امنیت داده و شبکه

کدهای احراز صحت پیام و توابع چکیده‌ساز

فهرست مطالب

- مفاهیم اولیه
- رمزگذاری پیام و کدهای تشخیص خطا
- کدهای احراز صحت پیام
- اصول توابع چکیده‌ساز
- توابع چکیده‌ساز مهم
- HMAC

احراز صحت پیام چیست؟

- اطمینان از:

۱- صحت محتوای پیام؛ یعنی پیام دریافتی دستکاری نشده است:

  - بدون تغییر
  - بدون درج
  - بدون حذف

۲- پیام از جانب فرستنده ادعا شده ارسال شده است.

احراز صحت پیام

- در بسیاری از کاربردها، مثلاً تراکنش‌های بانکی، حفظ محرمانگی محتوای ارتباطات اهمیت زیادی ندارد، ولی اینکه محتوای آنها قابل اعتماد باشند از اهمیت بسیار بالاتری برخوردار است.

- نیاز به دو عنصر کارکردی داریم:
  - عنصر اول: یک تابع برای تولید عامل احرازکننده
  - عنصر دوم: یک پروتکل که با استفاده از تابع فوق اصالت پیام را احراز کند.

راهکارهاي متصور برای احراز صحت پیام

- رمزگذاری پیام و کدهای تشخیص خطا
  - رمزشده کل پیام به عنوان احراز کننده اصالت پیام
  - رمز شده کدتشخیص خطا به عنوان احراز کننده اصالت پیام
- کد احراز صحت پیام (MAC)
  - تابعی از متن پیام و یک کلید سری (با خروجی با اندازه ثابت) به عنوان احرازکننده پیام
- استفاده از توابع چکیده‌ساز برای احراز صحت پیام
  - خروجی حاصل از نگاشت پیام به یک مقدار با طول ثابت (با استفاده از یک تابع چکیده‌ساز) به عنوان احرازکننده پیام

فهرست مطالب

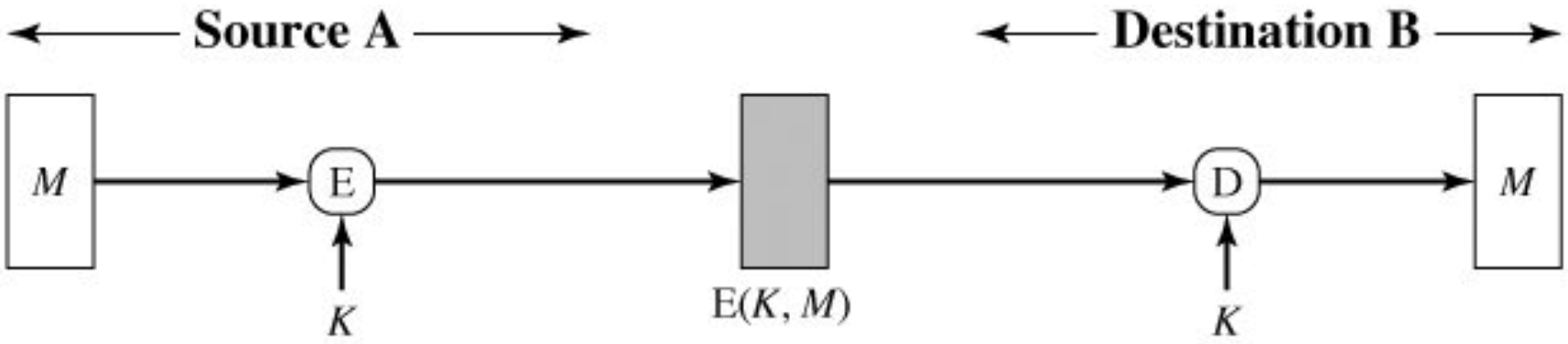
- مفاهیم اولیه
- **رمزگذاری پیام و کدهای تشخیص خطا**
- کدهای احراز صحت پیام
- اصول توابع چکیده‌ساز
- توابع چکیده‌ساز مهم
- HMAC

رمزگذاری پیام براي احراز صحت پیام

- فرستنده پیام را رمز می‌کند.
- اگر متن رمز شده دستکاری شود با رمزگشایی به متن آشکار نامفهوم (درهم و برهم) می‌رسیم.
- گیرنده، بعد از رمزگشایی چک می‌کند که آیا پیام مفهوم است یا نه؟ [به صورت الگوریتمی نمی‌توان چک کرد]
- می‌توان از الگوریتم‌های رمز متقارن و یا نامتقارن برای این منظور استفاده کرد.

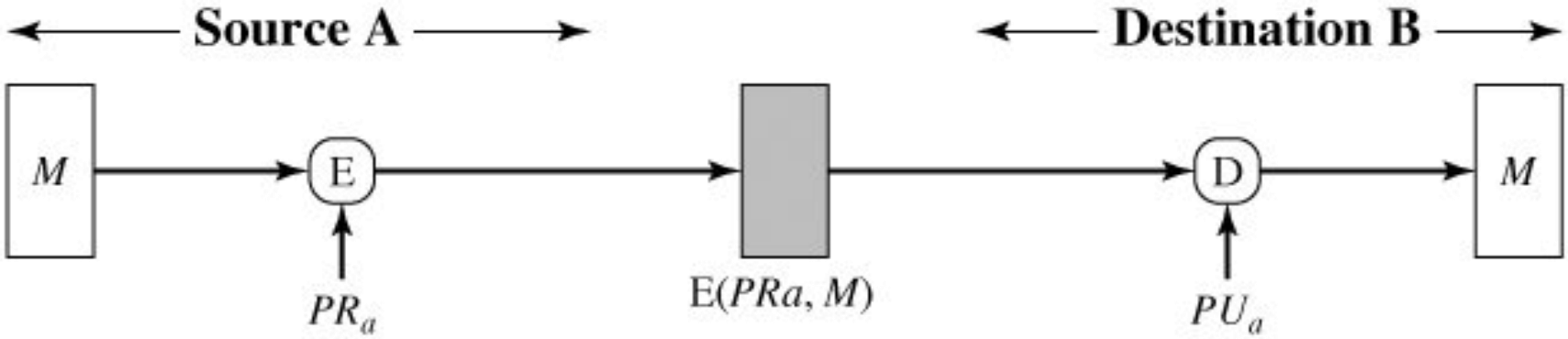
کاربرد رمزگذاری پیام

رمزنگاری متقارن: محرمانگی و احراز صحت



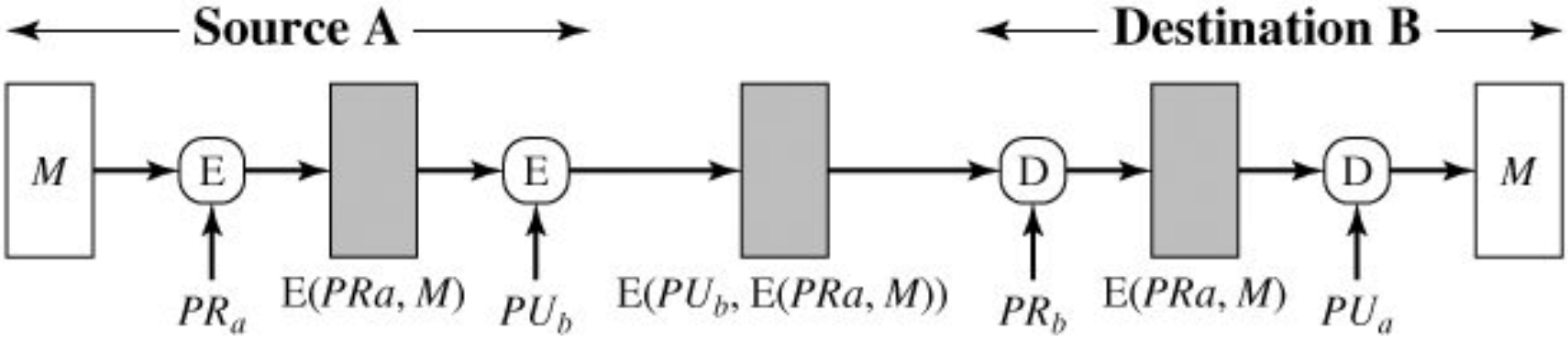
کاربرد رمزگذاری پیام

رمزنگاری کلید عمومی: احراز صحت و امضاء



کاربرد رمزگذاری پیام

رمزنگاری کلید عمومی: محرمانگی، احراز صحت و امضاء



مشکلات رمزنگاری

- بررسی مفهوم بودن محتوا همواره آسان نیست.
  - در حالت کلی با نوعی افزونگی، ساختار درونی مورد انتظار را بررسی می‌کنند.
  - دشواری خودکارسازی فرآیند چک کردن ....
- هنگام ارسال داده
  - اگر داده‌ها خود تصادفی به نظر برسند، یعنی از ساختار درونی خاصی تبعیت ننمایند، بررسی محتوا تقریباً ناممکن است. (مانند فایل اجرایی)
- **راه‌حل اولیه:** استفاده از کدهای تشخیص خطا
  - مثال: یک بیت به عنوان parity انتهای پیام اضافه نماییم، به گونه‌ای که تعداد بیتهای یک، زوج شود.

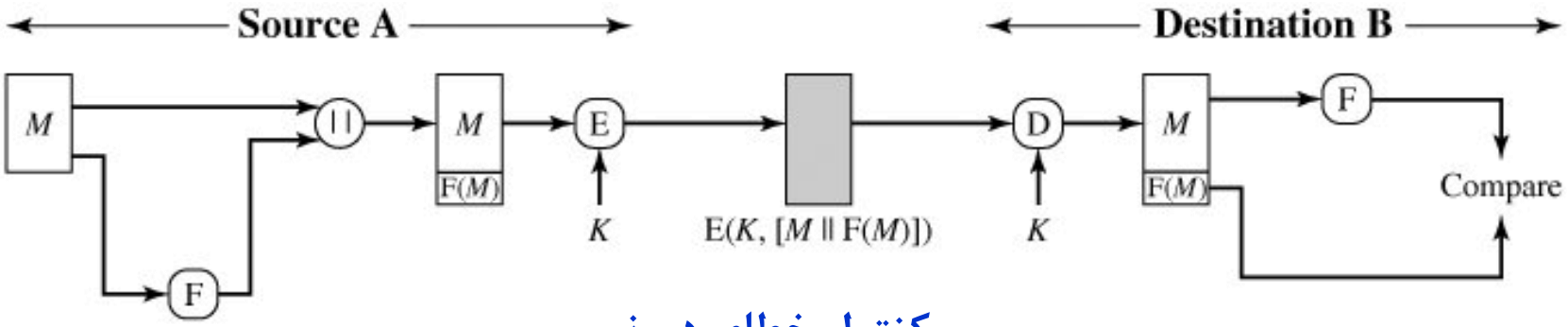
کدهاي تشخيص خطا

- تابع F یک کد تشخیص خطا است.
  - اضافه نمودن کدتشخیص خطا (به دنباله بررسی قالب یا FCS- Frame Check Seq نیز معروف است)، توسط تابع F
  - یک مثال از تابع F ، کد CRC است.

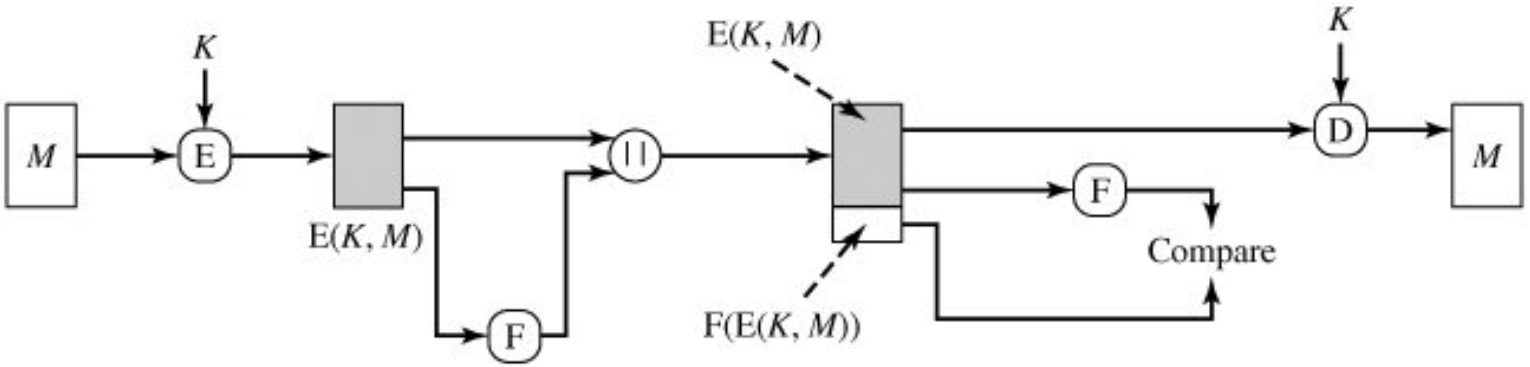
- گیرنده، بعد از رمز گشایی چک می‌کند که آیا «کد تشخیص خطای» محاسبه شده توسط F با برچسب پیام مطابقت دارد یا نه.

انواع کدهای تشخیص خطا

- دو مدل اضافه کردن کد تشخیص خطا (دومی کاملا ناامن است)



کنترل خطای درونی



کنترل خطای بیرونی

ناامن بودن کدهاي تشخيص خطا

- کدهای تشخیص خطا مانند CRC برای تشخیص خطای حاصل از نویز در کاربردهای مخابراتی طراحی شده‌اند.
  - **نویز:**
    - تغییرات غیرهوشمندانه و غیرعمدی
  - حمله دشمن:
    - تغییرات هوشمندانه و عمدی
- حملات موفقی به الگوریتم‌هایی که از کدهای تشخیص خطا استفاده می‌کردند، صورت پذیرفته است.

نتیجه گیری

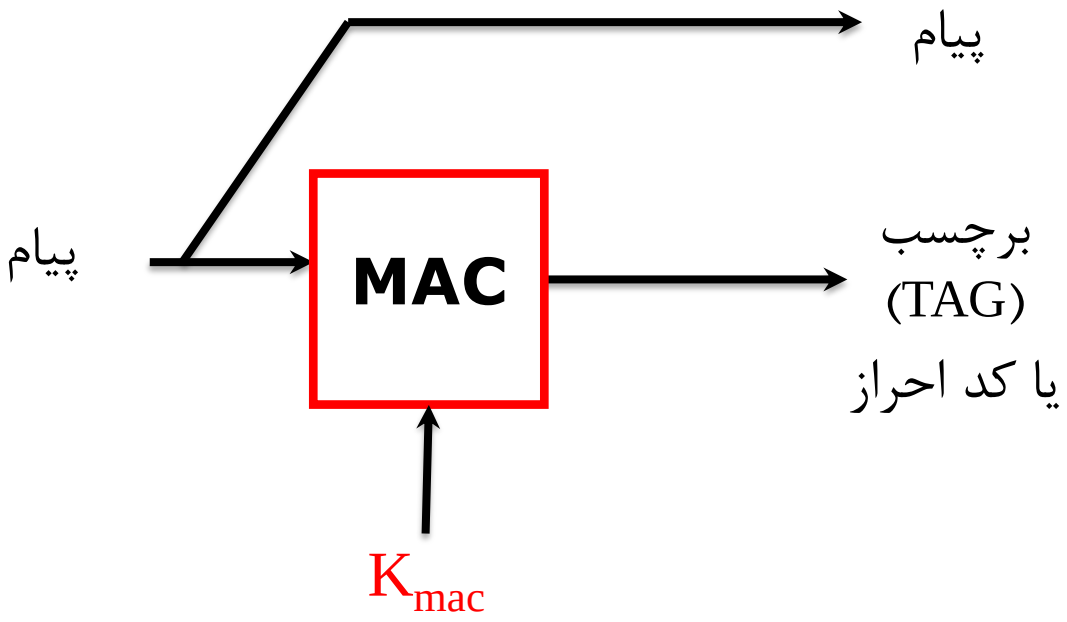
- کد تشخیص خطا نمی‌تواند در حالت کلی از دستکاری بسته‌ها جلوگیری کند.
- راه حل: کدهای احراز صحت پیام

فهرست مطالب

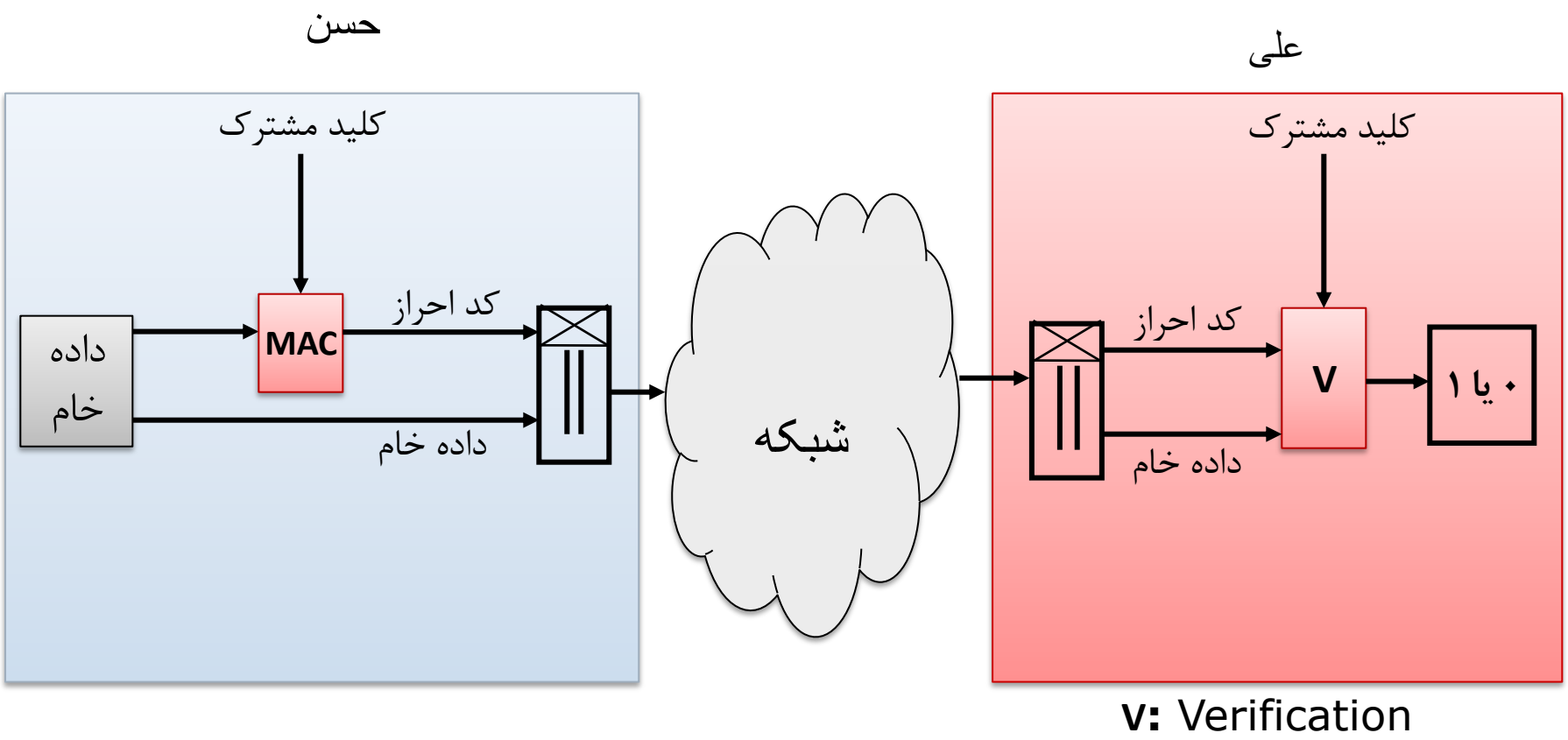
کد هاي احراز صحت پيام

- تولید یک برچسب با طول ثابت:
  - وابسته به پیام
  - لزوماً برگشت پذیر نیست (بر خلاف توابع رمزنگاری)
  - نیازمند اشتراک یک کلید مخفی بین طرفین
  - آنرا به اختصار MAC مینامند. نام دیگر "Cryptographic Checksum"
- این برچسب را به پیام اضافه می‌کنند.
- گیرنده برچسب پیام را محاسبه نموده و با برچسب ارسالی مقایسه می‌کند.
- از صحت پیام و هویت فرستنده اطمینان حاصل می‌شود.

کد هاي احراز صحت پيام



نحوه عملکرد کدهای احراز صحت پیام

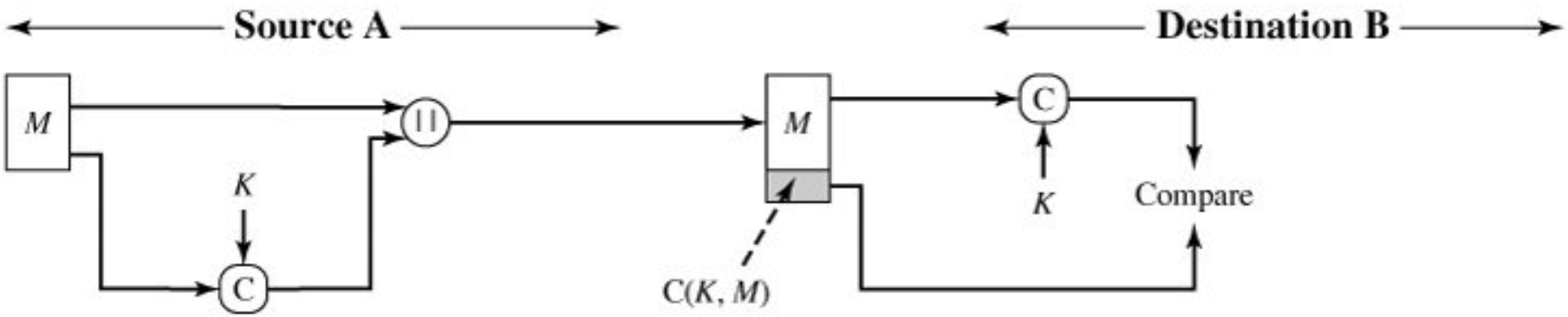


فرق MAC و رمزگذاری

- MAC نیازی ندارد که حتماً برگشت پذیر باشد، در صورتی که الگوریتم رمزگذاری باید برگشت‌پذیر باشد.
  - MAC تابع چند به یک است.
  - اندازه خروجی MAC برابر n بیت، تعداد MACهای ممکن = 2^n
  - اندازه کلید MAC برابر k بیت، تعداد نگاشتهای ممکن به MACها= 2^k
- با توجه به خصوصیات ریاضی MAC، آسیب‌پذیری‌های احتمالی برای شکست آن کمتر است.

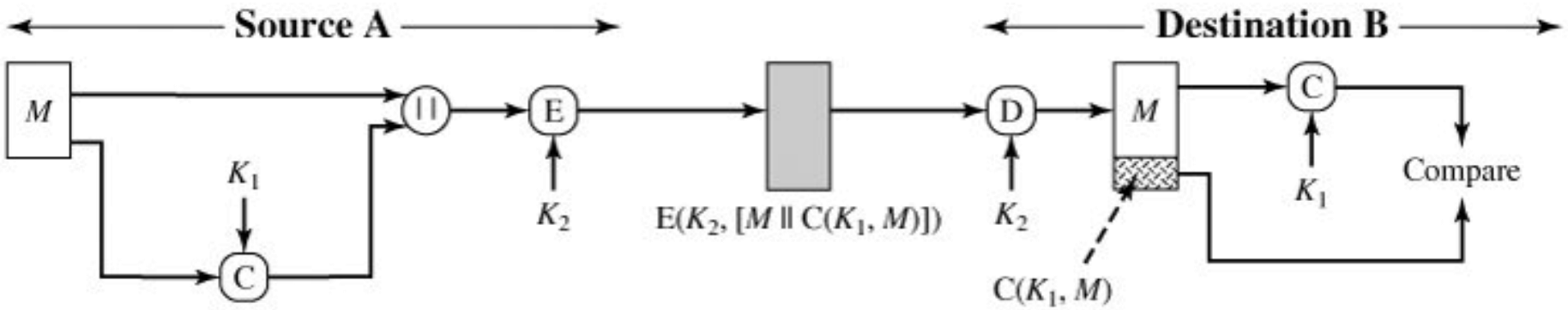
کاربرد کدهای احراز صحت پیام

احراز صحت پیام



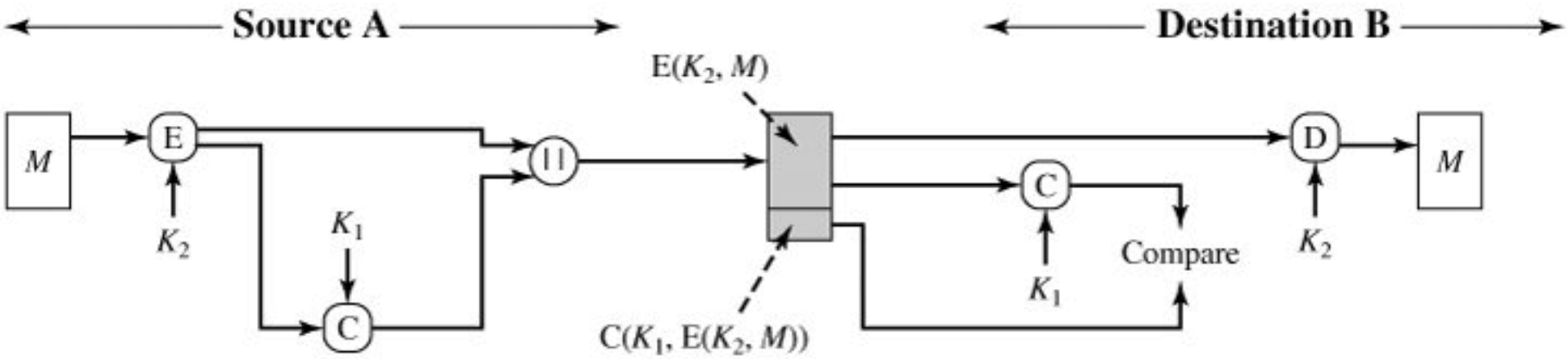
کاربرد کدهای احراز صحت پیام

احراز صحت پیام و محرمانگی؛ احراز صحت پیام آشکار



کاربرد کدهای احراز صحت پیام

احراز صحت پیام و محرمانگی؛ احراز صحت پیام رمز



سوالات متداول در مورد MAC

- چرا از MAC به جای رمزنگاری استفاده می‌کنیم؟
  - در بعضی کاربردها نیازی به محرمانگی نداریم...
  - در بعضی موقعیت‌ها، قوانین اجازه ارتباط رمزشده را نمی‌دهند...
  - اگر از رمزنگاری استفاده نماییم، برای خواندن پیام همیشه به واگشایی رمز نیاز داریم در صورتی که بررسی MAC اختیاری است...
  - الگوریتم‌های تولید چکیده پیام عموما از الگوریتم‌های رمزنگاری سریعتر هستند.

سوالات متداول در مورد MAC

- آیا MAC همانند امضا غیر قابل انکار است؟
  - خیر
    - امضاء با یک زوج کلید عمومی/خصوصی فراهم می‌شود ولی کلید MAC یک کلید مشترک سری است.
    - بر خلاف امضاء، دو طرف قادر به ایجاد MAC هستند.

امنیت MAC

- حمله آزمون جامع به کلید MAC
  - با داشتن یک متن و MAC آن، به صورت برون خط انجام می‌پذیرد.
  - اگر طول کلید k بیت باشد، 2^k کلید ممکن باید بررسی شود.
  - با یافتن یک کلید، باید آن را با زوجهای دیگری چک کرد، چون ممکن است چند کلید مختلف، یک متن را به چکیده یکسان نگاشت کنند.
- حمله آزمون جامع برای کشف تصادم
  - با داشتن یک MAC، به دنبال پیامی می‌گردیم که همان MAC را حاصل نماید.
  - اگر MAC، n بیتی باشد، به طور متوسط با 2^n پیام، به احتمال زیاد یک تصادم رخ می‌دهد.

امنیت MAC

- هزینه لازم برای حمله آزمون جامع به MAC برابر است با

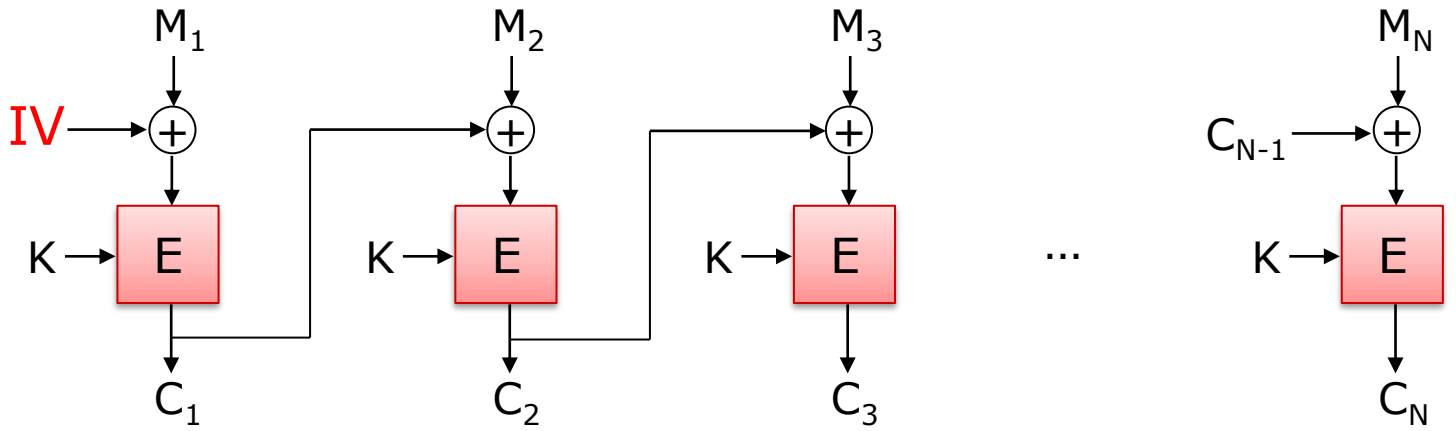
$$\min(2^k, 2^n)$$

- ویژگیهای یک MAC مناسب:
  - با دانستن یک پیام و بر چسب آن، یافتن پیام متفاوتی با برچسب یکسان از لحاظ محاسباتی ناممکن باشد.
  - توزیع خروجی MAC باید یکنواخت باشد تا احتمال اینکه دو پیام تصادفی MAC یکسان داشته باشند، کمینه شود.
- نکته: طول برچسب MAC همانند طول کلید در امنیت MAC تاثیر دارد.

ساختن MAC امن با استفاده از توابع رمزگذاری

- با استفاده از توابع رمزگذاری امن و برخی از سبک‌های رمزنگاری می‌توان توابع MAC امن ساخت.
  - **مثال:** سبک‌های CBC و CFB
- **مثال:** استاندارد DAA (Data Authentication Algorithm)
  - استاندارد NIST و ANSI X9.17
  - بر اساس رمز قطعه‌ای DES و مد کاری CBC
- در ساختن MAC از این سبک‌ها باید دقت زیادی کرد.
  - جزئیات بسیار مهم‌اند.
  - در ادامه تلاش می‌کنیم تا CBC-MAC بسازیم!

تلاش 1 – CBC-MAC



- **پیام:** $M = (M_1, \dots, M_N)$
- **برچسب:** $T = (IV, C_N)$
- پیام به همراه برچسب فرستاده می‌شود.
- برای احراز صحت، برچسب از نو محاسبه و با برچسب دریافتی مقایسه می‌شود.

حمله به تلاش 1

- مهاجم می‌تواند با انتخاب IV به دلخواه، قطعه اول پیام را تغییر دهد.
- با داشتن پیام $M = (M_1, M_2, \dots, M_N)$ و برچسب $T = (IV, C_N)$ می‌توان پیام و برچسب جدیدی را بدون داشتن کلید جعل کرد:

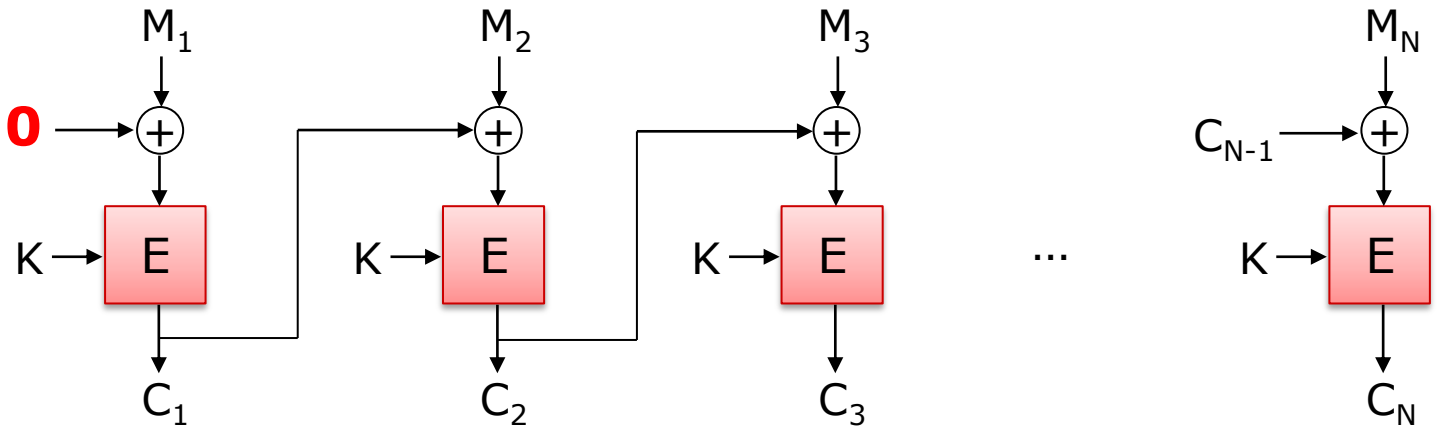
$$M' = (M_1', M_2, \dots, M_N)$$

$$T' = (IV', C_N)$$

$$IV' \oplus M_1' = IV \oplus M_1$$

تلاش 2 – CBC-MAC

- **راهکار:** استفاده از CBC-MAC با یک IV ثابت؛ مثلاً بردار تمام صفر.
- برچسب مساوی C_N است.



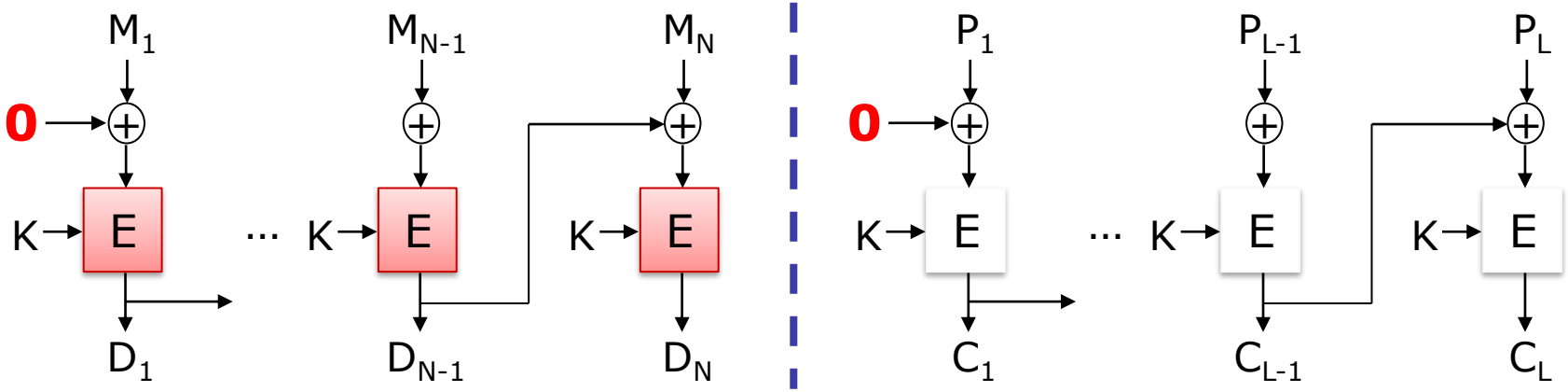
حمله به تلاش 2 – افزایش طول (Length Extension)

- با داشتن پیام تک‌قالبی $M = (M_1)$ و برچسب $T = C_1$ می‌توان پیام و برچسب جدیدی را بدون داشتن کلید جعل کرد:

$$M' = (M_1 \ , M_2)$$
$$T' = T = C_1$$
$$M_2 = M_1 \oplus C_1$$

- به همین ترتیب می‌توان جعل را ادامه داد و به پیام‌هایی با طول بیشتر رسید.

حمله به تلاش 2 – برچسب جدید از دو برچسب موجود



- دو پیام و برچسب روی هریک را داریم:
  - پیام $P = (P_1, \dots, P_L)$ با برچسب C_L.
  - پیام $M = (M_1, \dots, M_N)$ با برچسب D_N.

$$M' = (M_1, \dots, M_N, D_N \oplus P_1, P_2, \dots, P_L)$$

$$T' = T = C_L$$

راهکارها

- **راهکار ۱:** همه پیامهای سیستم، طول N دارند.
  - جلوگیری از حمله افزایش طول
  - مناسب برای بسیاری از پروتکلها
- **راهکار ۲:** همیشه طول پیام را به عنوان قطعه اول به تابع CBC-MAC می‌دهیم.
- **راهکار ۳:** قطعه آخر (C_N) را یک مرتبه مجدداً رمز می‌کنیم.
- **اثبات شده است که همه راهکارهای فوق امن هستند.**

معایب تولید MAC با رمزنگاری

- در بعضی مواقع، قوانین اجازه ارتباط با به کارگیری توابع رمزنگاری را نمی‌دهند.

- الگوریتم‌های بسیار سریع‌تری برای تولید MAC وجود دارد.
  - به کارگیری توابع چکیده‌ساز

فهرست مطالب

- مفاهیم اولیه
- رمزگذاری پیام و کدهای تشخیص خطا
- کدهای احراز صحت پیام
- **اصول توابع چکیده‌ساز**
- توابع چکیده‌ساز مهم
- HMAC

توابع چکیده‌ساز

- تابع یک‌طرفه
- طول ورودی متغیر
- طول خروجی ثابت (نگاشت از فضای بزرگتر به فضای کوچکتر)
- در حالت کلی، کلیدی در کار نیست!
  - بر خلاف MAC و رمزنگاری

امنیت توابع چکیده‌ساز-ایده کلی

- نگاشت پیام‌های طولانی به رشته‌های کوتاه به گونه‌ای که:
  - یافتن پیام‌های متفاوتی که به یک رشته یکسان نگاشته شوند دشوار باشد.

- به این رشته، **عصاره یا چکیده پیام (Message Digest)** می‌گوییم.

نیازمندیهای توابع چکیده‌ساز

- توابع چکیده‌ساز باید یک طرفه (One-Way) باشند.
  - برای یک h داده شده، باید یافتن x به گونه‌ای که h = H(x) از لحاظ محاسباتی ناممکن باشد.
- مقاومت در برابر تصادم ضعیف (Weak Collision)
  - برای یک x داده شده، باید یافتن y به گونه‌ای که H(y) = H(x) باشد، از لحاظ محاسباتی ناممکن باشد.
- مقاومت در برابر تصادم قوی (Strong Collision)
  - یافتن x و y به گونه‌ای که H(y) = H(x) باشد، از لحاظ محاسباتی ناممکن باشد.

مقایسه تصادم قوی و ضعیف

- ممکن است ساختار تابع H طوری باشد که:
  - بتوان تعداد محدودی x و y یافت به گونه‌ای که مقادیر تابع، تصادم پیدا کنند (تصادم قوی).
  - ولی برای یک x داده شده همواره نتوان یک y پیدا کرد بطوریکه H(y) = H(x) (تصادم ضعیف).

⇦ ارضاشدن شرط عدم وجود تصادم قوی برای یک تابع دشوارتر از ارضاشدن شرط عدم وجود تصادم ضعیف است.

⇦ توابعی که در برابر تصادم قوی مقاومت کنند امنیت بالاتری دارند.

امنیت توابع چکیده‌ساز

- توابع چکیده‌ساز باید یک طرفه باشند.
  - پیچیدگی جستجوی کامل (آزمون جامع) برای یافتن یک پیش‌نگاره (یعنی یک مقدار x که H(x)=h باشد) 2^n است، که n طول خروجی تابع است.
- مقاومت در برابر تصادم (ضعیف)
  - پیچیدگی جستجوی کامل (آزمون جامع) 2^n است.

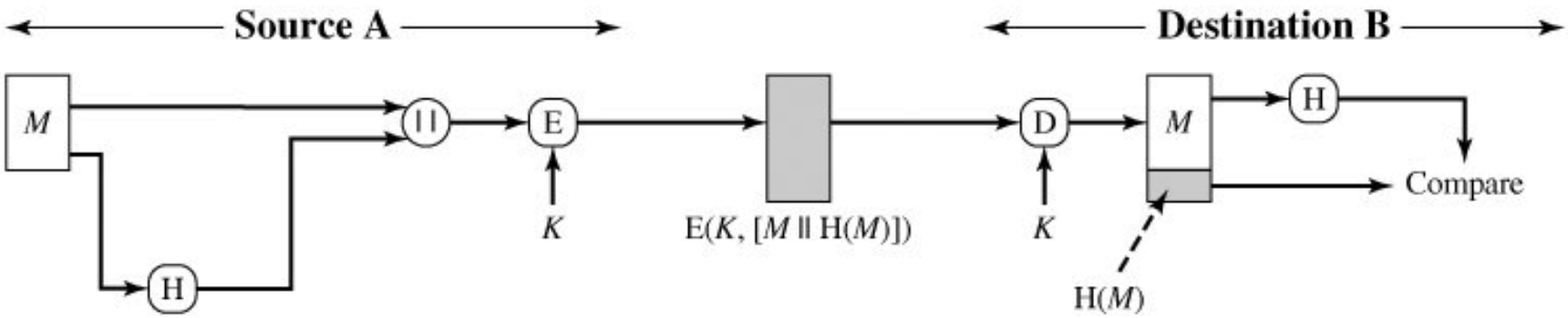
با کمک حمله روز تولد

- مقاومت در برابر تصادم (قوی)
  - پیچیدگی جستجوی کامل (آزمون جامع) $2^{n/2}$ است.

- دقت کنید آزمون جامع بیانگر حداکثر امنیت ممکن برای تابع است، زیرا ممکن است به دلیل ضعف طراحی، حملات موثرتری نیز امکان پذیر باشد.

کاربرد توابع چکیده‌ساز

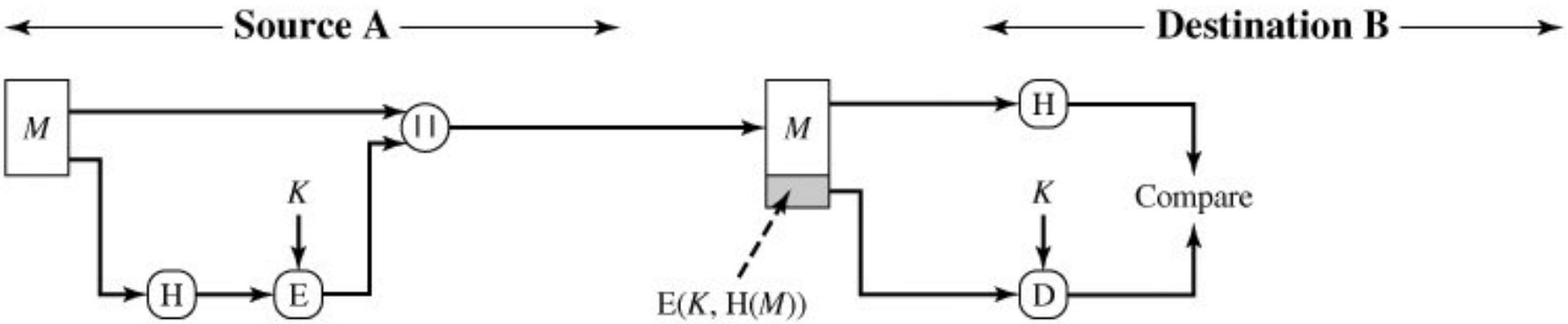
احراز صحت پیام و محرمانگی در ترکیب با رمز متقارن



کاربرد توابع چکیده‌ساز

احراز صحت پیام در ترکیب با رمز متقارن

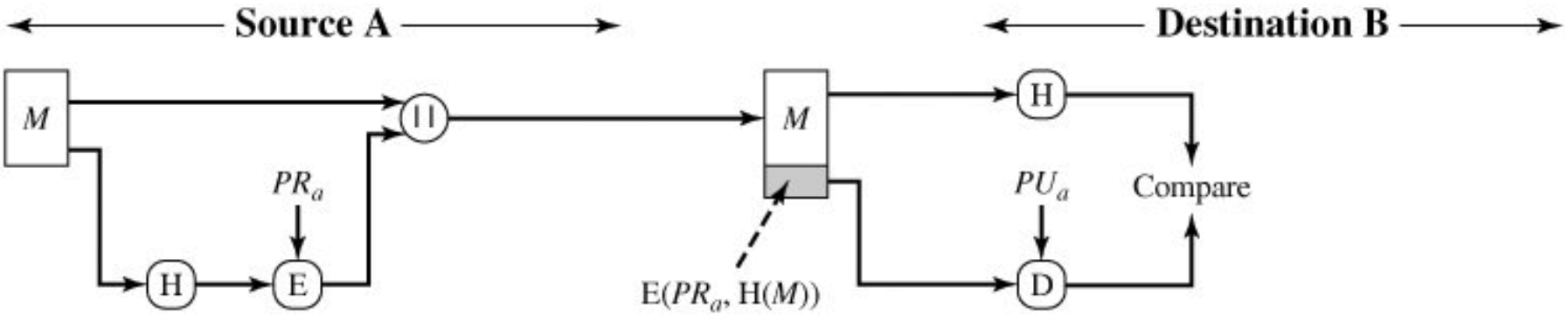
- صرفاً رمزگذاری چکیده پیام
- این ترکیب در واقع یک کد احراز صحت پیام را می‌سازد.



کاربرد توابع چکیده‌ساز

احراز صحت پیام در ترکیب با رمز کلید عمومی

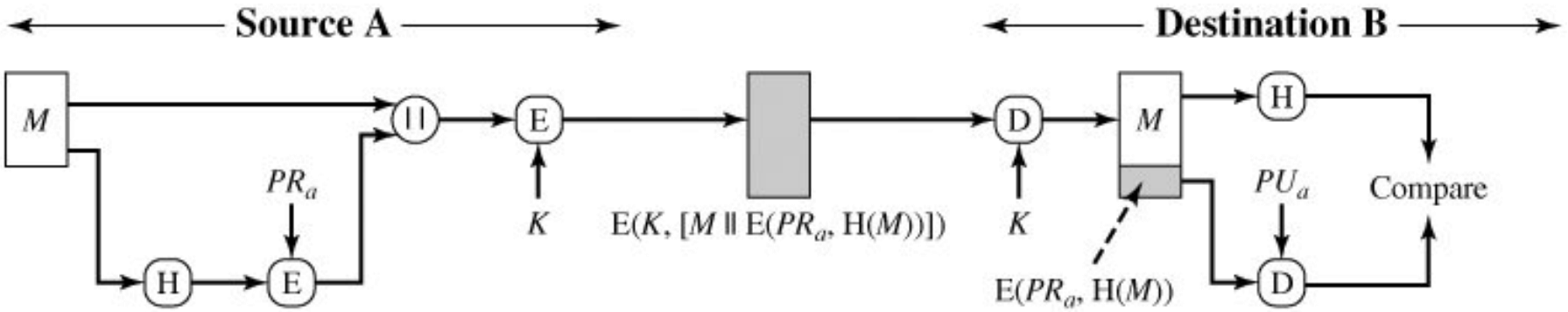
- صرفاً رمزگذاری چکیده پیام
- این ترکیب در واقع یک **امضای دیجیتال** را می‌سازد.



کاربرد توابع چکیده‌ساز

احراز صحت پیام و محرمانگی در ترکیب با رمز متقارن و نامتقارن

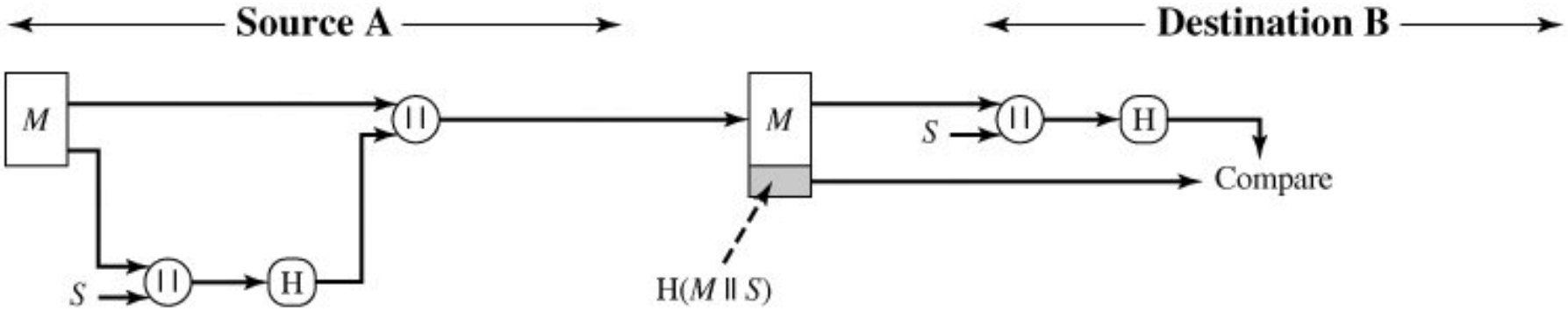
- امضای دیجیتال با رمز نامتقارن برای حفظ صحت
- رمز متقارن برای حفظ محرمانگی



کاربرد توابع چکیده‌ساز

احراز صحت پیام بدون رمزگذاری

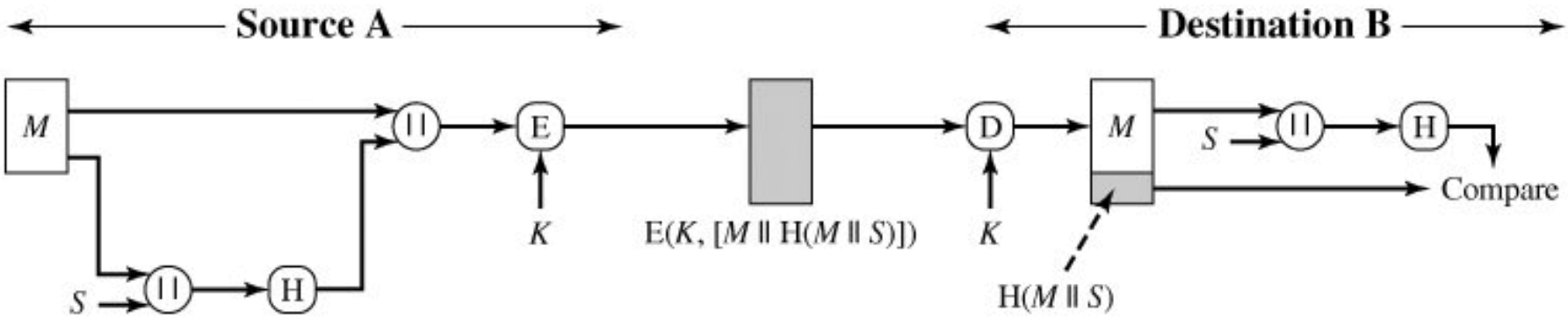
- طرفین راز S را مخفیانه به اشتراک می‌گذارند.
- بدون استفاده از رمزگذاری
- کاربرد عملی زیاد ولیکن آسیب‌پذیر در برابر حمله افزایش طول (توضیح در اسلایدهای بعدی)



کاربرد توابع چکیده‌ساز

احراز صحت پیام بدون رمزگذاری و محرمانگی با رمز متقارن

- رمزگذاری صرفاً برای محرمانگی



پارادوکس روز تولد

- در میان ۲۳ نفر، احتمال یافتن دو نفر که در یک روز از سال متولد شده اند بیش از ۵۰٪ است.
- این پارادوکس به دو شکل عمومی برای یک تابع دلخواه توسعه یافت.
  - اندازه حداقلِ **یک** مجموعه برای یافتن یک زوج در آن با خروجی یکسان با احتمال بیش از ۰/۵
  - اندازه حداقلِ **دو** مجموعه برای یافتن یک تصادم بین اعضای آنها با احتمال بیش از ۰/۵

پارادوکس روز تولد

مبنای ریاضی

- تابع H با 2^n خروجی ممکن را در نظر بگیرید (خروجی n بیتی).
- به H، k ورودی تصادفی اعمال کنیم و خروجی را مجموعه X در نظر می‌گیریم.
- به همین ترتیب مجموعه Y را تشکیل می‌دهیم.
- اگر k بزرگتر از $2^{n/2}$ باشد، احتمال حداقل یک تصادم در بین اعضای دو مجموعه X و Y بیش از ۰/۵ است.

حمله روز تولد

- ممکن است تصور کنید یک MAC یا Hash ۶۴ بیتی امن است اما
- با حمله روز تولد امنیت از بین می‌رود:
  - مهاجم $2^{n/2}$ پیام معتبر که اساساً هم معنا هستند تولید می‌کند. n طول خروجی Hash است.
  - مهاجم همین تعداد از گونه‌های هم معنا از پیام بدخواهانه دلخواه خود را تولید می‌کند.
  - دو دسته پیام مقایسه می‌شوند تا زوجی یافت شود که چکیده یکسان داشته باشند.
  - از کاربر می‌خواهیم تا پیام معتبر را امضا نماید، و سپس پیام بدخواهانه دلخواه مهاجم را جایگزین می‌کنیم.

مثالي از حمله (نمونه معتبر)

Dear Dean Smith,

This [letter | message] is to give my [honest | frank] opinion of Prof Tom Wilson, who is [a candidate | up] for tenure [now | this year]. I have [known | worked with] Prof Wilson for [about | almost] six years. He is an [outstanding | excellent] researcher of great [talent | ability] known [worldwide | internationally] for his [brilliant | creative] insights into [many | a wide variety of] [difficult | challenging] problems.

مثالي از حمله (پيام مهاجم)

Dear Dean Smith,

This [letter | message] is to give my [honest | frank] opinion of Prof Tom Wilson, who is [a candidate | up] for tenure [now | this year]. I have [known | worked with] Prof Wilson for [about | almost] six years. He is an [poor | weak] researcher not well known in his [field | area]. His research [hardly ever | rarely] shows [insight in | understanding of] the [key | major] problems of [the | our] day.

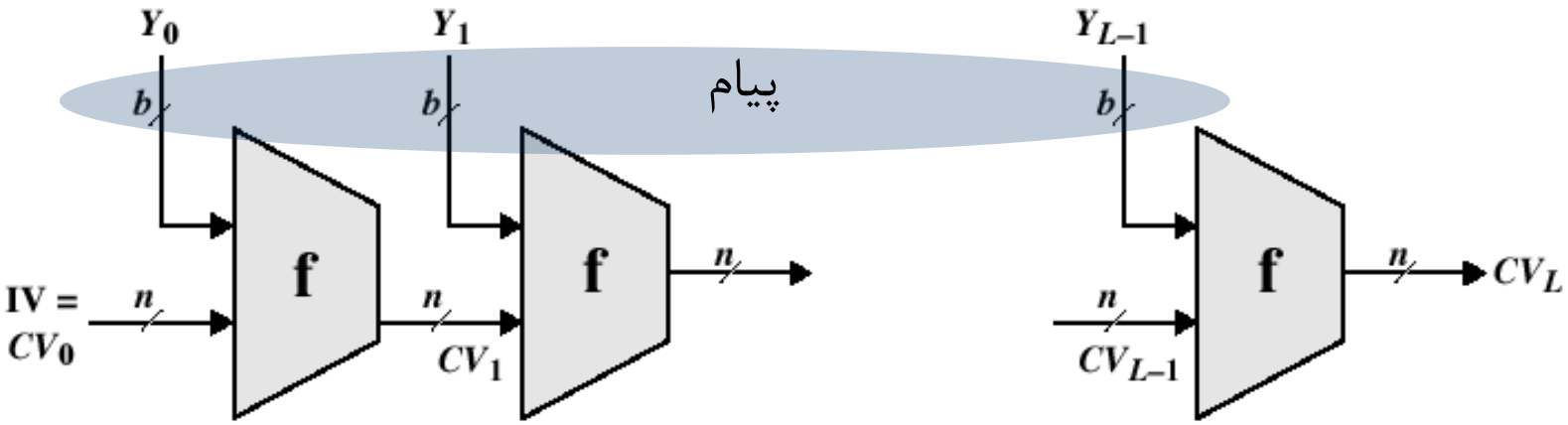
ساختار مرکل-دَمگارد برای توابع چکیده‌ساز

- مورد استفاده در بسیاری از توابع چکیده‌ساز
- اِعمال مکرر یک تابع فشرده‌ساز به یک رشته با طول ثابت
- اگر تابع فشرده‌ساز مقاوم در برابر تصادم باشد، تابع چکیده‌ساز نیز همین‌گونه خواهد بود.
- توابع معروفی مانند MD5 و SHA-1 از همین ایده استفاده می‌کنند.

Ralph Merkle
(1952 –)

Ivan Bjerre
Damgård (1956 –)

ساختار مرکل-دمگارد برای توابع چکیده‌ساز



IV = Initial value
CV = chaining variable
Y_i = ith input block
f = compression algorithm
L = number of input blocks
n = length of hash code
b = length of input block

- پیام به قطعات Y_i تقسیم شده است.
- IV یک رشته ثابت می‌باشد.

CV_0=IV

CV_i= f(CV_{i-1},Y_{i-1})

Hash = CV_L

فهرست مطالب

- مفاهیم اولیه
- رمزگذاری پیام و کدهای تشخیص خطا
- کدهای احراز صحت پیام
- اصول توابع چکیده‌ساز
- **توابع چکیده‌ساز مهم**
- HMAC

توابع چکیده‌ساز مهم: MD5

- MD5: Message Digest 5
  - طراحی 1992 توسط "ران ریوست"، یکی از سه طراح RSA
- استفاده گسترده در گذشته، اما از کاربرد آن کاسته شده است.
- ویژگیها:
  - پیام به قطعات ۵۱۲ بیتی تقسیم می‌شود.
  - خروجی ۱۲۸ بیتی

امنیت MD5

- مقاومت در برابر تصادم (قوی) تحت حمله آزمون جامع: 2^{64}
  - امروزه امن محسوب نمی‌شود.

- حملات کاراتری به این الگوریتم یافت شده‌اند:
  - Berson سال ۱۹۹۲: حمله تفاضلی به یک دور الگوریتم
  - Dobbertin سال ۱۹۹۶: تصادم در تابع فشرده‌ساز

توابع چکیده‌ساز مهم: SHA-1

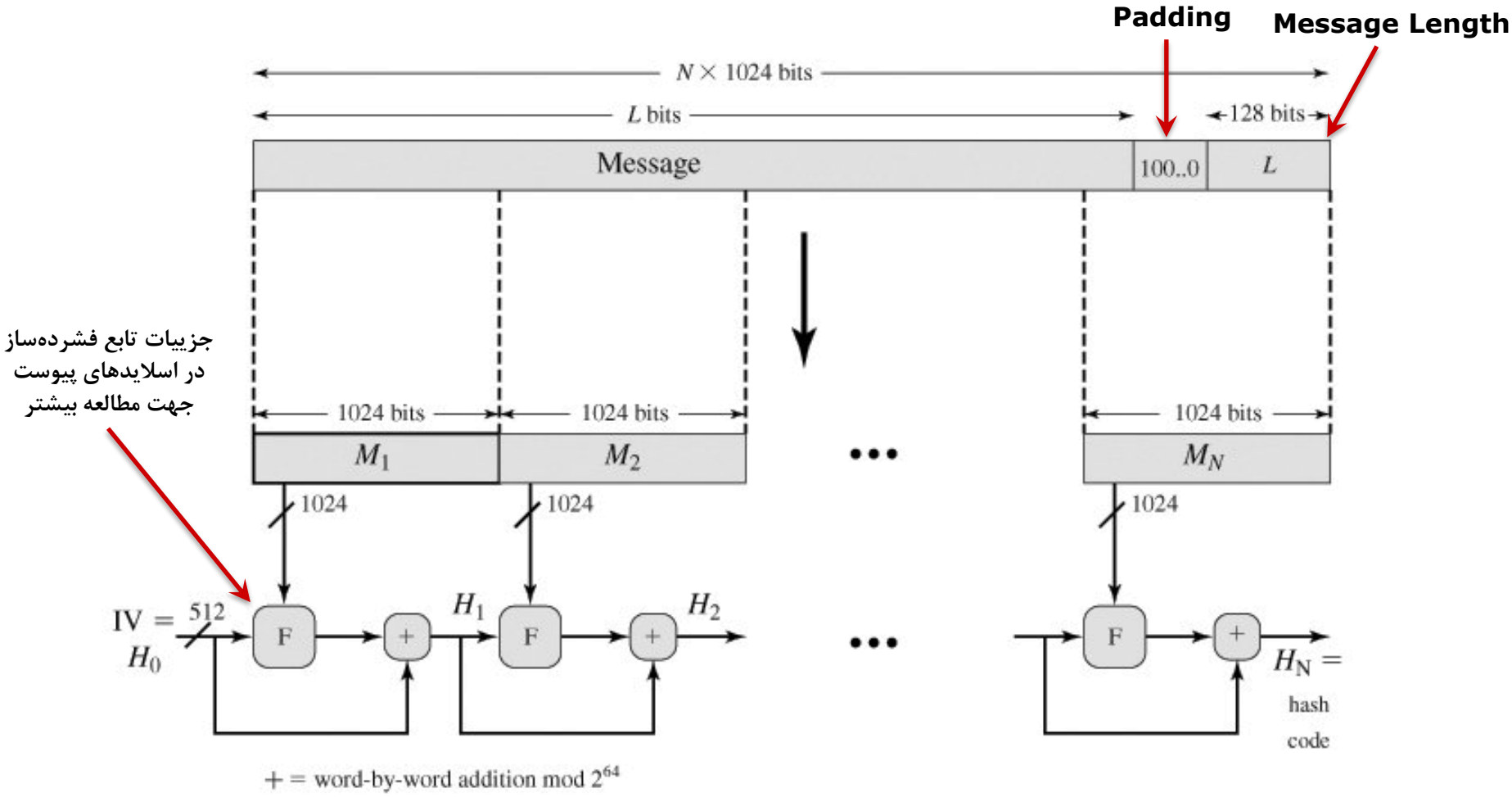
- SHA-1: Secure Hash Algorithm – 1
  - استاندارد NIST، ۱۹۹۵
  - طول ورودی کوچکتر از $۲^{۶۴}$ بیت
  - طول خروجی ۱۶۰ بیت
  - استفاده شده در استاندارد امضای دیجیتال DSS
- امنیت:
  - مقاومت در برابر تصادم (قوی) تحت حمله روز تولد: $۲^{۸۰}$
  - در سال ۲۰۰۵ توانستند با $۲^{۶۹}$ عمل یک تصادم در آن بیابند.
  - در آمریکا از سال ۲۰۱۰ الزام شد که با گونه‌های امن‌تر آن یعنی خانواده SHA-2 جایگزین شود.

توابع چکیده‌ساز مهم: SHA-2

- نسخه‌های زیر نیز علاوه بر SHA-1 استاندارد شده‌اند:
  - SHA-256، SHA-384 و SHA-512
  - معروف به **خانواده SHA-2** هستند.
  - از لحاظ ساختار و جزئیات مشابه SHA-1 هستند.

Algorithm	Digest size	Block size	Message size	Security
SHA-1	160	512	$< 2^{64}$	80 bits
SHA-256	256	512	$< 2^{64}$	128 bits
SHA-384	384	1024	$< 2^{128}$	192 bits
SHA-512	512	1024	$< 2^{128}$	256 bits

الگوریتم SHA-512



جزییات تابع فشرده‌ساز در اسلایدهای پیوست جهت مطالعه بیشتر

حمله افزایش طول

- MD5، SHA-1 و تمامی توابع خانواده SHA-2 در برابر حملات افزایش طول آسیب‌پذیر هستند.
  - اگر بخواهیم پیام m را به صورت $H(K \| m)$ احراز صحت کنیم، مهاجم با دانستن m برای مقدار دلخواه m' می‌تواند به سادگی مقدار $H(K \| m \| pad \| L \| m')$ را بدست آورد.
  - در این حالت، مهاجم می‌تواند m را با $m \| pad \| L \| m'$ جایگزین نماید.

حمله افزایش طول

- برای حل این مشکل، می‌توان:
  - طول پیام را به عنوان قطعه نخست به ساختار تابع چکیده‌ساز داد.
  - برای قطعه آخر، از یک تابع فشرده‌ساز متفاوت بهره گرفت.

تابع چکیده‌ساز SHA-3

- NIST در سال ۲۰۰۷ مسابقه‌ای را برای انتخاب تابع چکیده‌ساز جدید و معرفی آن به عنوان استاندارد **SHA-3** آغاز کرد.
- از شرایط SHA-3 آن است که حمله افزایش طول به آن وارد نباشد.
- ساختارهای جدید (غیر مرکل-دمگارد) بیشتر مورد استقبال هستند.
- SHA-3 در حال حاضر به عنوان جایگزین SHA-2 مطرح نیست و هر دو الگوریتم به عنوان الگوریتم‌های استاندارد قابل استفاده هستند.

استاندارد SHA-3

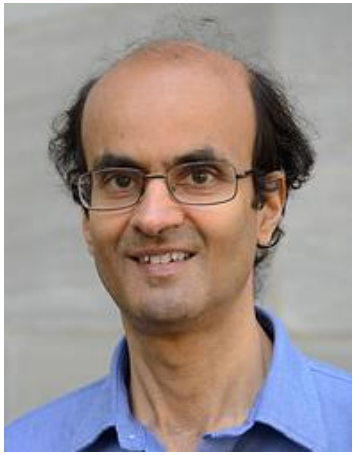
- در سال ۲۰۱۲ تابع چکیده‌ساز Keccak به عنوان برنده و تابع SHA-3 تعیین گردید.
  - **تلفظ رسمی**: Catch-Ack
  - **ساختار**: توابع اسفنجی (Sponge)
  - **طراحان**: Bertoni، Daemen (طراح Rijndael)، Peeters و Van Assche.
- استاندارد SHA-3 در آگوست ۲۰۱۵ منتشر شد.
- طول ورودی: دلخواه؛ طول خروجی ۲۲۴، ۲۵۶، ۳۸۴، ۵۱۲، و دلخواه.

فهرست مطالب

- مفاهیم اولیه
- رمزگذاری پیام و کدهای تشخیص خطا
- کدهای احراز صحت پیام
- اصول توابع چکیده‌ساز
- توابع چکیده‌ساز مهم
- **HMAC**

کد احراز اصالت HMAC

- ابداع توسط بلّاری، کانِتی و کِرَفْچیک در سال ۱۹۹۶.



Mihir Bellare

Ran Canetti

Hugo Krawczyk

کد احراز اصالت HMAC

- HMAC یک الگوریتم احراز صحت پیام است.
- HMAC اساساً روشی برای ترکیب کردن کلید مخفی با الگوریتم‌های چکیده‌ساز فعلی است.
- حمله افزایش طول به HMAC وارد نیست.
- برای تولید چکیده پیام، از توابع چکیده‌ساز استفاده شده است.
  - در مقابل استفاده از رمزهای قطعه‌ای
  - بدلیل مزایای عملی توابع چکیده‌ساز

کد احراز اصالت HMAC

- HMAC جزو ملزومات پیاده‌سازی IPSec است.
- HMAC به طور گسترده استفاده می‌شود (مثلاً SSL).

اهداف طراحي HMAC

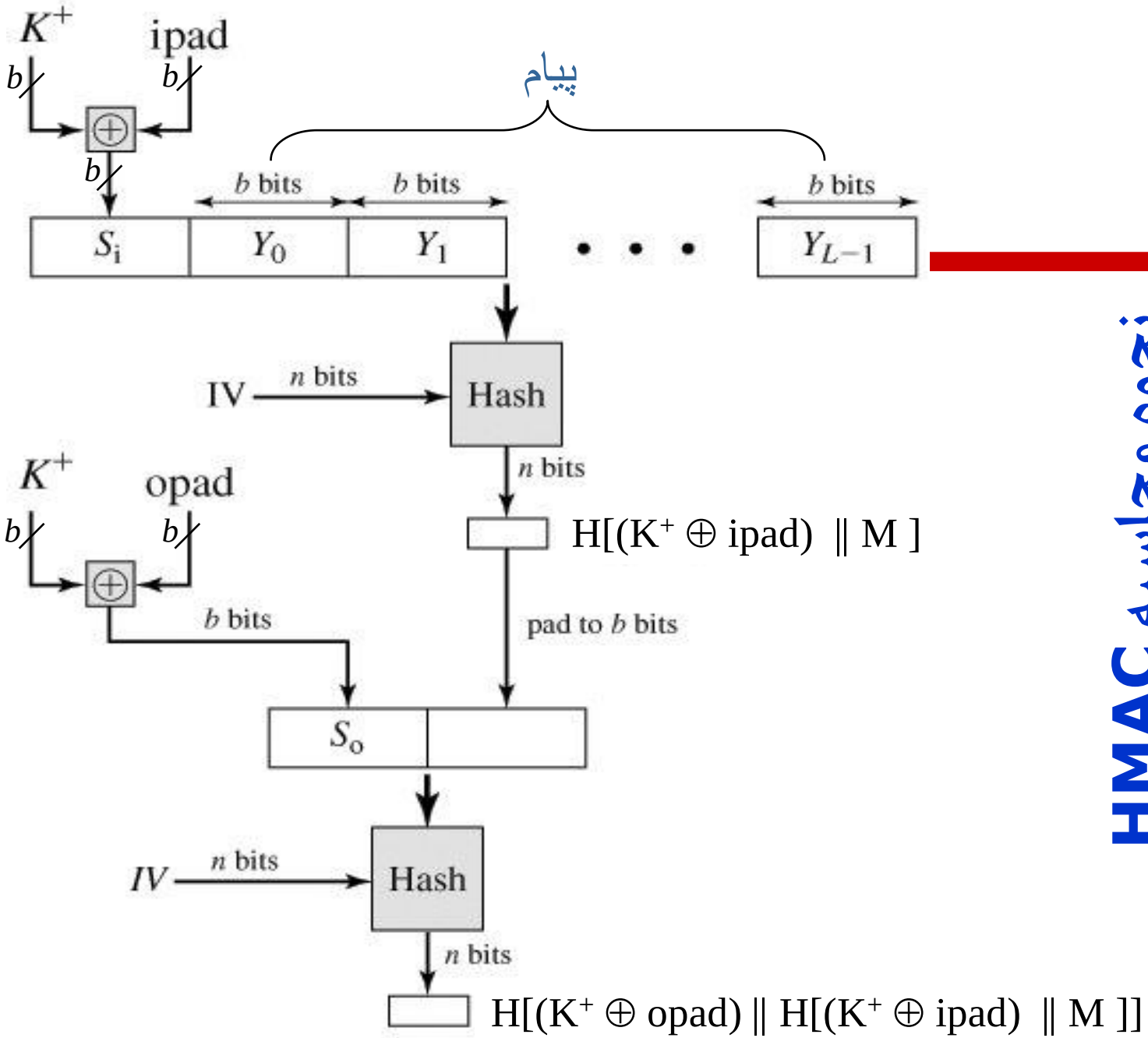
- استفاده از توابع چکیده‌ساز بدون تغییر آنها
- پشتیبانی از توابع چکیده‌ساز متنوع
  - مانند MD5، SHA-1، SHA-2، Whirlpool و RIPEMD-160
- حفظ کارایی و سرعت تابع چکیده‌ساز به کار گرفته شده
- استفاده ساده از کلید
- طراحی روشن و بدون ابهام

الگوریتم HMAC

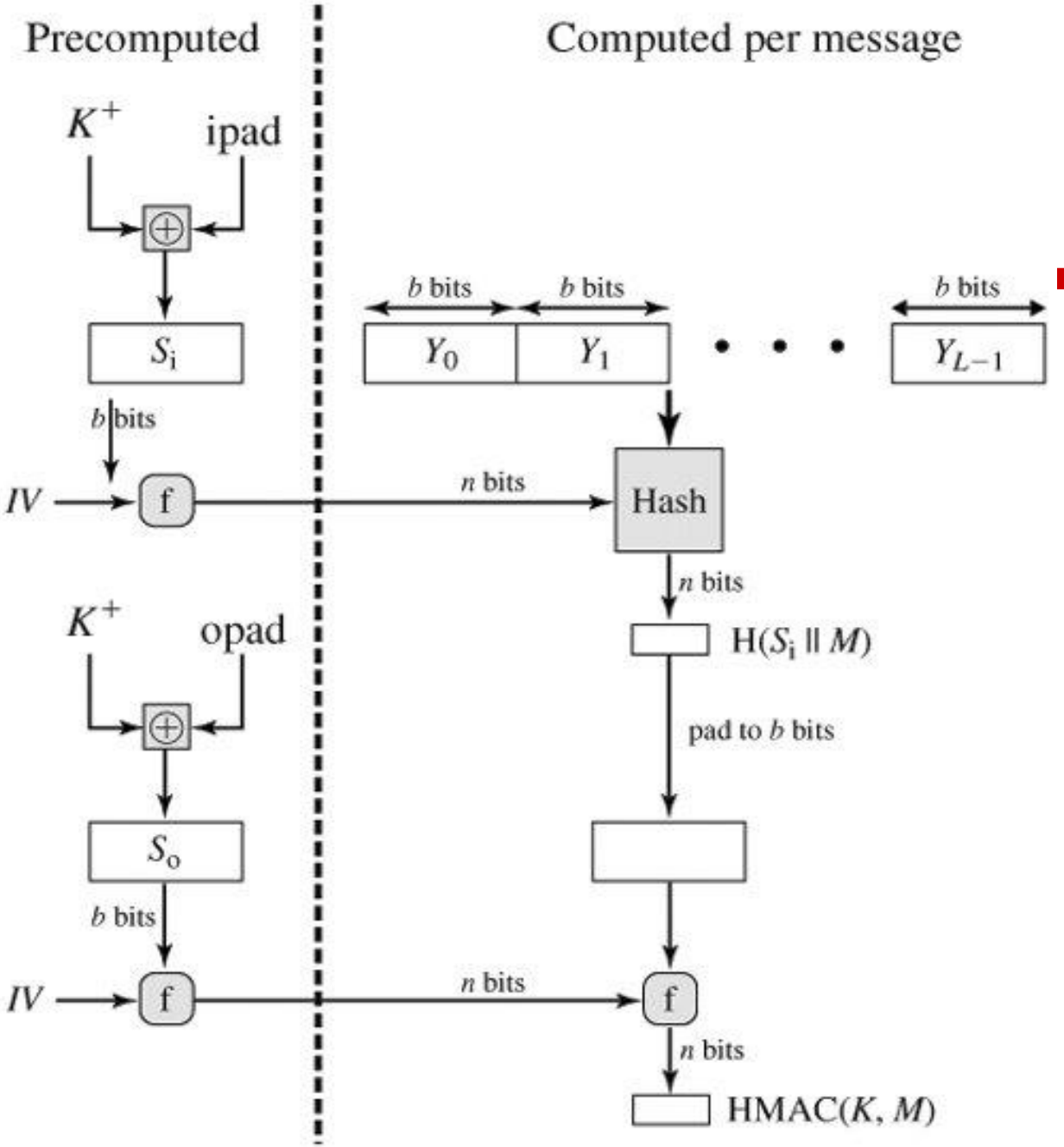
- H: تابع چکیده‌ساز به کار گرفته شده (با خروجی n بیتی)
- M: پیام ورودی (با قطعات b بیتی)
- K: کلید مخفی (طول پیشنهادی بیشتر از n، در صورتیکه طول بیشتر از b بیت باشد، کاهش به n بیت با استفاده از توابع چکیده‌ساز)
- K^+: کلید مخفی که یک دنباله صفر به سمت چپ آن اضافه شده است (تا به طول b برسد)
- ipad : رشته b بیتی حاصل از تکرار رشته ۰۰۱۱۰۱۱۰ به تعداد b/8
- opad: رشته b بیتی حاصل از تکرار رشته ۰۱۰۱۱۰۱۰ به تعداد b/8

$$HMAC(K,M) = H\left[(K^+ \oplus opad) \,\|\, H[(K^+ \oplus ipad) \,\|\, M\,]\right]$$

نحوه محاسبه HMAC



پیاده‌سازی کارای HMAC



امنیت HMAC

- ارتباط دقیق بین امنیت HMAC با امنیت تابع در هم‌ساز اثبات شده است.
- حمله به HMAC نیاز دارد به
  - حمله آزمون جامع بر روی کلید (میزان مقاومت بسته به طول کلید)
  - حمله روز تولد که با توجه به نداشتن کلید نیازمند مشاهده تعداد زیادی پیام و MAC آنهاست که از کلید یکسانی در آنها استفاده شده است.
- مقاومت HMAC در برابر حمله روز تولد از تابع چکیده‌ساز به کار گرفته شده، بیشتر است.
  - لذا استفاده از MD5 در هنگام نیاز به سرعت بیشتر مجاز است.

پایان

توابع چکیده‌ساز ساده

تابع چکیده‌ساز ساده XOR

- XOR قطعات داده به عنوان خروجی تابع.
- اگر داده ورودی m قطعه n بیتی باشد:

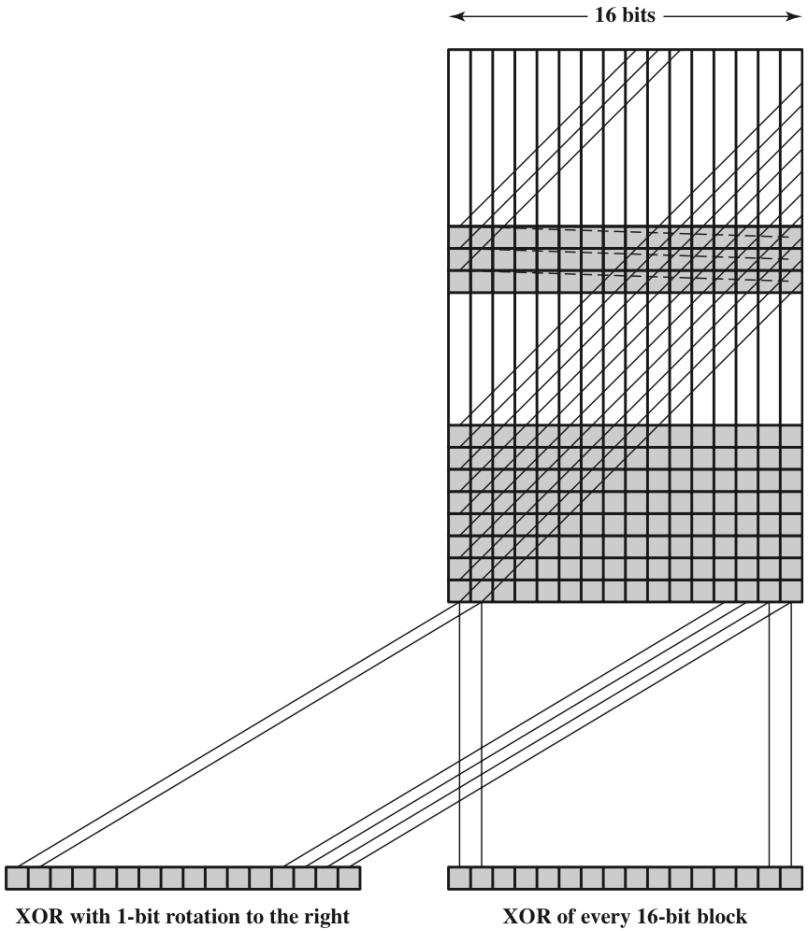
$$C_i = b_{i1} \oplus b_{i2} \oplus \ldots \oplus b_{im} \quad (1 \le i \le n)$$

- احتمال عدم تغییر چکیده در صورت وجود خطا = 2^{-n}
- در متون عادی، بیت بالای هر بایت معمولا صفر است (مگر اینکه کاراکتر خاصی باشد که کد اسکی آن بالای ۱۲۸ باشد).
- درعمل تاثیر این تابع ۱۲۸ بیتی از 2^{-128} به 2^{-112} کاهش می‌یابد.

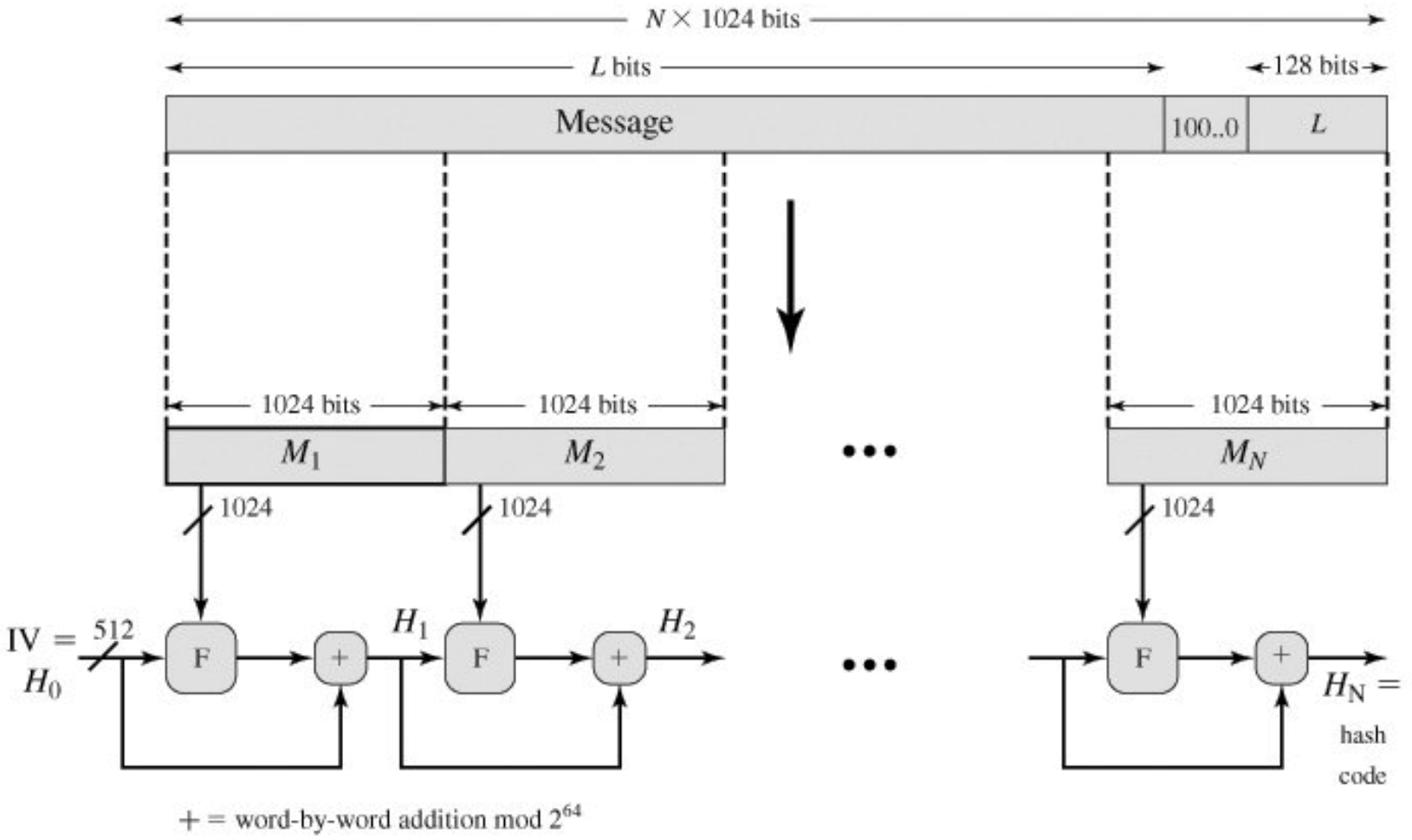
توابع چکیده‌ساز ساده

تابع چکیده‌ساز ساده RXOR

- در هر مرحله قبل از XOR کردن قطعه جدید با حاصل مراحل قبل، یک شیفت چرخشی تک بیتی به چپ انجام می‌دهد.
- با توجه به سادگی پیدا کردن تصادم در این تابع، نمی‌توان آن را برای احراز صحت پیام‌هایی که آشکار ارسال می‌شوند (مشابه آنچه که در اسلاید ۳۶ و ۳۷ آمده) استفاده کرد.



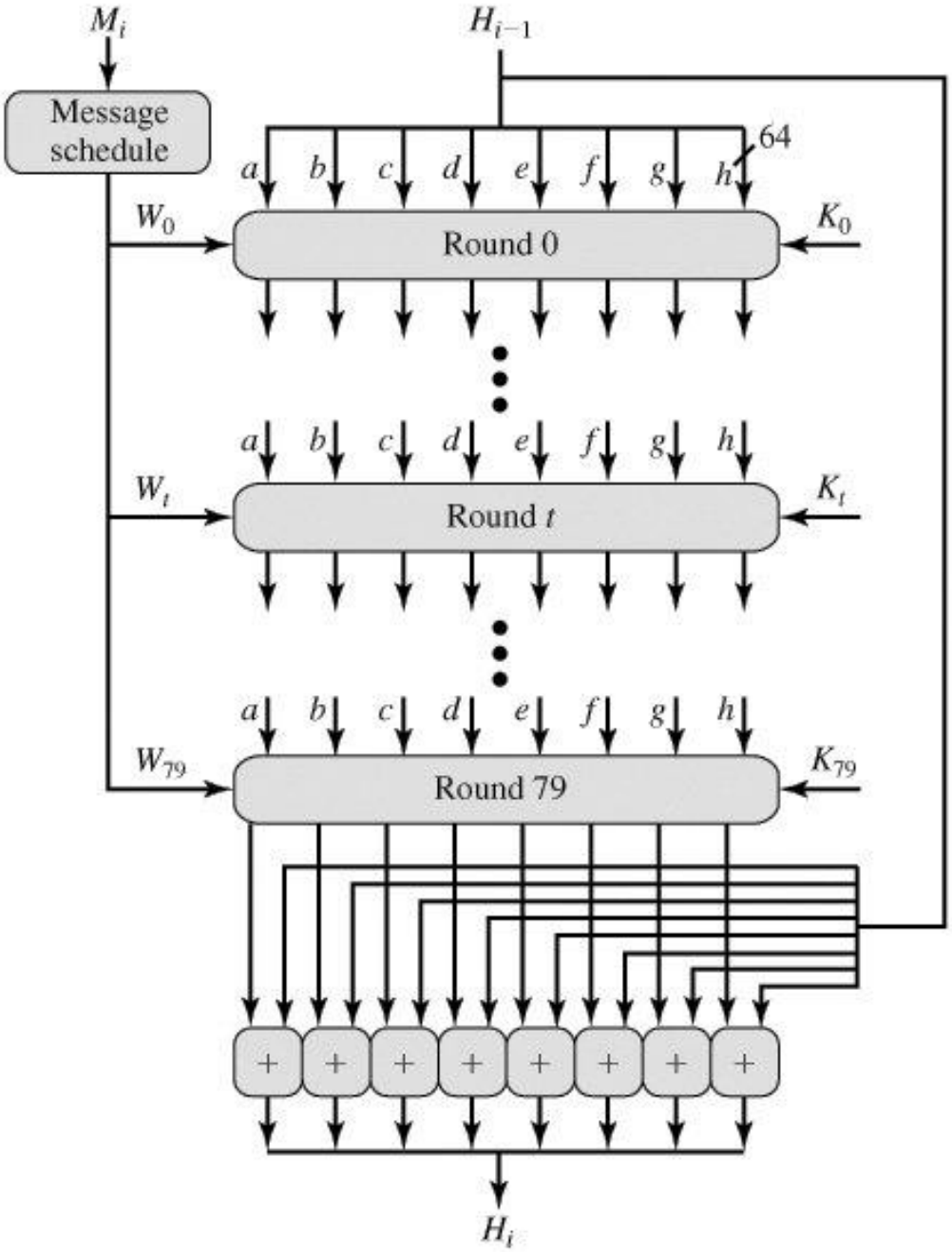
الگوریتم SHA-512



مراحل اجرای الگوریتم SHA-512

- افزودن بیتهای padding
  - افزودن 1000...0 به اندازه ای که طول پیام هم‌نهشت با ۸۹۶ شود.
- افزودن اندازه پیام به انتهای آن
  - ثبت طول پیام در ۱۲۸ بیت باقیمانده از قطعه آخر
- مقداردهی اولیه بافر hash
  - مقدار اولیه H_0 در ۸ ثبات ۶۴ بیتی abcdefgh ذخیره می‌شود.
- پردازش پیام در قطعات ۱۰۲۴ بیتی (۱۲۸ کلمه‌ای)
  - هر قطعه در ۸۰ دور طبق اسلاید بعد پردازش می شود.

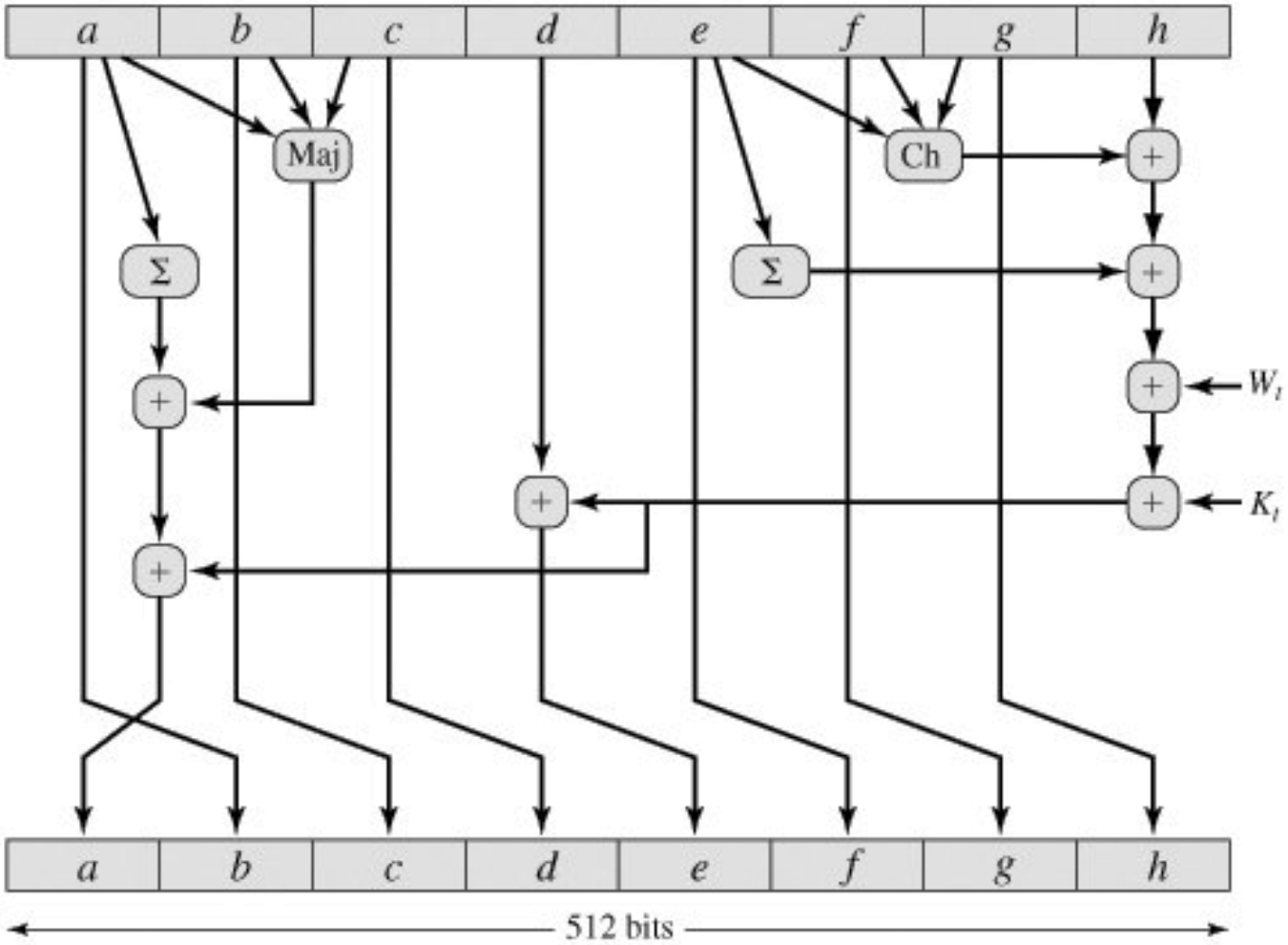
۶۴ بیت اول قسمت اعشاری جذر ۸ عدد اول نخستین



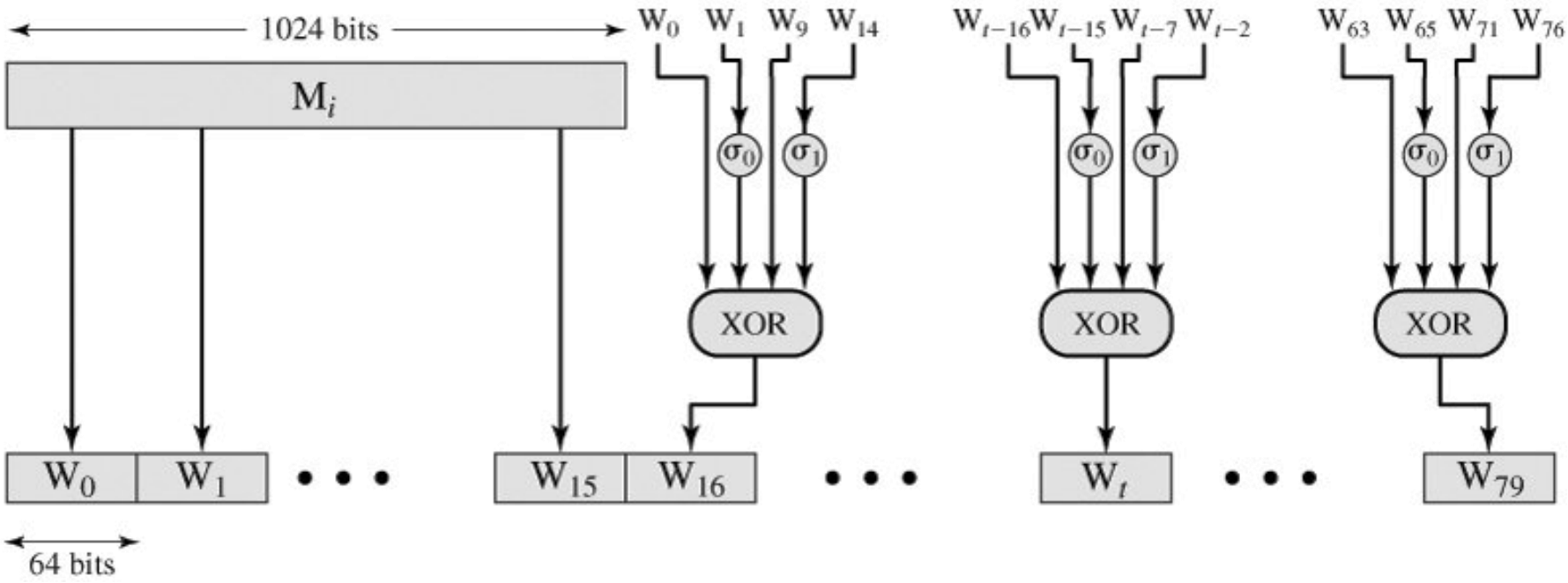
پردازش یک قطعه در SHA-512

- K_iها ثابت هستند.
  - K_iها شامل ۶۴ بیت اول قسمت اعشاری ریشه سوم ۸۰ عدد اول نخستین هستند.
- W_iهای ۶۴ بیتی توسط زمانبند پیام تولید می شوند.

عملیات هر دور در SHA-512



زمانبند پیام در SHA-512



عملگرهای مورد نیاز در SHA-512

$\text{Ch}(e, f, g) = (e \text{ AND } f) \oplus (\text{NOT } e \text{ AND } g)$
the conditional function: If e then f else g

$\text{Maj}(a, b, c) = (a \text{ AND } b) \oplus (a \text{ AND } c) \oplus (b \text{ AND } c)$
the function is true only of the majority (two or three) of the arguments are true

$\left(\sum_0^{512} a\right) = \text{ROTR}^{28}(a) \oplus \text{ROTR}^{34}(a) \oplus \text{ROTR}^{39}(a)$

$\left(\sum_1^{512} e\right) = \text{ROTR}^{14}(e) \oplus \text{ROTR}^{18}(e) \oplus \text{ROTR}^{41}(e)$

$\text{ROTR}^n(x) =$ circular right shift (rotation) of the 64-bit argument x by n bits

$\sigma_0^{512}(x) = \text{ROTR}^1(x) \oplus \text{ROTR}^8(x) \oplus \text{SHR}^7(x)$

$\sigma_1^{512}(x) = \text{ROTR}^{19}(x) \oplus \text{ROTR}^{61}(x) \oplus \text{SHR}^6(x)$

$\text{ROTR}^n(x) =$ circular right shift (rotation) of the 64-bit argument x by n bits

$\text{SHR}^n(x) =$ left shift of the 64-bit argument x by n bits with padding by zeros on the right

$+ =$ addition modulo 2^{64}