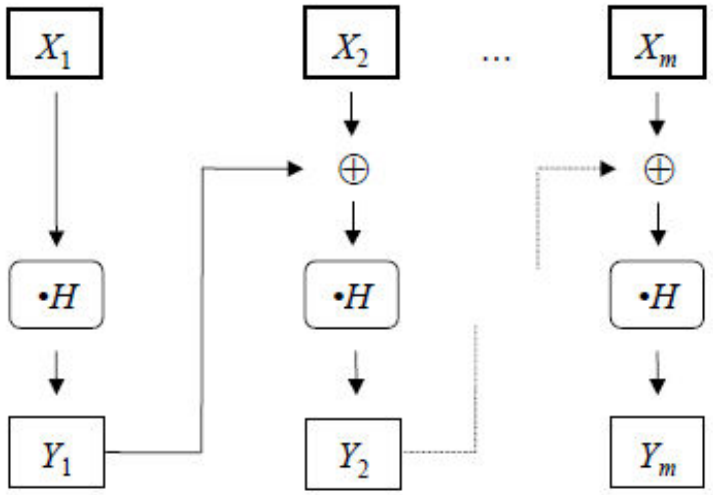
GCM模式

NIST SP 800-38D



$\mathbf{GCM\_AE_K(IV, P, A)=(C,T)}$

GHASH函数



图：$GHASH_H(X_1||X_2||\ldots||X_m) = Y_m$

R: $x^{128}+x^7+x^2+1$

算法：$X \bullet Y$

输入：blocks X, Y

输出：block $X \bullet Y$

Steps:

1. 设 $x_0x_1\ldots x_{127}$ 代表比特序列X.
2. 设$Z_0=0^{128}$， $V_0=Y$.
3. 对i=0 to 127, 计算Z_{i+1}和V_{i+1}如下：

$$Z_{i+1} = \begin{cases} Z_i & \text{if } x_i = 0; \\ Z_i \oplus V_i & \text{if } x_i = 1. \end{cases}$$

$$V_{i+1} = \begin{cases} V_i \gg 1 & \text{if } \mathrm{LSB}_1(V_i)=0; \\ (V_i \gg 1) \oplus R & \text{if } \mathrm{LSB}_1(V_i)=1. \end{cases}$$

GCTR函数

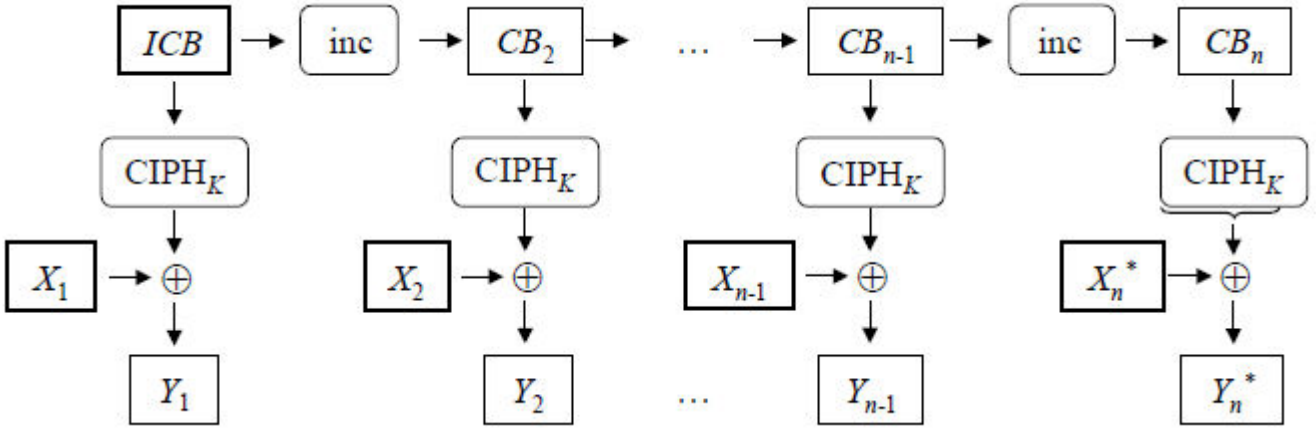
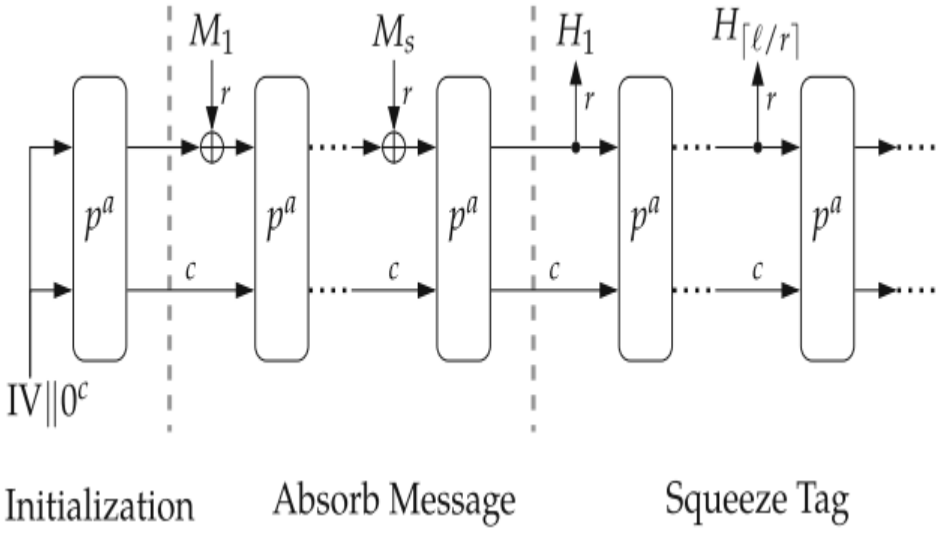


图:$GCTR_K(ICB, X_1 \| X_2 \| \cdots \| X_n^*) = Y_1 \| Y_2 \| \cdots \| Y_n^*$

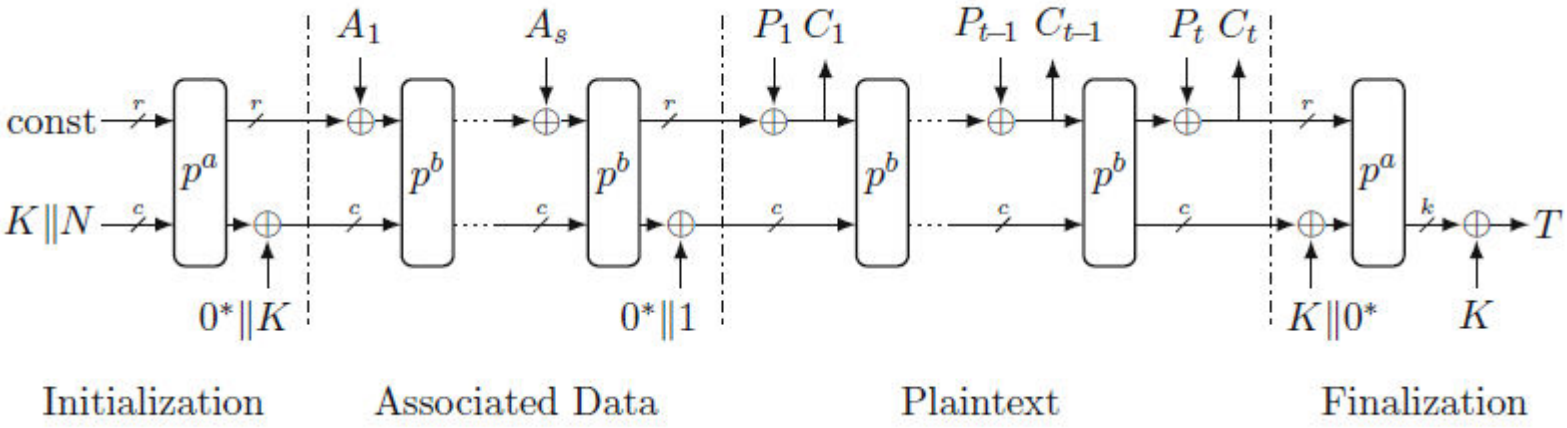
ASCON密码标准

- ASCON 密码套件认证加密算法和哈希算法
- 哈希算法包括定长输出的ASCON-HASH和可变长输出的ASCON-XOF，采用海绵结构



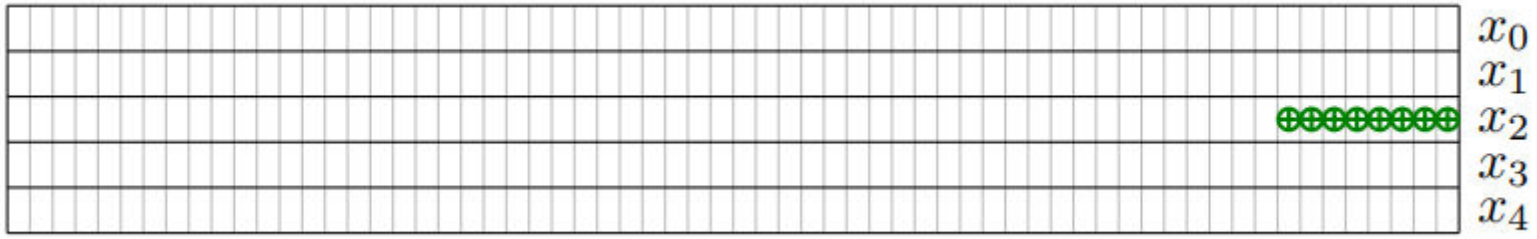
ASCON认证加密算法

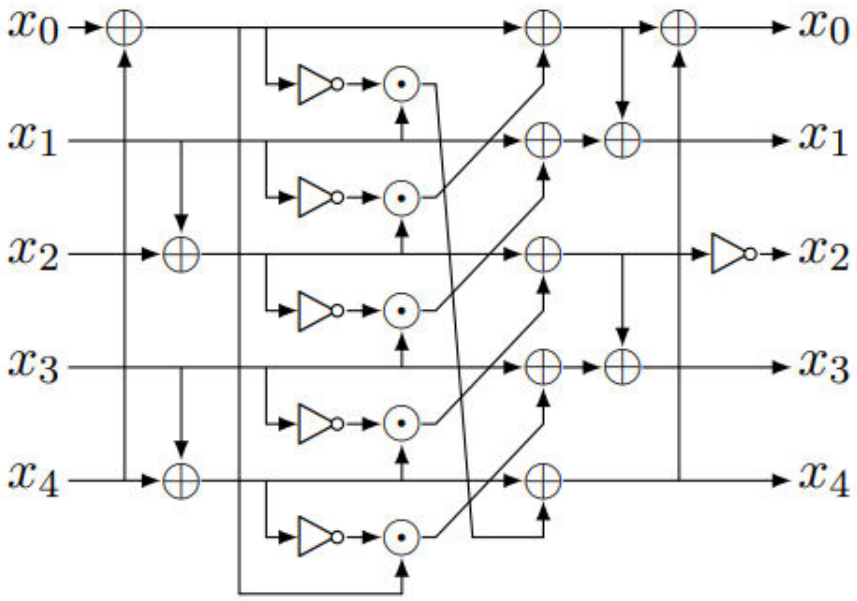
name	bit size of				rounds	
	key	nonce	tag	data block	p^a	p^b
ASCON-128	128	128	128	64	12	6
ASCON-96	96	96	96	128	12	8



ASCON轮函数

- 轮函数 $p = p_L \circ p_S \circ p_C.$





ASCON轮函数

线性层

$$\Sigma_0(x_0) = x_0 \oplus (x_0 \ggg 19) \oplus (x_0 \ggg 28)$$
$$\Sigma_1(x_1) = x_1 \oplus (x_1 \ggg 61) \oplus (x_1 \ggg 39)$$
$$\Sigma_2(x_2) = x_2 \oplus (x_2 \ggg 1) \oplus (x_2 \ggg 6)$$
$$\Sigma_3(x_3) = x_3 \oplus (x_3 \ggg 10) \oplus (x_3 \ggg 17)$$
$$\Sigma_4(x_4) = x_4 \oplus (x_4 \ggg 7) \oplus (x_4 \ggg 41)$$

谢谢！