

امنیت داده ها

فصل یازدهم : چکیده پیام و امضاء دیجیتال

دکتر یعقوب فرجامی

عضو هیات علمی دانشکده فنی قم

مفهوم احراز اصالت پیام

• اطمینان از:

– تمامیت پیام؛ یعنی پیام دریافتی دستکاری نشده است:

• بدون تغییر،

• بدون درج،

• بدون حذف

Data Integrity

• بدون تکرار و تغییر توالی

– این که پیام از جانب فرستنده ادعا شده ارسال شده است

**Data Origin
Authentication**



اهمیت اصالت پیام

در بسیاری از کاربردها مانند:

- تراکنشهای مالی

- ثبت احوال و اسناد

- بانکهای اطلاعاتی

در این موارد ممکن است ارائه سرویس محرمانگی اهمیت زیادی نداشته باشد ولی اینکه محتوای اطلاعات قابل اعتماد باشند از اهمیت بسیار بالاتری برخوردار است.

DIGITAL SIGNATURE

- در این قسمت به بررسی چکیده سازی یا هشینگ یک پیام میپردازیم.
- گرچه از نظر مفهومی درست نیست ولی معمولاً چکیده سازی بعنوان نوعی از امضاء شناخته میشود.
- کلمه امضاء در فارسی بمعنی تایید و گواهی صحت یک متن یا یک پیام از سوی امضاء کننده است. انگلیسی آن signature است
- همین کلمه signature انگلیسی در فارسی معنای دیگری هم دارد و آن اثر، رد، ردپا، علائم و شواهد و ... دارد که در این درس منظور از چکیده سازی همین معناست. گرچه بعداً خواهیم دید که در امضای دیجیتال توسط یک شخص نیز چکیده سازی استفاده میشود.
- معمولاً برای چکیده سازی از روش تابعهای هشینگ یا درهم ساز استفاده میشود و معمولاً نام digesting به آن داده میشود.
- مثلاً اگر پیام عبارت 1000101011101010 باشد آنگاه چکیده آن میتواند بصورت مجموع ارقام آن باشد که میشود 1000
- حال اگر پیام اصلی در ارسال دارای یک بیت یا چند بیت خطا باشد در مقصد به احتمال زیاد پیام و چکیده باهم سازگار نخواهند بود که نشان میدهد پیام اصلی تغییر یافته است.
- خصوصیات امضاء دیجیتال
 1. تشخیص جعلی نبودن هویت صاحب سند، برای دریافت کننده سند
 2. عدم انکار محتوی سند توسط صاحب سند
 3. عدم ساخت اسناد جعلی و نسبت دادن آن دیگران
- انواع روش های تولید امضاء
 1. امضاء دیجیتال مبتنی بر چکیده پیام
 2. امضاء دیجیتالی کلید متقارن مبتنی بر مرکز مورد اعتماد گواهی امضاء
 3. امضاء دیجیتال مبتنی بر رمزنگاری کلید عمومی

DIGITAL SIGNATURE WITH MESSAGE DIGEST

○ نحوه تولید امضاء با چکیده پیام شامل :

1. تولید یک چکیده کوتاه چند بایتی متاثر از کل پیام

2. رمز کردن چکیده با کلید خصوصی صاحب پیام

3. ضمیمه کردن چکیده رمز شده به پیام

○ اعتبارسنجی و تایید اصالت سند :

1. باز کردن چکیده رمز شده با کلید عمومی صاحب پیام توسط گیرنده

2. محاسبه مجدد چکیده از روی پیام اصلی

3. در صورت برابری چکیده از روش ۱ و ۲ اصالت پیام تایید می شود



DIGESTING PRINCIPLES

- محاسبه ساده و سریع چکیده پیام ($MD(p)$)، با داشتن پیام مشخص (p)
 - عدم توانایی رسیدن از چکیده پیام به خود پیام
 - به ازای هر دو پیام متفاوت p و p' ، هرگز چکیده آن ها برابر نخواهد بود
- $$p \neq p' \rightarrow MD(p) \neq MD(p')$$
- به تابعی که از یک متن با طول متغیر چکیده ای با طول کوتاه و ثابت بدست می آورد «تابع درهم ساز» و به چکیده «کد درهم شده یا hash» گویند

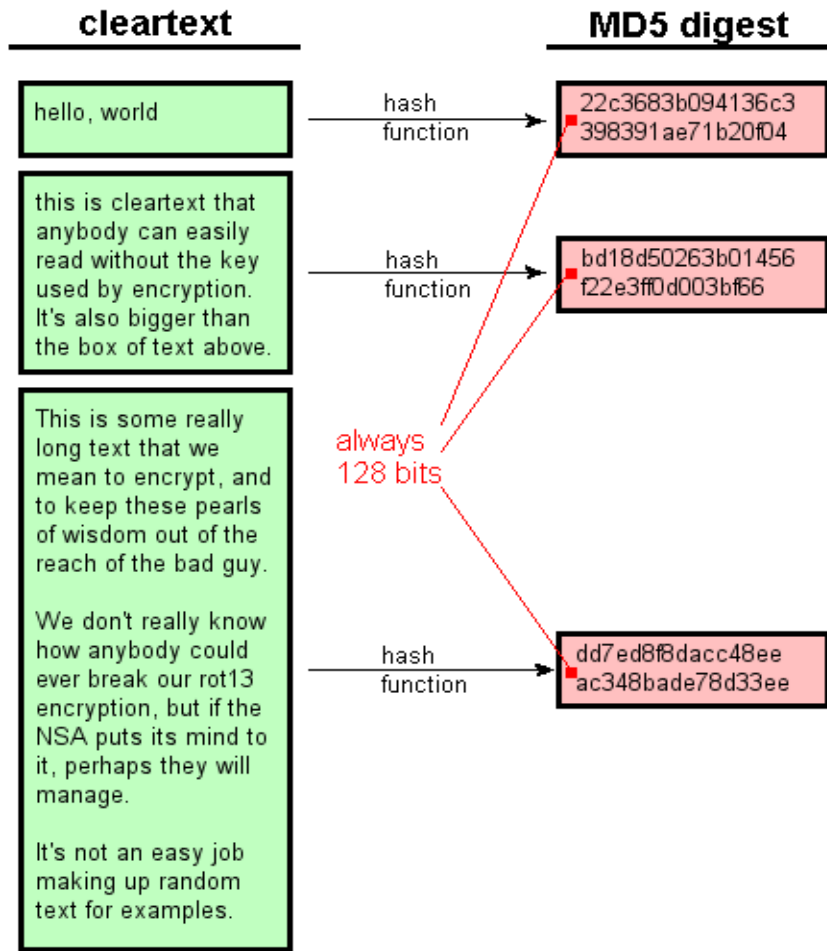
نکته : این تابع کاملاً یکطرفه عمل می کند

نکته : از لحاظ تئوری امکان اینکه دو پیام به یک چکیده یکسان برسند وجود دارد (فضای چکیده بسیار کوچکتر از فضای پیام است)، اما در عمل پیدا کردن دو متن با مفهوم با چکیده یکسان مشکل است



خصوصیات الگوریتم‌های HASHING

- معمولا در بیشتر الگوریتم‌های Hashing اندازه خروجی آنها (طول رشته خروجی) به اندازه ورودی بستگی نداشته و ثابت است.



موارد استفاده از Hash ها :

hash کردن کلمه عبور (Hashing passwords)
تشخیص تمامیت یک فایل (Verifying file integrity)



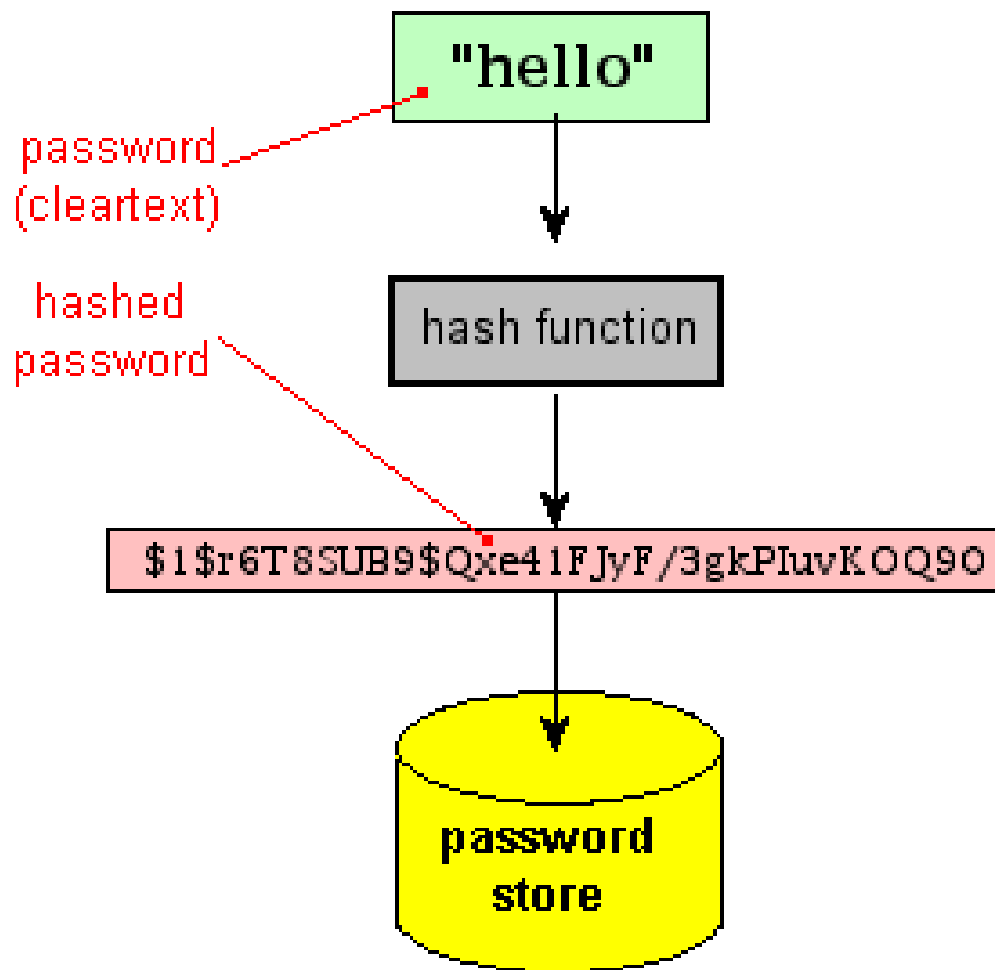
تایید هویت با استفاده از hash کردن کلمه عبور

بازیابی کلمه عبور اصلی از روی رشته هش تقریبا غیر ممکن است
(عدم معکوس پذیری هش)

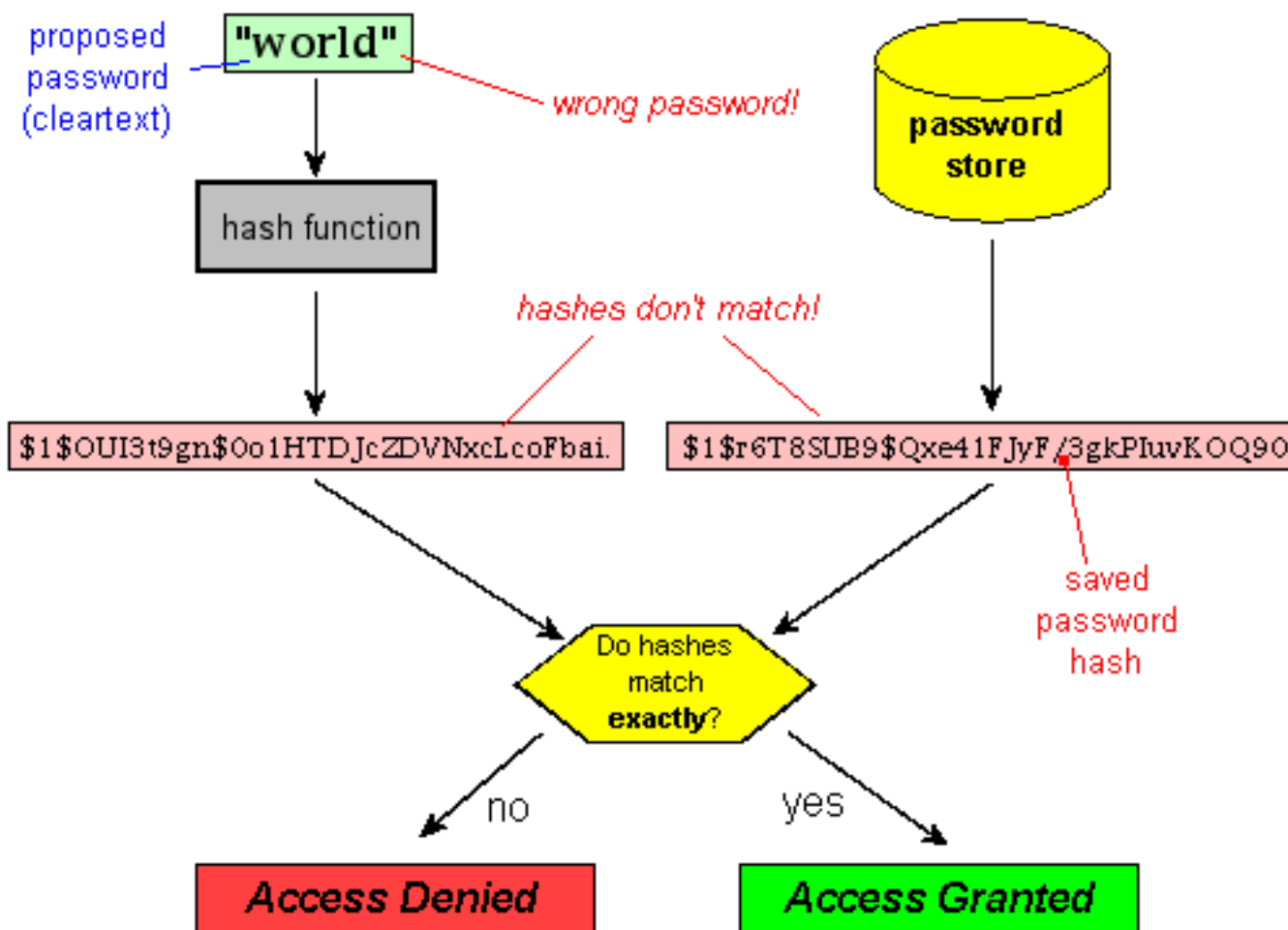
با تولید رشته هش کلمه عبور وارد شده توسط کاربر و مقایسه آن
با رشته هش ذخیره شده در رکورد بانک اطلاعاتی مربوط به کاربر
می توانید متوجه شوید که آیا دو رشته با هم برابرند یا نه.



کاربرد HASH در ذخیره سازی کلمه عبور



کاربرد HASH در چک کردن صحت کلمه عبور (AUTHENTICATION)



تشخیص تمامیت یک فایل

در سمت فرستنده HASH پیغام ارسالی محاسبه شده و با کلید خصوصی رمزنگاری میشود و سپس به طرف گیرنده ارسال میشود

در سمت گیرنده پس از دریافت پیغام و رمزگشایی آن HASH مجددا محاسبه میشود

HASH دریافت شده از فرستنده نیز رمزگشایی میشود و با HASH محاسبه شده مقایسه میشود، اگر مطابقت داشت تمامیت داده ها احراز میشود



الگوریتم‌های HASHING

○ معروفترین روشهای Hashing :

- MD5(Message Digest algorithm 5)
128bits
- SHA-1 (Secure Hash Algorithm)(160 bits)
- SHA-256
- SHA-384
- SHA-512



توابع درهم ساز 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



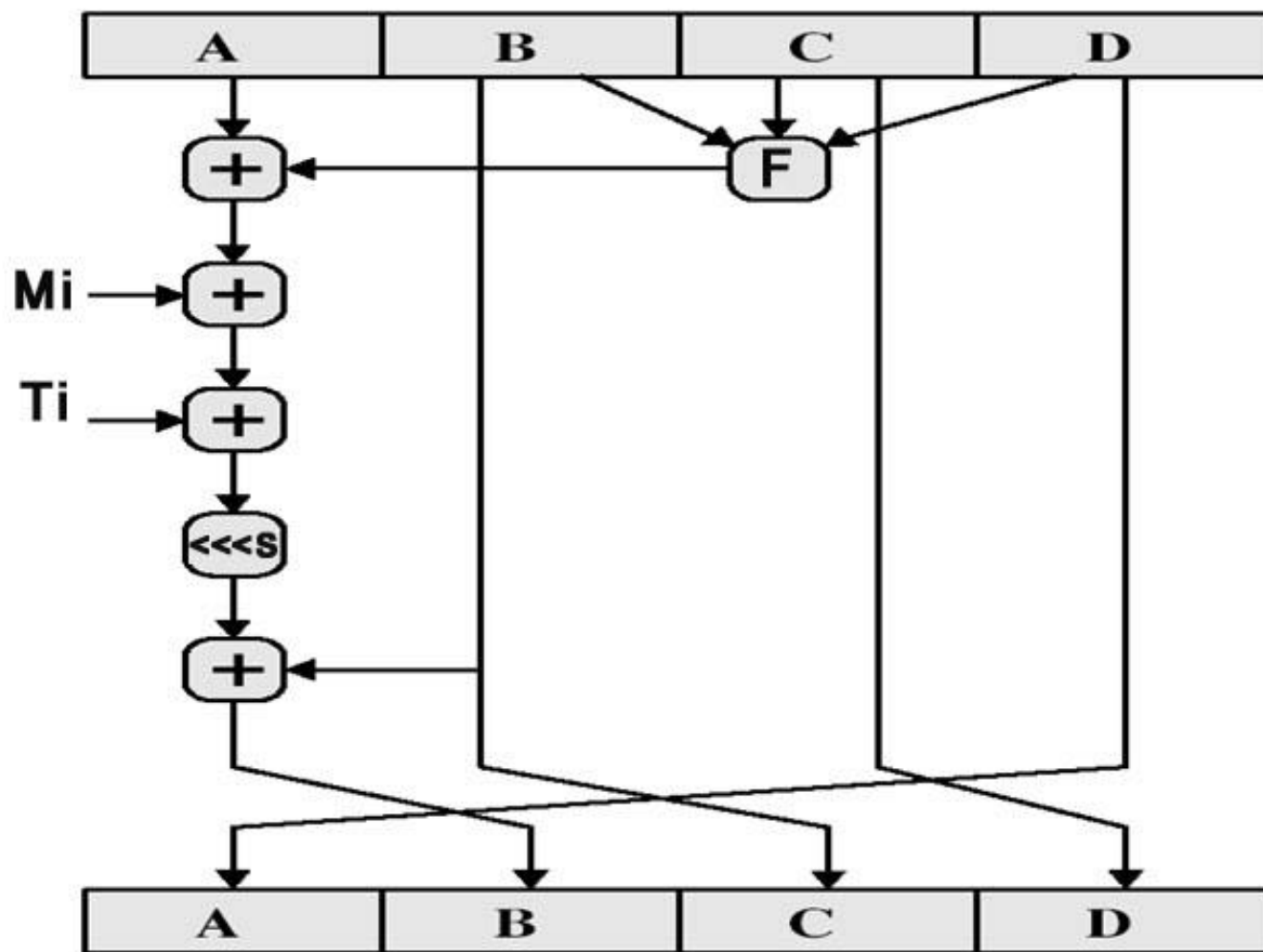
امنیت MD5

- مقاومت در برابر تصادم (قوی) تحت حمله آزمون جامع: ۲۶۴
 - امروزه امن محسوب نمیشود.
- حملات کارگر به این الگوریتم یافت شده اند:
 - Berson سال ۱۹۹۲: حمله تفاضلی به یک دور الگوریتم
 - Bosselaers و Boer سال ۹۳: یافتن تصادم های مجازی
 - Dobbertin سال ۹۶: تصادم در تابع فشرده ساز

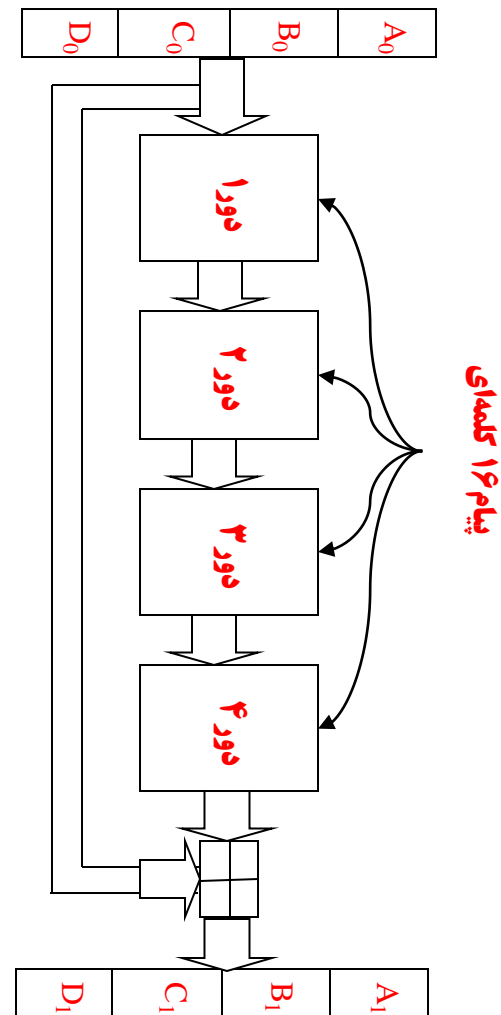
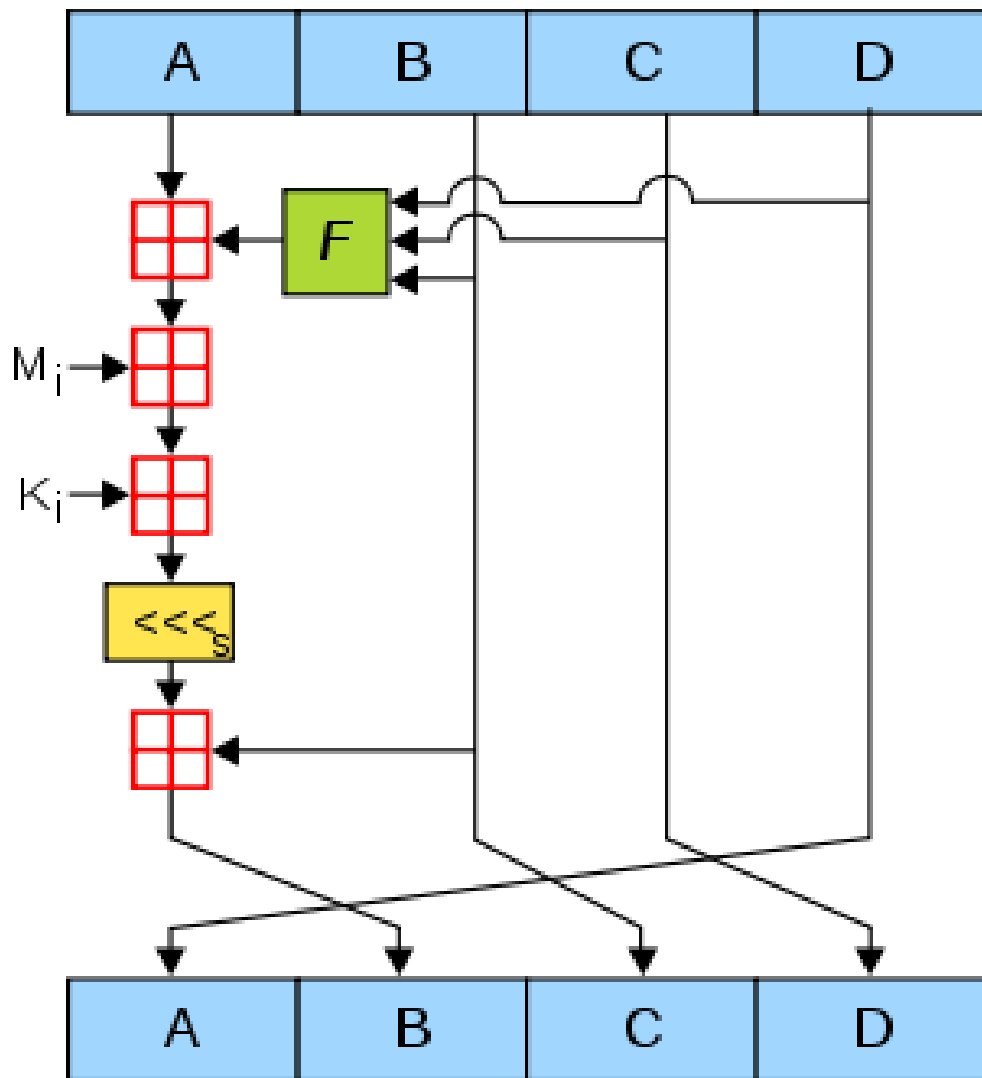


MD5

یکی از مشهورترین روشهای چکیده سازی که هنوز هم بکار میرود



شمای الگوریتم MD5



MD5

○ متن با هر طولی به یک چکیده ۱۲۸ بیتی تبدیل می شود

○ نحوه کار :

○ متن به بلوک های ۵۱۲ بیتی شکسته می شود

○ بلوک آخر باید ۴۴۸ بیتی باشد تا ۶۴ بیتی که به انتهای آن افزوده می شود نشان دهنده طول واقعی متن باشد

○ وجود ۴ متغیر کمکی A، B، C و D هر کدام ۳۲ بیت که جمعاً میشوند ۱۲۸ بیت

○ شکستن بلوک ۵۱۲ بیتی به ۱۶ کلمه ۳۲ بیتی و ذخیره در آرایه $M[0-15]$

○ ورود به یک حلقه با ۶۴ دور، که ۶۴ دور در ۴ دور کوچکتر هر کدام ۱۶ دور تقسیم بندی می شوند

Round 1: $i=0-15$ $k=0-15$

Round 2: $i=16-31$ $k=0-15$

Round 3: $i=32-47$ $k=0-15$

Round 4: $i=48-63$ $k=0-15$



MD5

○ نحوه عملکرد تابع F برای هر یک از ۴ دور :

Round No.	$F(b, c, d)$
Round 1	$(b \wedge c) \vee (\neg b \wedge d)$
Round 2	$(b \wedge d) \vee (c \wedge \neg d)$
Round 3	$b \oplus c \oplus d$
Round 4	$c \oplus (b \vee \neg d)$



MD5

○ شروع عملیات اصلی :

۱- کپی مقادیر اولیه از چهار متغیر اصلی H_0, H_1, H_2 و H_3 به ۴ متغیر کمکی A, B, C و D که بر این اساس مقدار اولیه آنها به شرح زیر است،

A = 01 23 45 67

B = 89 ab cd ef

C = fe dc ba 98

D = 76 54 32 10

۲- مقادیر B, C و D به عنوان ورودی تابع F

۳- جمع معمولی A با خروجی F (جمع در مبنای 2^{32}) حذف (carry)

۴- حاصل مرحله ۳ با $M[i]$ این مرحله جمع معمولی می شود

۵- جمع معمولی حاصل مرحله قبل با $T[i]$

نکته : مقادیر آرایه $T[i]$ از قبل وجود داشته و برابر

For $i = 0$ to 63

$$T[i] = \text{floor}(\text{abs}(\sin(i+1)) \times 2^{32})$$



MD5

۶- شیفٲ چرخشی به چپ، که اندازه چرخش در هر مرحله متفاوت و از روی جدول $r[]$ مشخص می شود

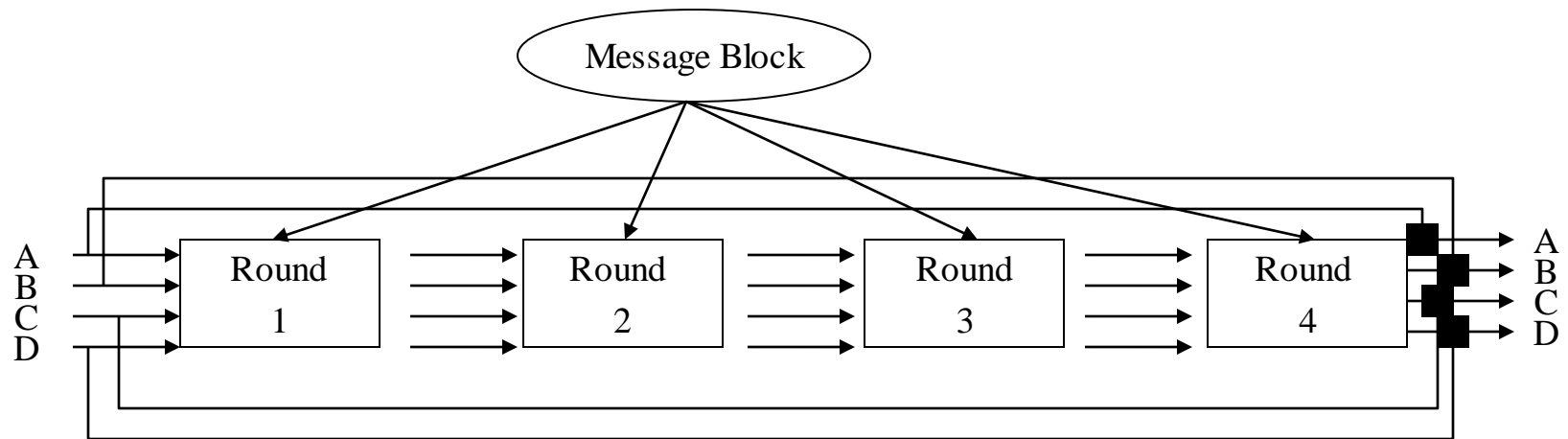
7	12	17	22	7	12	17	22	7	12	17	22	7	12	17	22
5	9	14	20	5	9	14	20	5	9	14	20	5	9	14	20
4	11	16	23	4	11	16	23	4	11	16	23	4	11	16	23
6	10	15	21	6	10	15	21	6	10	15	21	6	10	15	21

۷- حاصل مرحله ۶ با B جمع می شود

۸- در آخرین مرحله یک دور از ۶۴ دور، مقادیر B، C و D به ترتیب به C، D و A کپی می شوند و حاصل مرحله ۷ داخل B کپی می گردد



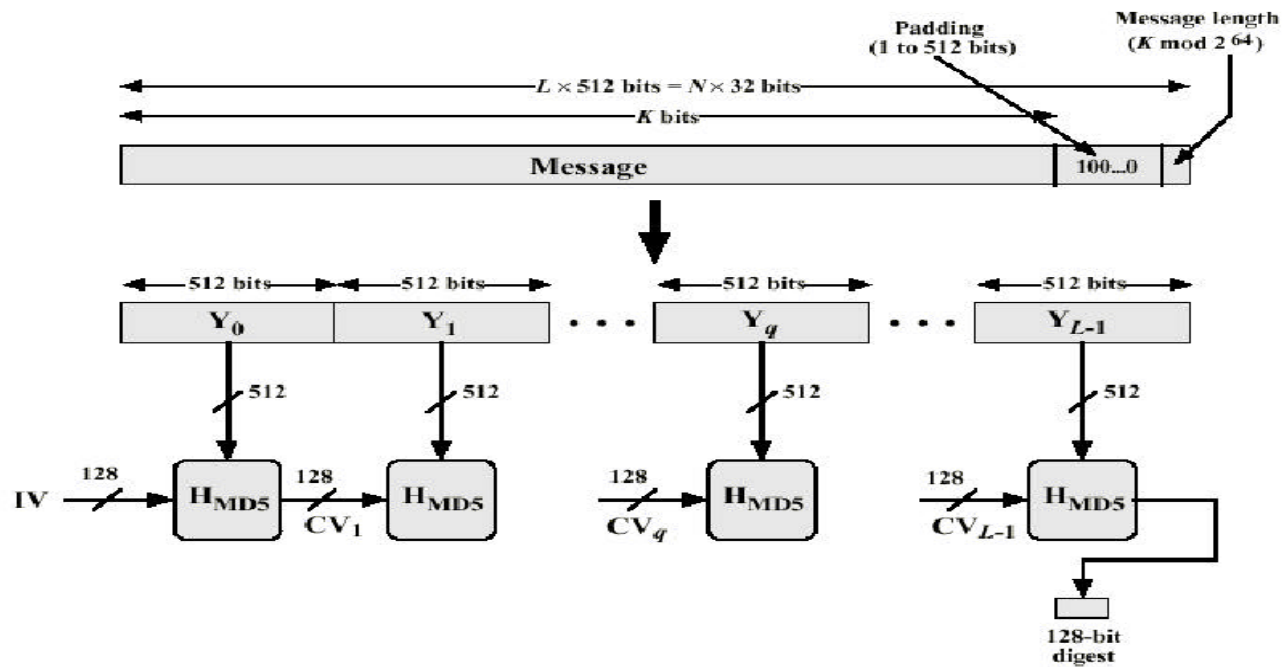
MD5 نمای کلی



(Each message block is 512 bits, and output from each iteration is input to next iteration.
Hash output is 128 bits)



MD5



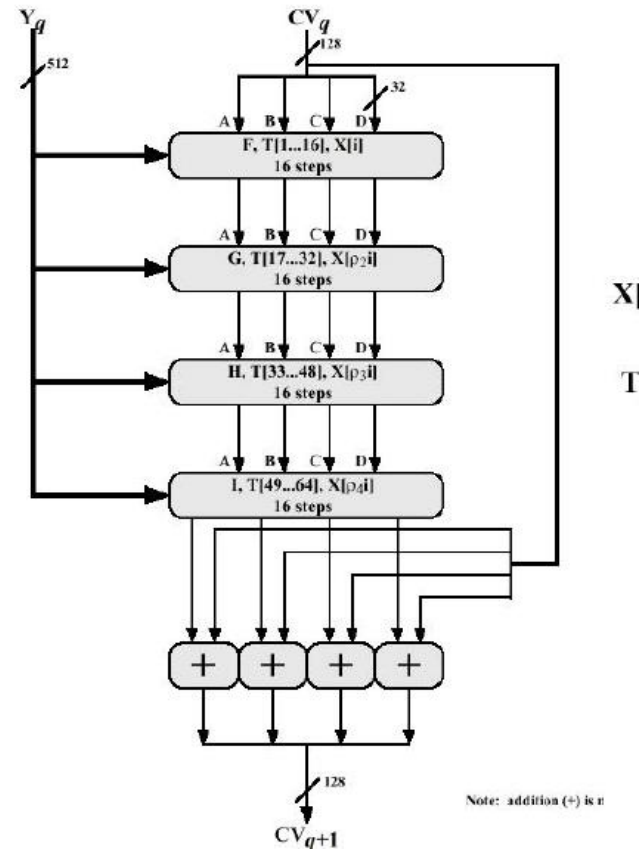
MD5 LOGIC

- Step 1. Appending padding bits
 - Congruent to 448 modulo 512
- Step 2. Appending length
 - Length modulo 64
 - $L * 512\text{-bit}$, $N * 32\text{ bit}$
 - $M[i] : i_{\text{th}}$ word
- Step 3. Initialize MD buffer
 - Four 32-bit register (A B C D), little endian
 - $A = 67482301$
 - $B = \text{EFCDAB89}$
 - $C = 98BADCFE$
 - $D = 10325476$



MD5 LOGIC

- Step 4. Process message in 512-bit blocks
 - Four rounds
 - Each round has a different primitive function, F, G, H and I
 - Use one-fourth of $T[i]$
 - $T[i] = 2^{32} * \text{abs}(\sin(i))$



تابع بولی G در MD5

Round	Primitive function g	$g(b,c,d)$
1	$F(b,c,d)$	$(b \wedge c) \vee (\neg b \wedge d)$
2	$G(b,c,d)$	$(b \wedge d) \vee (c \wedge \neg d)$
3	$H(b,c,d)$	$b \oplus c \oplus d$
4	$I(b,c,d)$	$c \oplus (b \vee \neg d)$



MD5 LOGIC

○ Step 5. Output

$$CV_0 = IV$$

$$CV_{q+1} = \text{SUM}_{32}(CV_q, RF_I[Y_q, RF_H[Y_q, RF_G[Y_q, RF_F[Y_q, CV_q]]]])$$

$$MD = CV_L$$

where

IV = initial value of the ABCD buffer

Y_q = the q th 512-block of the message.

L = the number of blocks

CV_q = chaining variable processed with the the q th block of the message

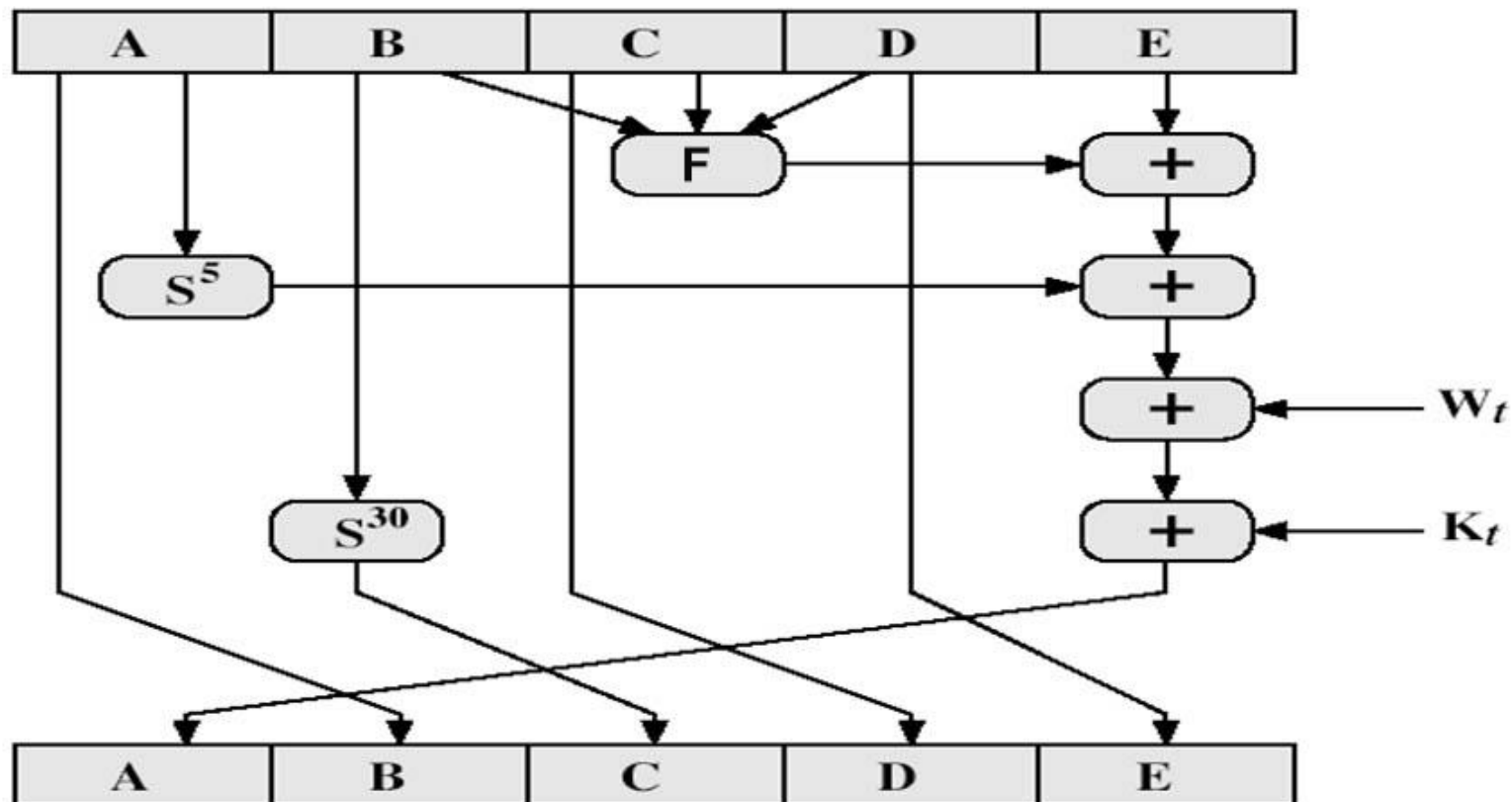
RF_X = round function using primitive logical function X

MD = final message digest value

SUM_{32} = Addition module 2^{32} performed separately on each pair of words of the two inputs.

SHA-1

استاندارد جاری و معمولی چکیده سازی



توابع درهم ساز مهم: SHA-1

1 ○ SHA-1: Secure Hash Algorithm – 1

- استاندارد NIST، ۱۹۹۵

- طول ورودی $2^{64} >$ بیت

- طول خروجی ۱۶۰ بیت

- استفاده شده در استاندارد امضای دیجیتال DSS

○ امنیت:

- مقاومت در برابر تصادم (قوی) تحت حمله آزمون جامع: 2^{80}

- امن محسوب میشود

- در برابر حملات شناخته شده مقاومت بالایی دارد

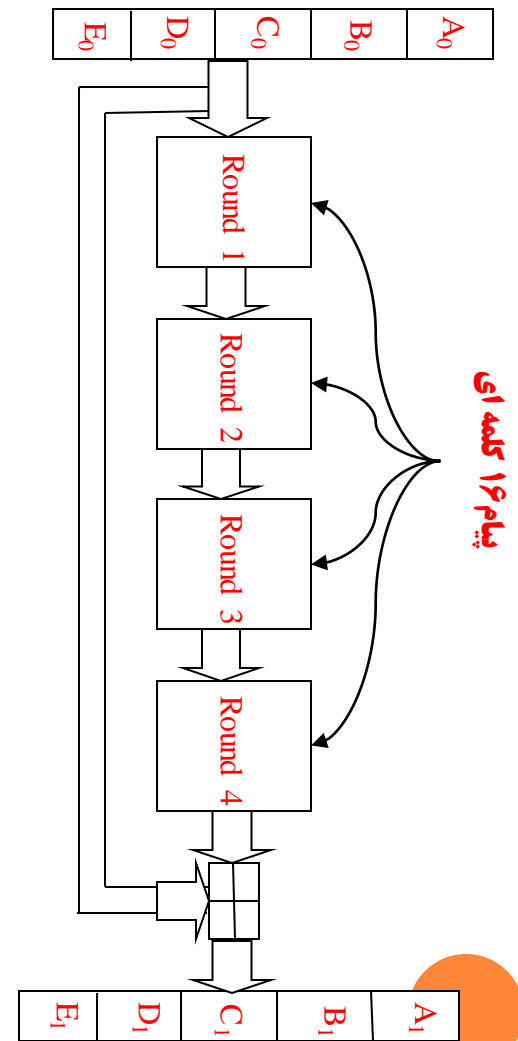
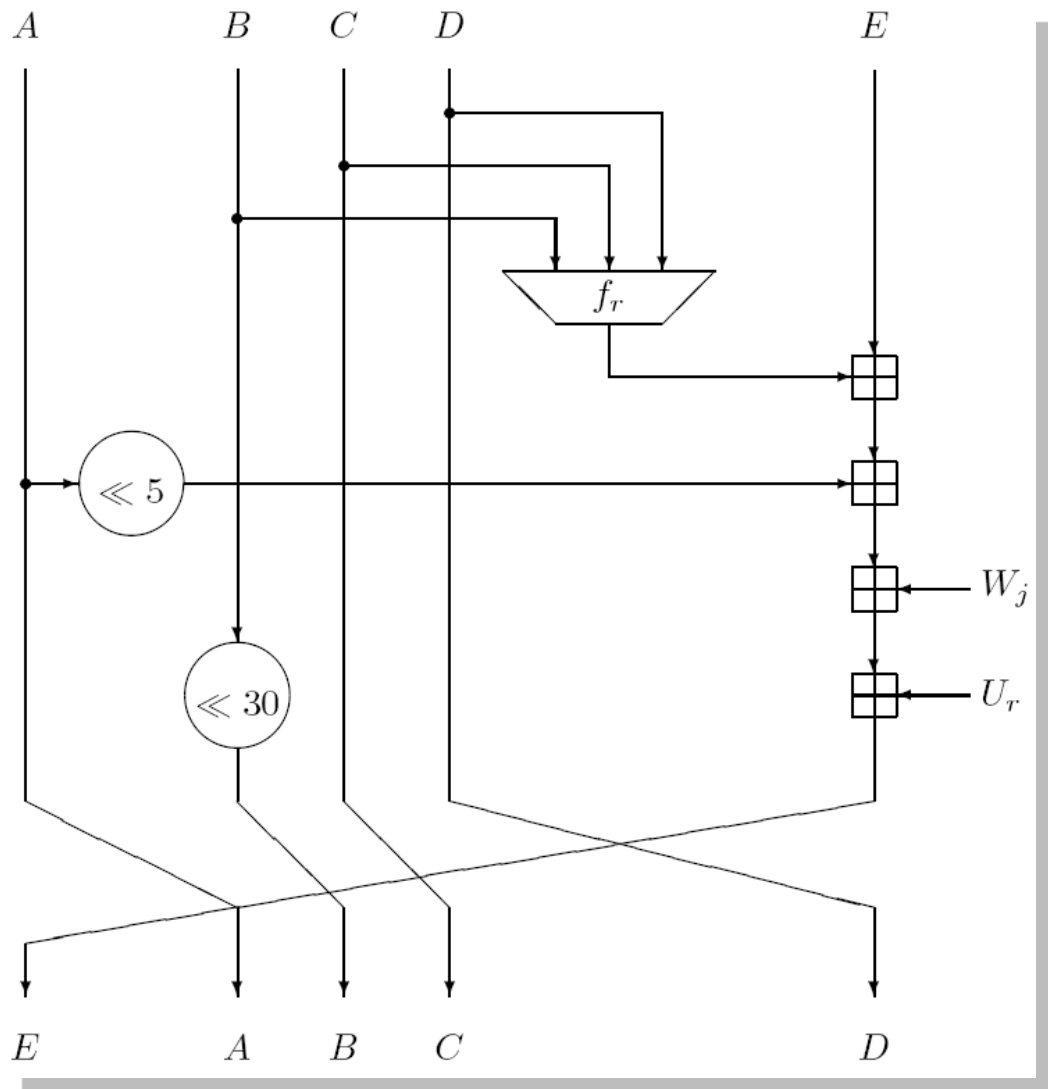


SHA-1 گونه های

- برای سازگاری با AES نسخه های زیر نیز استاندارد شده اند:
SHA-512، SHA-256 و SHA-384 •

algorithm	bit length	block size	max message	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

شمای الگوریتم SHA-1



SHA-1

○ متن با هر طولی به یک چکیده ۱۶۰ بیتی تبدیل می شود

○ **نحوه کار :**

- متن به بلوک های ۵۱۲ بیتی شکسته می شود
- بلوک آخر باید ۴۴۸ بیتی باشد تا ۶۴ بیتی که به انتهای آن افزوده می شود نشان دهنده طول واقعی متن باشد
- الگوریتم دارای ۸۰ مرحله خواهد بود که در ۴ دور ۲۰ مرحله ای تقسیم می شود
- بلوک ۵۱۲ بیتی را به ۱۶ کلمه ۳۲ بیتی می شکنیم
- برای ۶۴ کلمه باقی مانده از روش زیر استفاده می کنیم (ما برای ۸۰ مرحله به ۸۰ کلمه احتیاج داریم) :

for i := 16 to 79

$$W[i] := S^1 (W[i-3] \oplus W[i-8] \oplus W[i-14] \oplus W[i-16])$$



SHA-1

○ وجود ۵ متغیر اصلی H0 تا H4 که مقادیر اولیه آنها بشرح زیر است،

H0 = 67 45 23 01

H1 = ef cd ab 89

H2 = 98 ba dc fe

H3 = 10 32 54 76

H4 = c3 d2 e1 f0

○ وجود ۵ متغیر کمکی A، B، C، D و E

○ تابع F برای ۴ دور به شرح زیر :

Round No.	F(b, c, d)
Round 1	$(b \wedge c) \vee (\neg b \wedge d)$
Round 2	$b \oplus c \oplus d$
Round 3	$(b \wedge c) \vee (b \wedge d) \vee (c \wedge d)$
Round 4	$b \oplus c \oplus d$



SHA-1

○ حلقه تکرار

```
for i := 0 to 79 do {  
    Temp := S5(A) + F(B, C, D) + E + W[i] + K[i];  
    E := D;  
    D := C;  
    C := S30(B);  
    B := A;  
    A := Temp;  
}
```



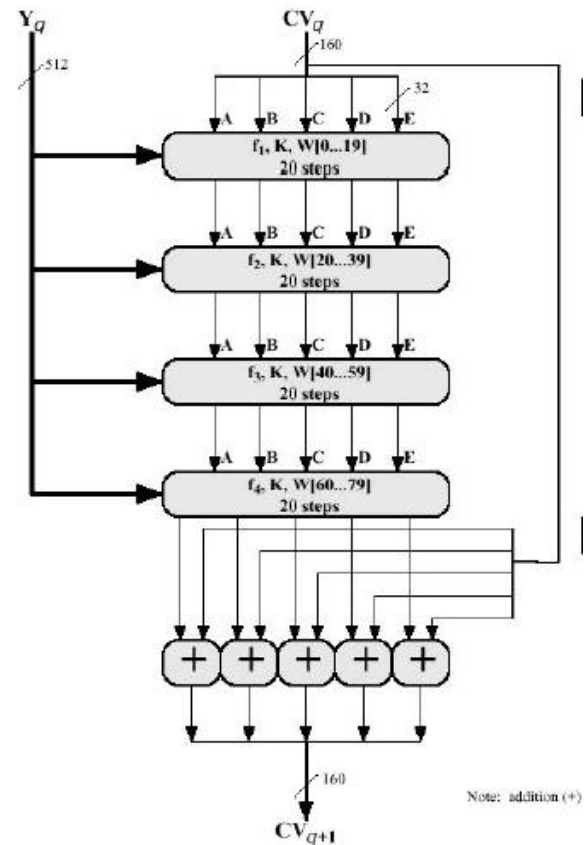
SHA-1 LOGIC

- Step 1. Appending padding bits
- Step 2. Append length
- Step 3. Initialize MD buffer
 - Five 32-bit registers, $E = C3D2E1F0$
 - Big-endian

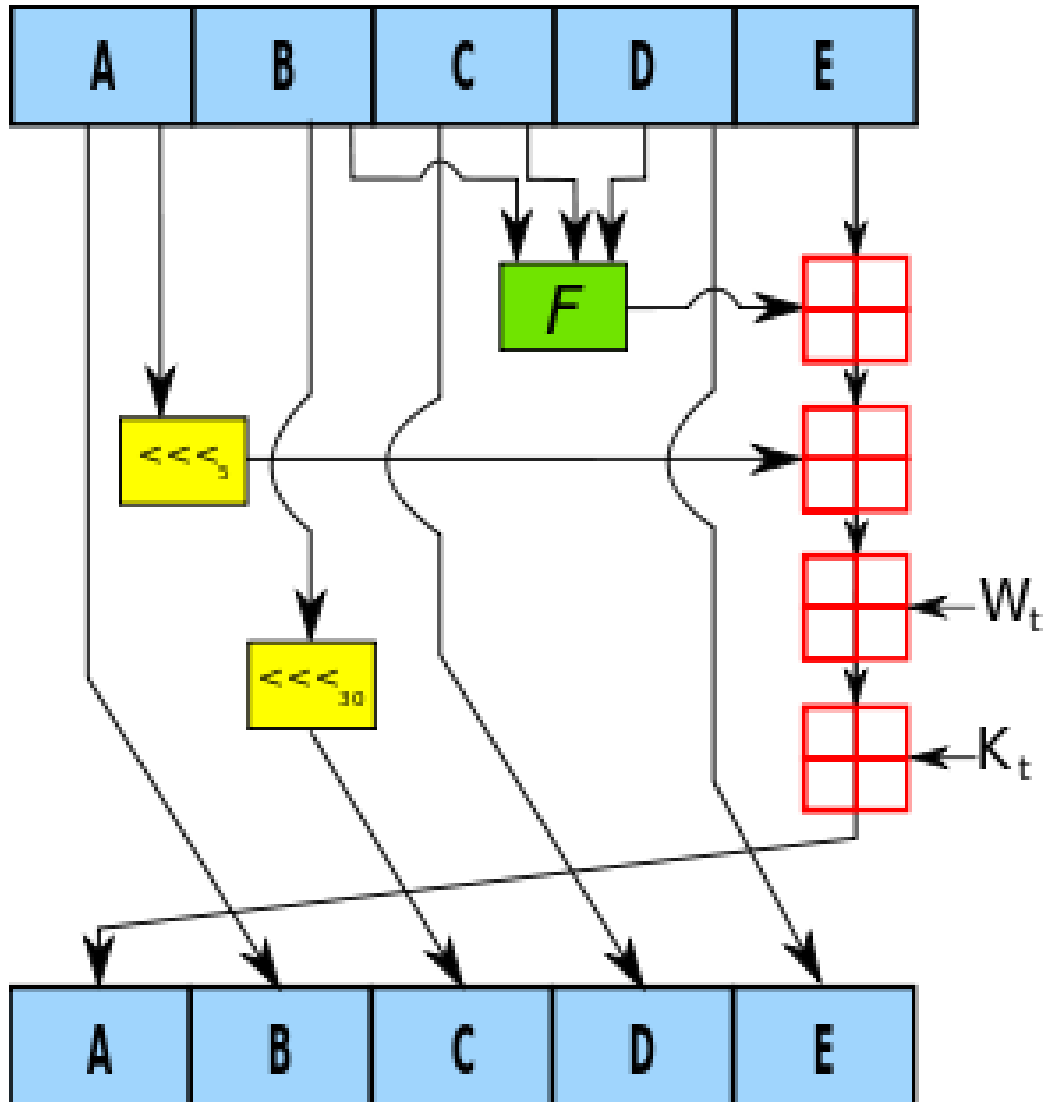


SHA-1 LOGIC

- Step 4. Process message in 512-bit blocks
 - Four rounds of 20 steps
- Step 5. Output



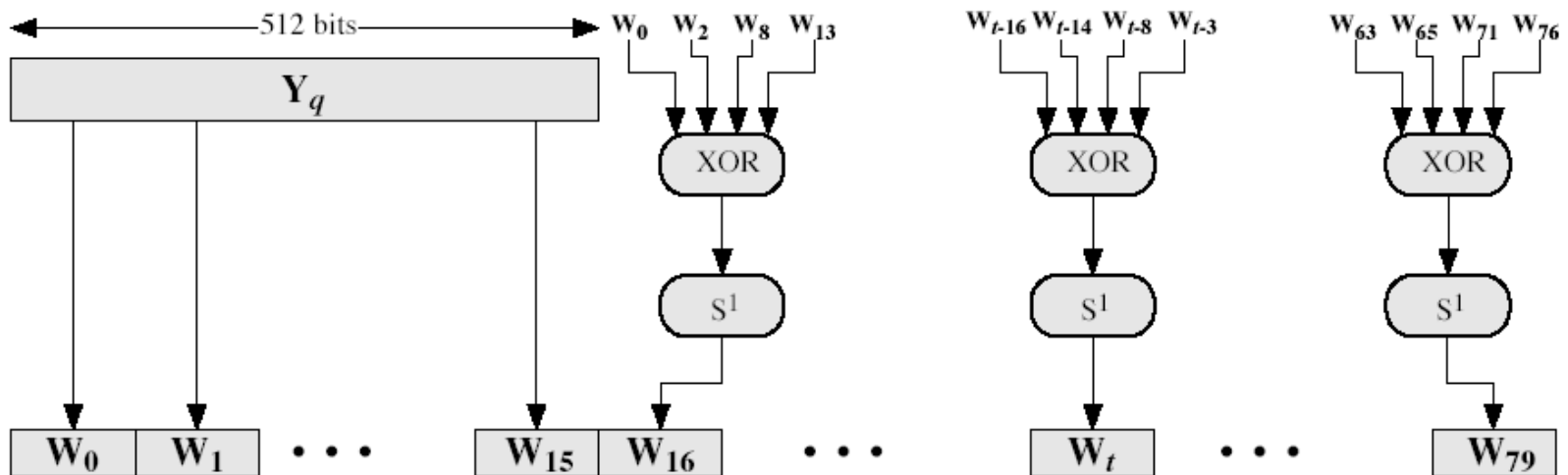
SHA-1 COMPRESSION FUNCTION



K_t is a additive constant
varying across rounds



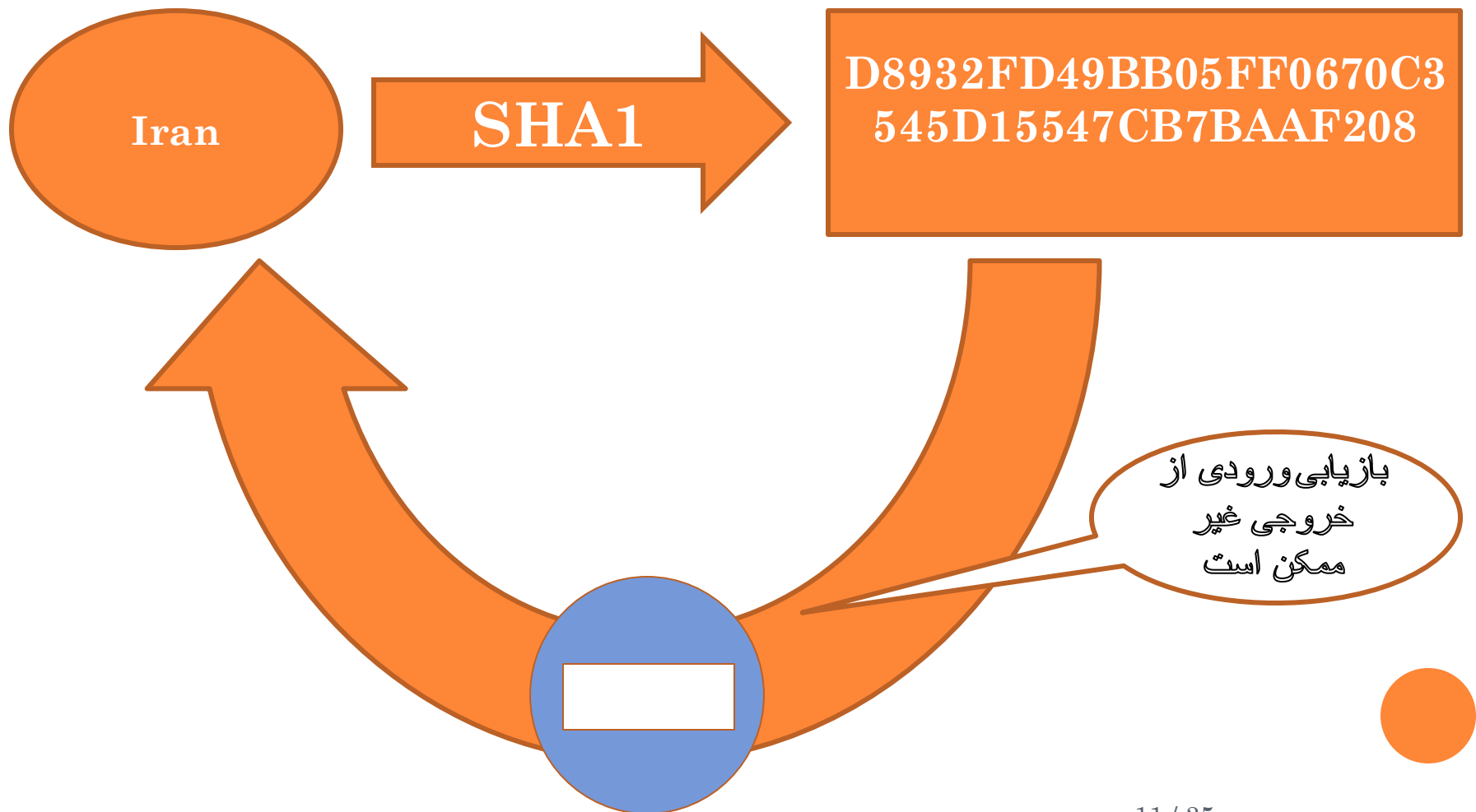
SHA-1 COMPRESSION FUNCTION



$$W_t = S^1(W_{t-16} + W_{t-14} + W_{t-8} + W_{t-3})$$



الگوریتم SHA1

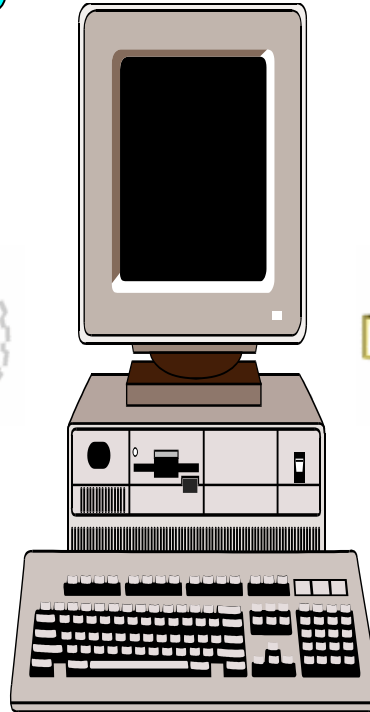


تابع درهم ساز (هشینگ) چیست ؟

تابع درهم ساز

پیام طولانی

Hash functions, most notably MD5 and SHA-1, initially crafted for use in a handful of cryptographic schemes with specific security requirements, have become standard fare for many developers and protocol designers who treat them as black boxes with magic properties. This practice had not been seriously challenged until 2004, since both functions appeared to have withstood the test of time and intense scrutiny of cryptanalysts. Starting last year, we have seen an explosive growth in the number and power of attacks on the standard hash functions. In this note we discuss the extent to which the hash functions can be thought of as black boxes, review some recent attacks, and, most importantly, revisit common applications of hash functions in programming practice.



چکیده پیام

b34d0fcefef36b3ff420b

$$H: \{0,1\}^* \longrightarrow \{0,1\}^n$$

تابع درهم ساز

لمول ورودی: دلخواه



لمول خروجی: مقدار ثابت



ساده بودن محاسبات



۱- طرفه بودن



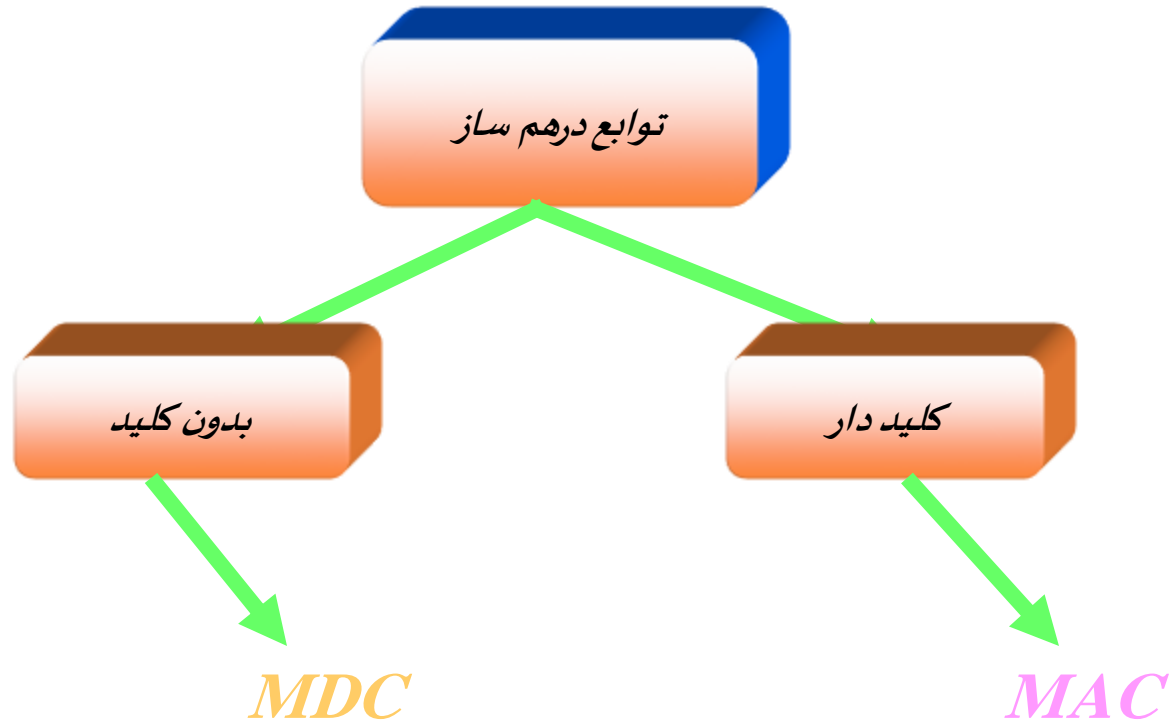
تابع درهم ساز = توابع بدون بازگشت

تابع درهم ساز

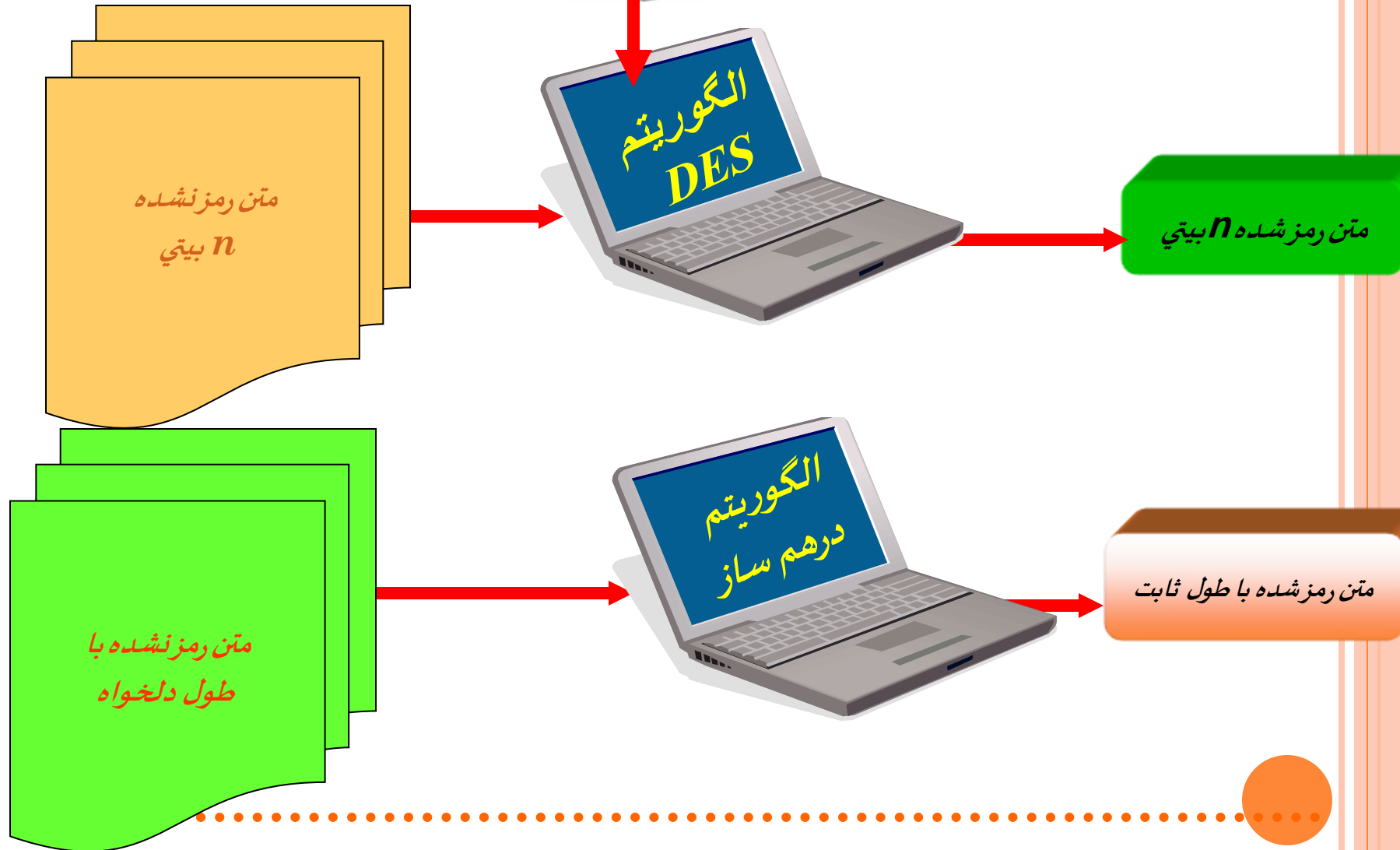
فایل یا پیام



طبقه بندی عمومی توابع درهم ساز

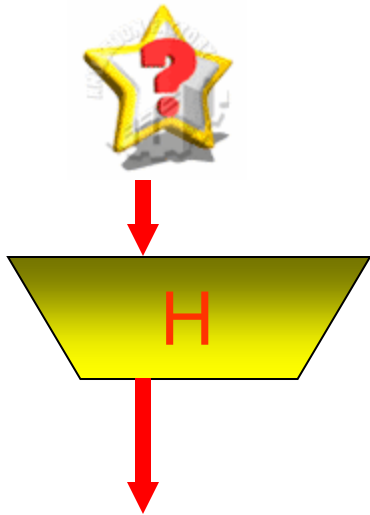


تفاوت توابع درهم ساز و گوریتم DES



ویژگیهای تابع درهم ساز

مقاومت در مقابل پیش تصویر



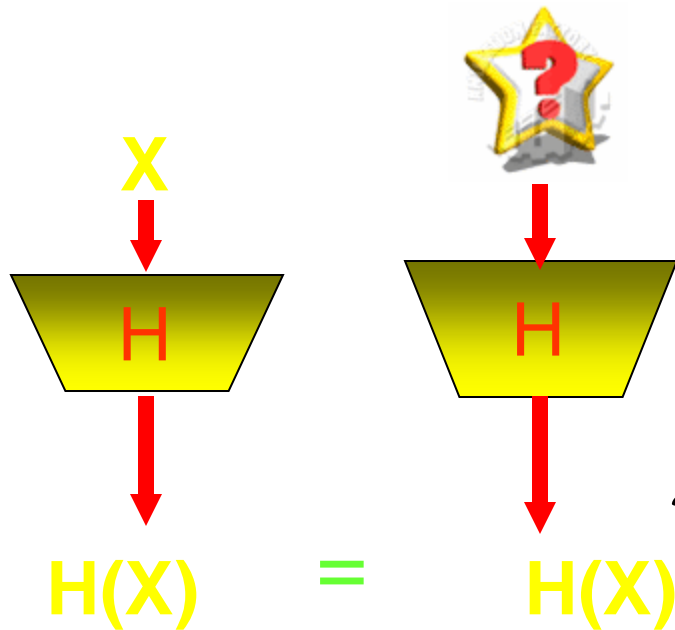
$H(x)$

یعنی نتوانیم از روی چکیده متن
اصلی را پیدا کنیم

```
1-choose  $X_0 \subseteq X$  ,  $|X_0|=q$   
2-for all  $x \in X_0$  do  
3- if  $h(x)=y$  then return  $x$   
4-end for  
5-return failure
```

ویژگی‌های تابع درهم ساز

مقاومت در برابر پیش تصویر دوم



این مورد هم مثل مسئله قبلی یعنی اینکه به ازای یک متن مشخص مثل X نتوانیم یک متن دیگر پیدا کنیم که دارای چکیده یکسان باشند.

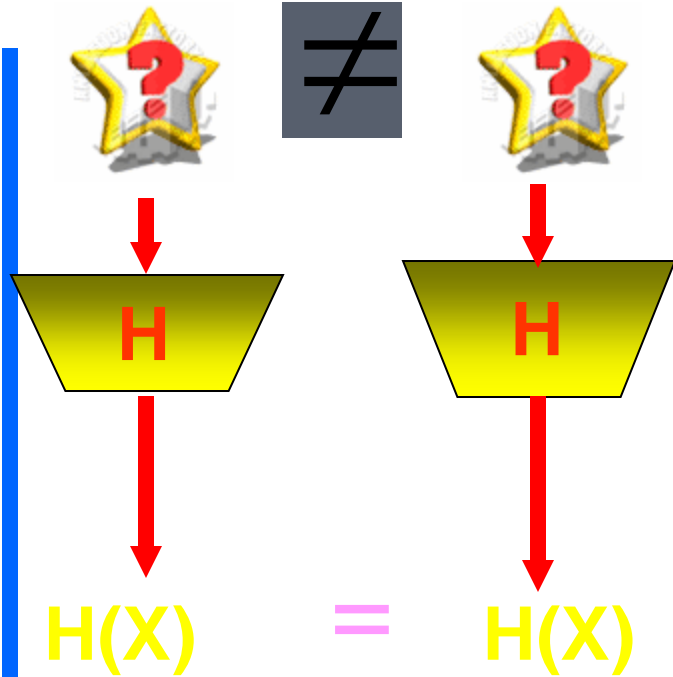
```
1-  $y \leftarrow h(x)$   
2- choose  $X_0 \subseteq X / \{x\}$ ,  $|X_0| = q-1$   
3- for all  $x_0 \in X_0$  do  
4-   if  $h(x_0) = y$  then return  $x_0$   
5- end for  
6- return failure
```

ویژگیهای تابع درهم ساز

مقاومت در برابر برخورد (عدم برخورد)

یعنی اینکه به طور کلی نتوانیم دو متن متمایز پیدا کنیم که دارای چکیده یکسان باشند.

این ویژگی به جعل ناپذیری امضاء الکترونیکی کمک می کند.



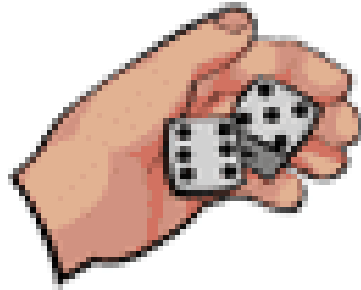
1-choose $X_0 \subseteq X$, $|X_0|=q$

2-for all $x \in X_0$ do $y_x \leftarrow h(x)$

3-if $y_x = y_{x'}$, for some $x' \neq x$ then return (x, x')

4-else return failure

ویژگیهای تابع درهم ساز



ویژگی اوراکل تصادفی

47

Flash

تابع $h()$ بعنوان تابعی که به صورت تصادفی رفتار می کند ، انتخاب می گردد.

هر حمله ای ممکن است ویژگی اوراکل تصادفی را غیر معتبر می سازد.

کاربردهای توابع درهم سازی

امضای یک فایل یا چکیده فایل

کد احراز اصالت پیام

توابع شبه تصادفی (رمزهای دنباله ای)

رمزهای قطعه ای

پروتکل های تصدیق هویت براساس رمز متقارن

که این مورد اخیر بنام **MAC** مشهور است

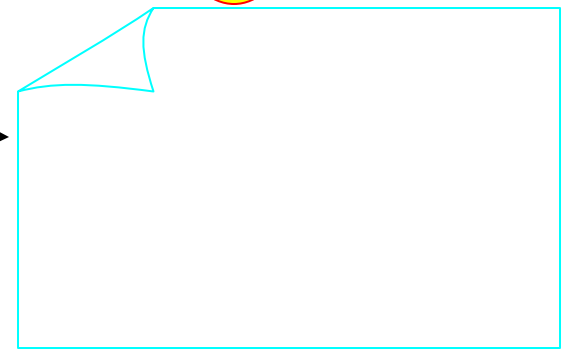
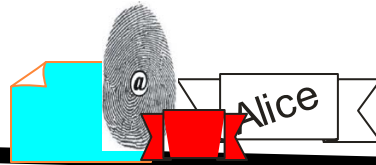
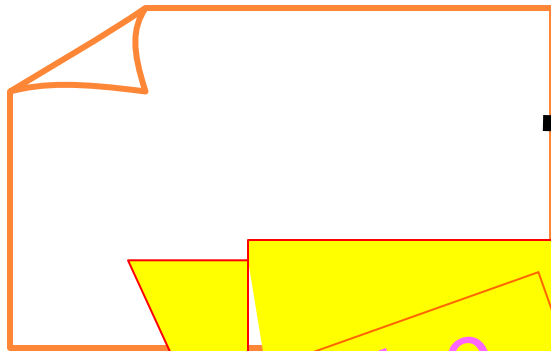
کاربرد در طرح های امضای دیجیتال



آلیس



باب



h



?



آیا امضای دیجیتال صحیح

است؟



آلیس



باب

امضاء صحیح
است!



Alice

h



h

h



تصادم!



?



Alice

h



Alice

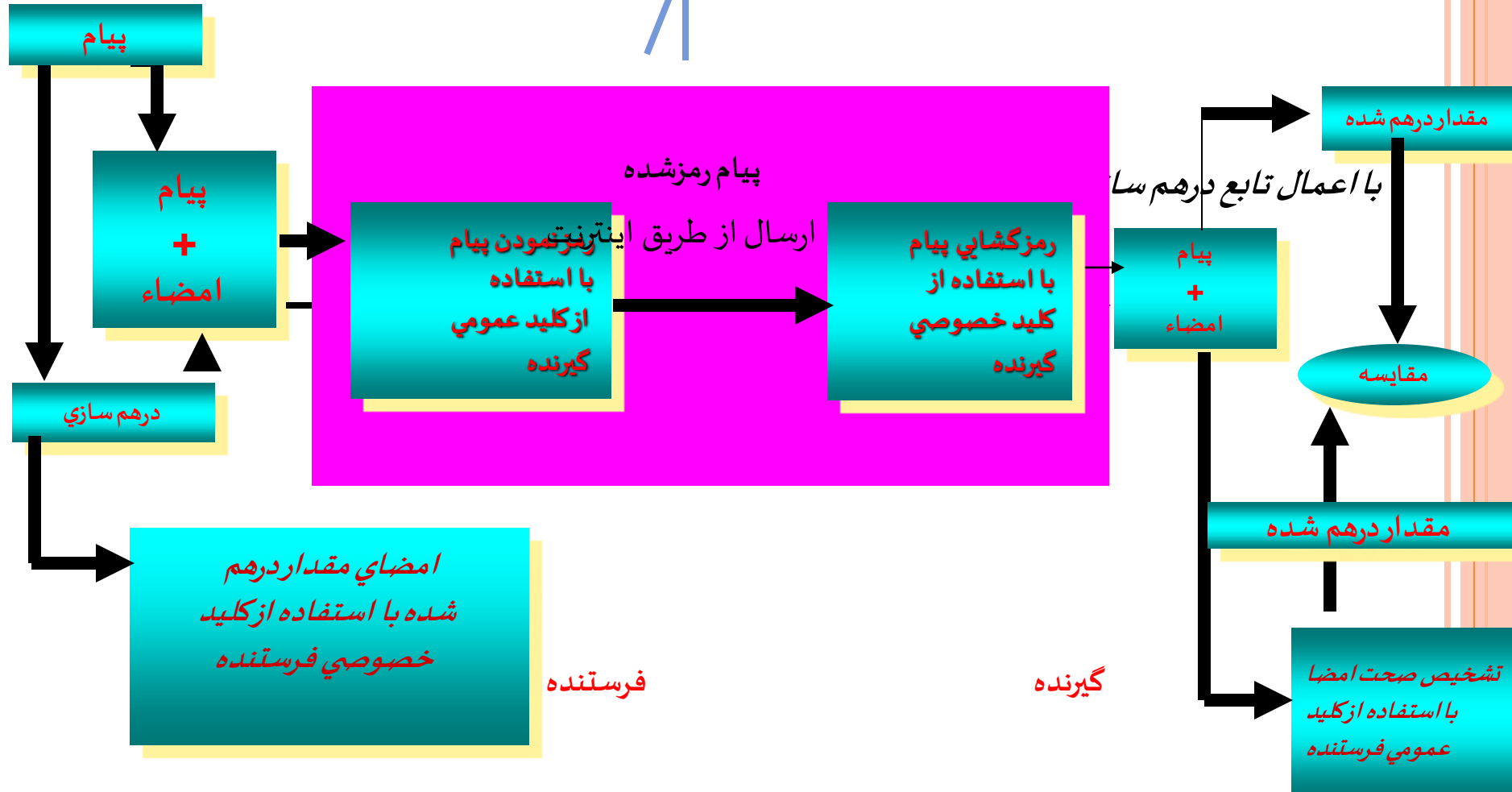
آلیس لطفا این پیام را
امضاء کن

باب، این پیام را آلیس
امضاء کرده است.

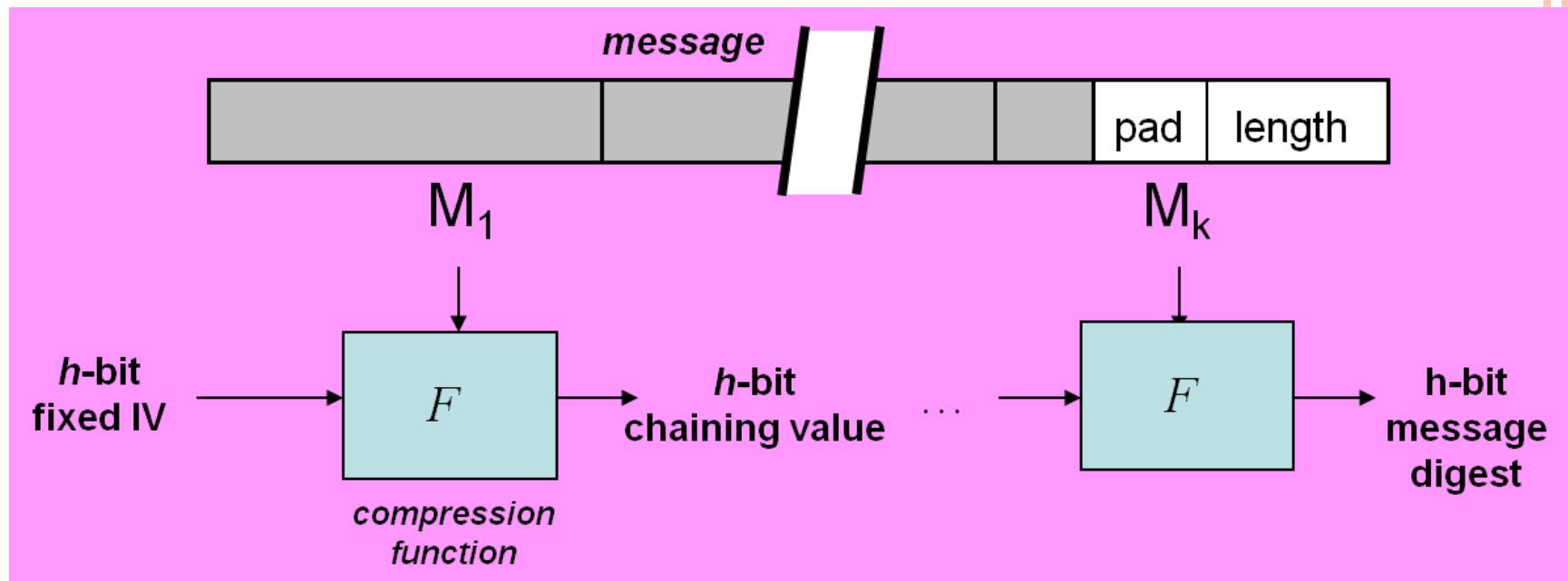


Eve

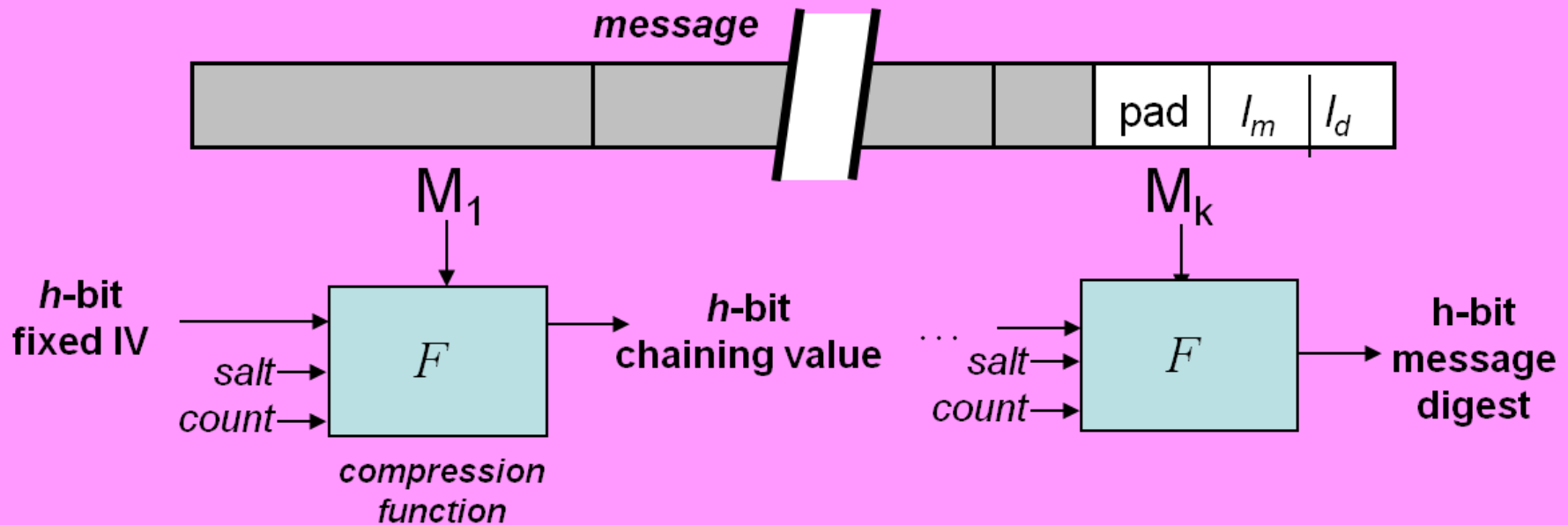
ایجاد یک امضای دیجیتالی



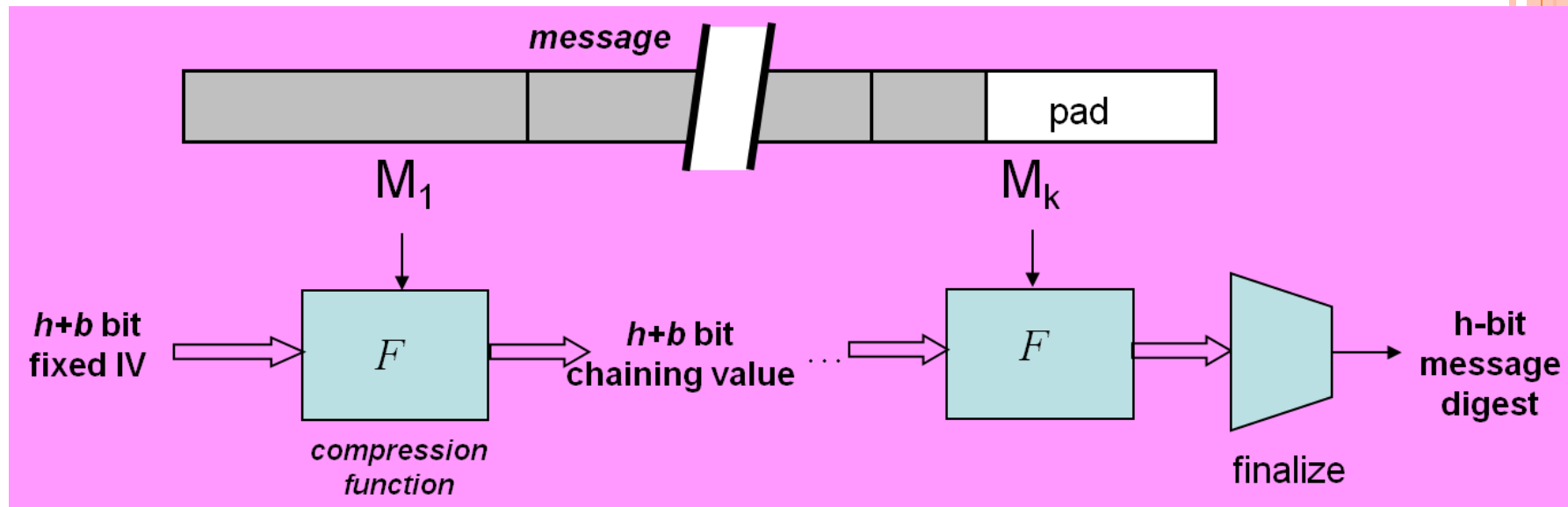
توابع درهم ساز تکراری (مرکل-دمگارد)



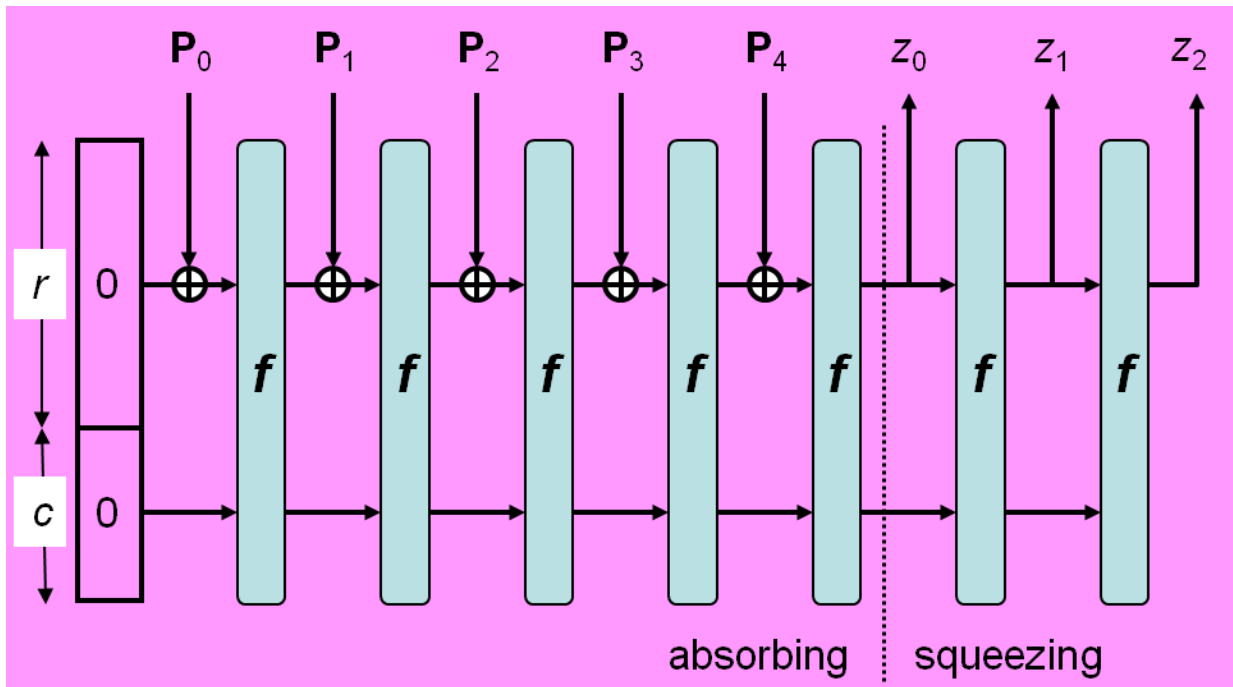
توابع درهم ساز تکراری (HAFA)



توابع درهم ساز تکراری (*Wide Pipe MD*)



توابع درهم ساز تکراری (Sponge)



در اینجا تابع f یک جایگشت است

حمله گسترش طول

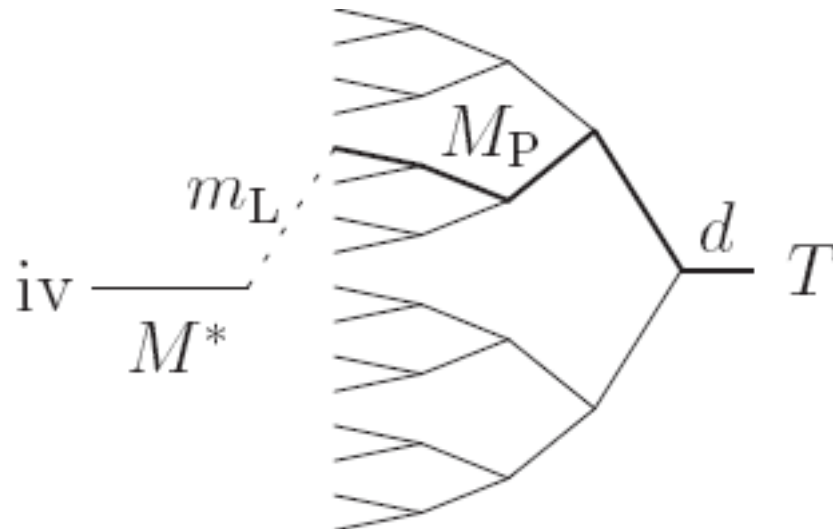
- ضعف گسترش طول در ساختار مرکب - دماغارد:

- اگر بتوانیم دو پیام متفاوت m و m' بیابیم که $f(IV, m') = f(IV, m)$ در این صورت می توان بینهایت زوج پیام متفاوت M و M' ساخت که $H(M) = H(M')$ در اینجا مشاهده می شود که با داشتن يك تلاقي می توان به تعداد دلخواه تلاقي رسید.
- این موجب ساخت یک سری پیامهای متفاوت با چکیده یکسان خواهد بود.
- با داشتن $H(M)$ و طول M و بدون داشتن M مقدار $H(M || m')$ برای هر m' دلخواه را می توان حساب کرد.
- که این مورد اخیر منجر به تزریق در پیام میشود که بنوبه خود منجر به ساخت و تزریق بدافزارهایی به نرم افزارهای مجاز میشود که امضایی اصیل دارند.



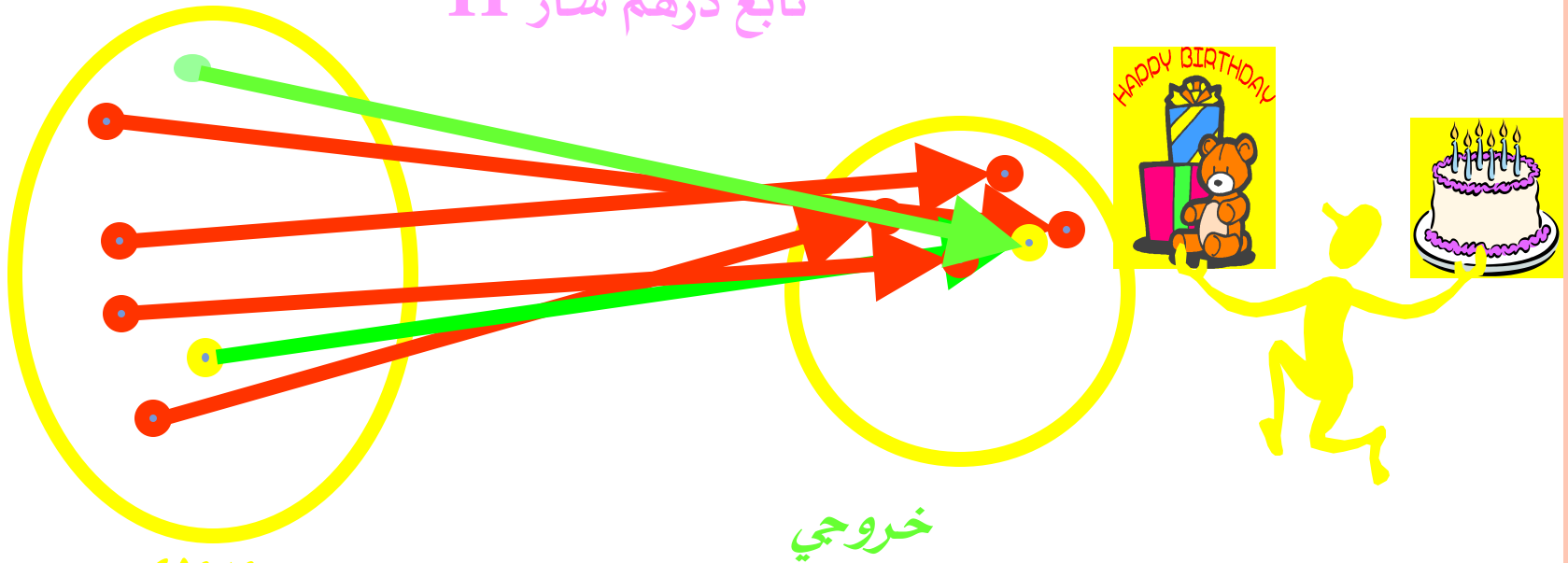
حمله غیب گو

- حمله کننده مدعي مي شود که مي تواند نتیجه يك اتفاقي که قرار است در آینده رخ دهد را پيشگويي کند. او اين کار را با منتشر کردن يك مقدار درهم سازي T قبل از رخ دادن اتفاق انجام مي دهد. بعد از اينکه اتفاق رخ داد، او يك پيام M ، که حاوي اطلاعات کافي براي اثبات اينکه او از نتیجه آگاه بوده است، و نتیجه درهم سازي متناظر با آن، به گونه اي که $H(M)=T$ ، را ارائه مي کند.



طول خروجي (حمله روز تولد)

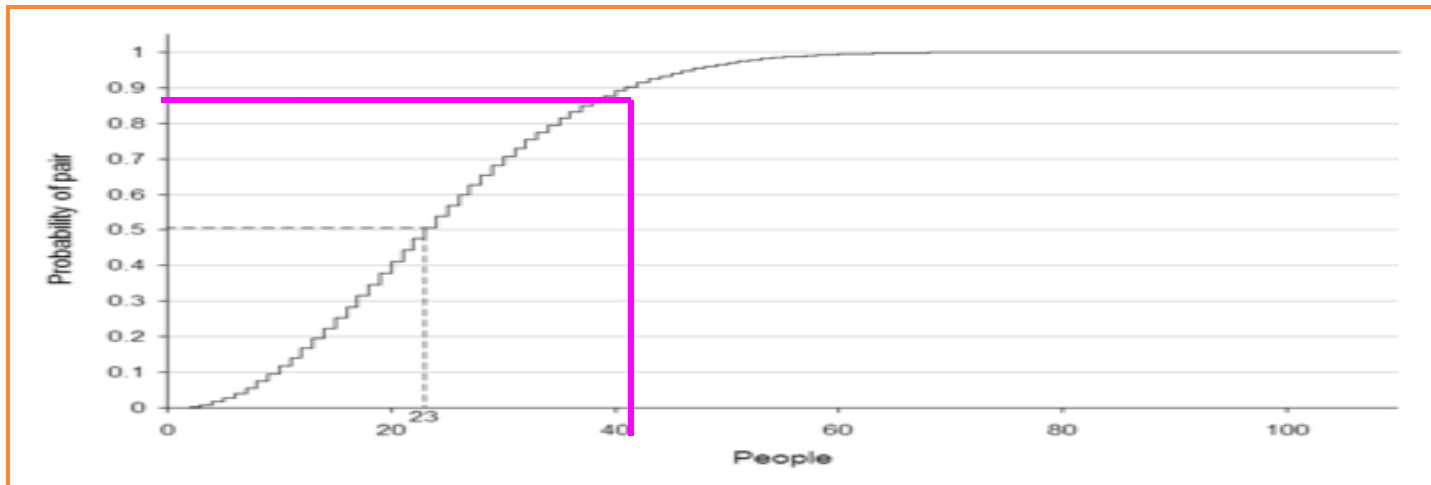
تابع درهم ساز H



ورودي

خروجي

براي مثال درميان 40 نفر، احتمال اينکه 2 نفر روز تولد يکسان داشته باشند برابر است با 89 درصد.



طول خروجي مورد نياز



Flash

دريك تابع درهم ساز n بيتي امن بايد:

- 1- توليد يك پيش تصوير يا پيش تصوير دوم به حدود 2^n عمليات نياز داشته باشد.
- 2- توليد يك برخورد به حدود $2^{n/2}$ عمليات نياز داشته باشد.

$$n/2 = 80 \longrightarrow n = 160$$



يك تابع درهم ساز مقاوم در برابر برخورد امروزه، حداقل 160 بيتي مي باشد.

BIRTHDAY ATTACK

○ سوال : در یک مهمانی باید چند نفر حضور داشته باشند، که حداقل دو نفر با احتمال بالاتر از ۵۰٪ در یک روز به دنیا آمده باشند؟

○ جواب : ۲۳ نفر!

○ طرح مسئله

○ در یک گروه n نفره احتمال آنکه هر n نفر روز تولد مشترک نداشته باشند برابر است با :

$$P_n = \left(1 - \frac{1}{365}\right) \cdot \left(1 - \frac{2}{365}\right) \cdot \left(1 - \frac{3}{365}\right) \cdots \left(1 - \frac{n-1}{365}\right)$$
$$= \frac{364}{365} \cdot \frac{363}{365} \cdot \frac{362}{365} \cdots \frac{364-n}{365}$$

○ اگر بخواهیم حداقل دو نفر در یک روز به دنیا آمده باشند $\leftarrow 1 - P_n$

○ $1 - P_n$ برای $n = 23$ برابر ۵۰٪ است



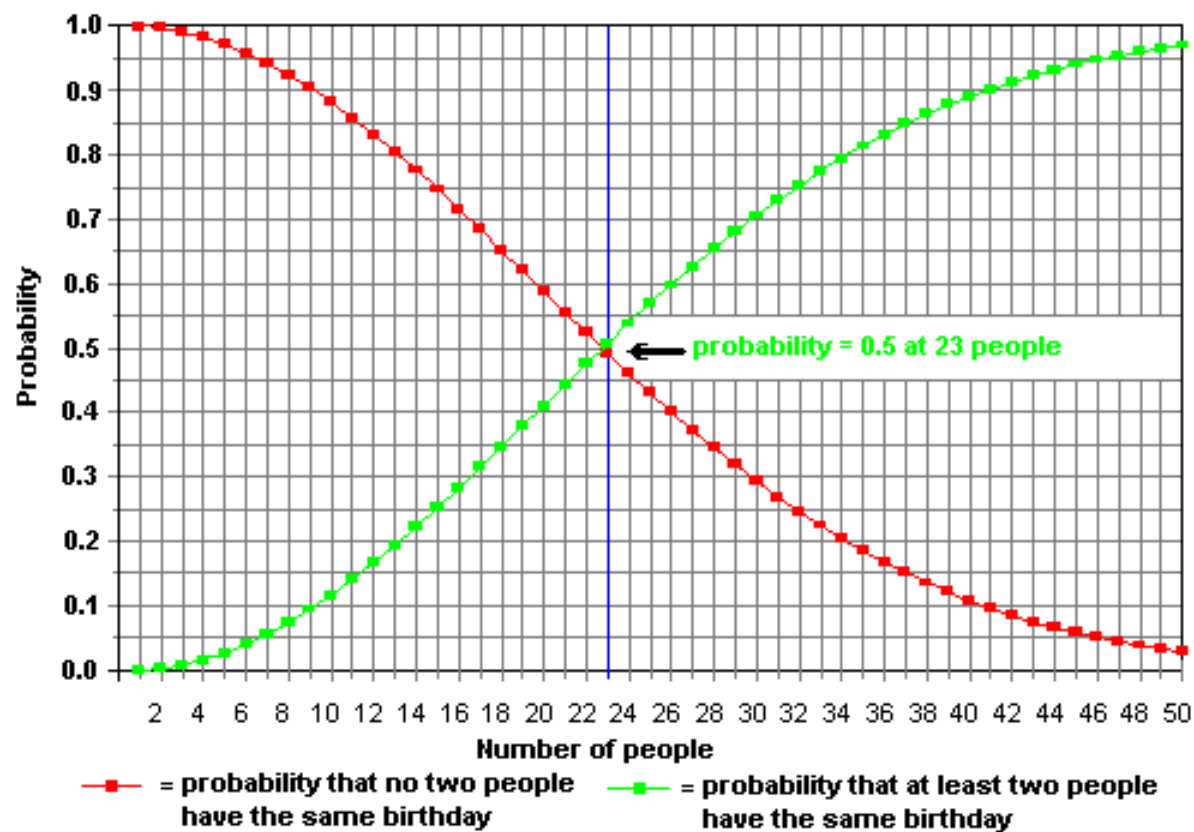
پارادکس روز تولد

○ مبنای ریاضی

- تابع H با 2^m خروجی ممکن را در نظر بگیرید (خروجی m بیتی) H را به k ورودی تصادفی اعمال کنیم و خروجی را مجموعه X در نظر می گیریم. به همین ترتیب مجموعه Y را تشکیل می دهیم. اگر k بزرگتر از باشد، احتمال حداقل یک تصادم در بین $2^{m/2}$ تصای دو مجموعه X و Y بیش از $1/2$ می باشد



پارادکس روز تولد



BIRTHDAY ATTACK

○ حالت کلی تر

○ به جای ۳۶۵ روز سال، N روز داشته باشیم و n نفری که در جشن حاضر دارند روز تولد مشترکی نداشته باشند

$$P_n = \left(1 - \frac{1}{N}\right) \cdot \left(1 - \frac{2}{N}\right) \cdot \left(1 - \frac{3}{N}\right) \dots \left(1 - \frac{n-1}{N}\right)$$

$$\ln(P_n) = \ln\left(1 - \frac{1}{N}\right) + \ln\left(1 - \frac{2}{N}\right) + \ln\left(1 - \frac{3}{N}\right) + \dots + \ln\left(1 - \frac{n-1}{N}\right)$$

$$\ln(P_n) = -\left(\frac{1}{N} + \frac{2}{N} + \dots + \frac{n-1}{N}\right) - \frac{1}{2}\left(\frac{1}{N^2} + \frac{4}{N^2} + \dots + \frac{(n-1)^2}{N^2}\right) \dots$$

$$\ln(P_n) \cong -\frac{n(n-1)}{2N}$$

○ با فرض $P_n = \frac{1}{2}$ یعنی $\Pr(h(M)=h(M')) \geq 0.5$ و تقریب $n-1 \cong n$:

$$n = \sqrt{2 \ln 2} \sqrt{N}$$

$$N = 365 \rightarrow n = 22.5$$

○ و در حالت کلی داریم

$$n \cong \sqrt{2 \ln\left(\frac{1}{1-P_n}\right)} \sqrt{N}$$



BIRTHDAY ATTACK

○ برای الگوریتم SHA-1، اگر چکیده k بیتی باشد 2^k حالت مختلف خواهیم داشت یعنی $N = 2^k$

○ احتمال بالای ۵۰٪ برای اینکه دو چکیده یکسان داشته باشیم برابر است با

$$n \cong \sqrt{2 \ln 2} * \sqrt{N} \cong 2^{k/2}$$

○ پس در SHA-1، که چکیده ۱۶۰ بیت دارد

$$k = 160, \quad P_n = \frac{1}{2}, \quad N = 2^{80}$$

○ یعنی باید $n=2^{80}$ پیام تولید شود تا با احتمال بالاتر از ۵۰٪ دو پیام با چکیده های یکسان حاصل شود

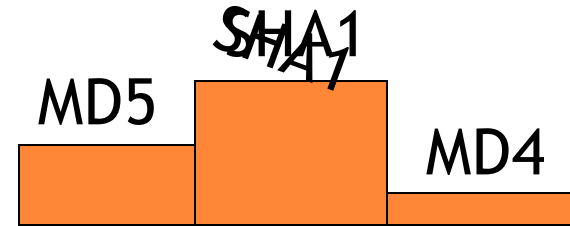


گذشته، حال، آینده

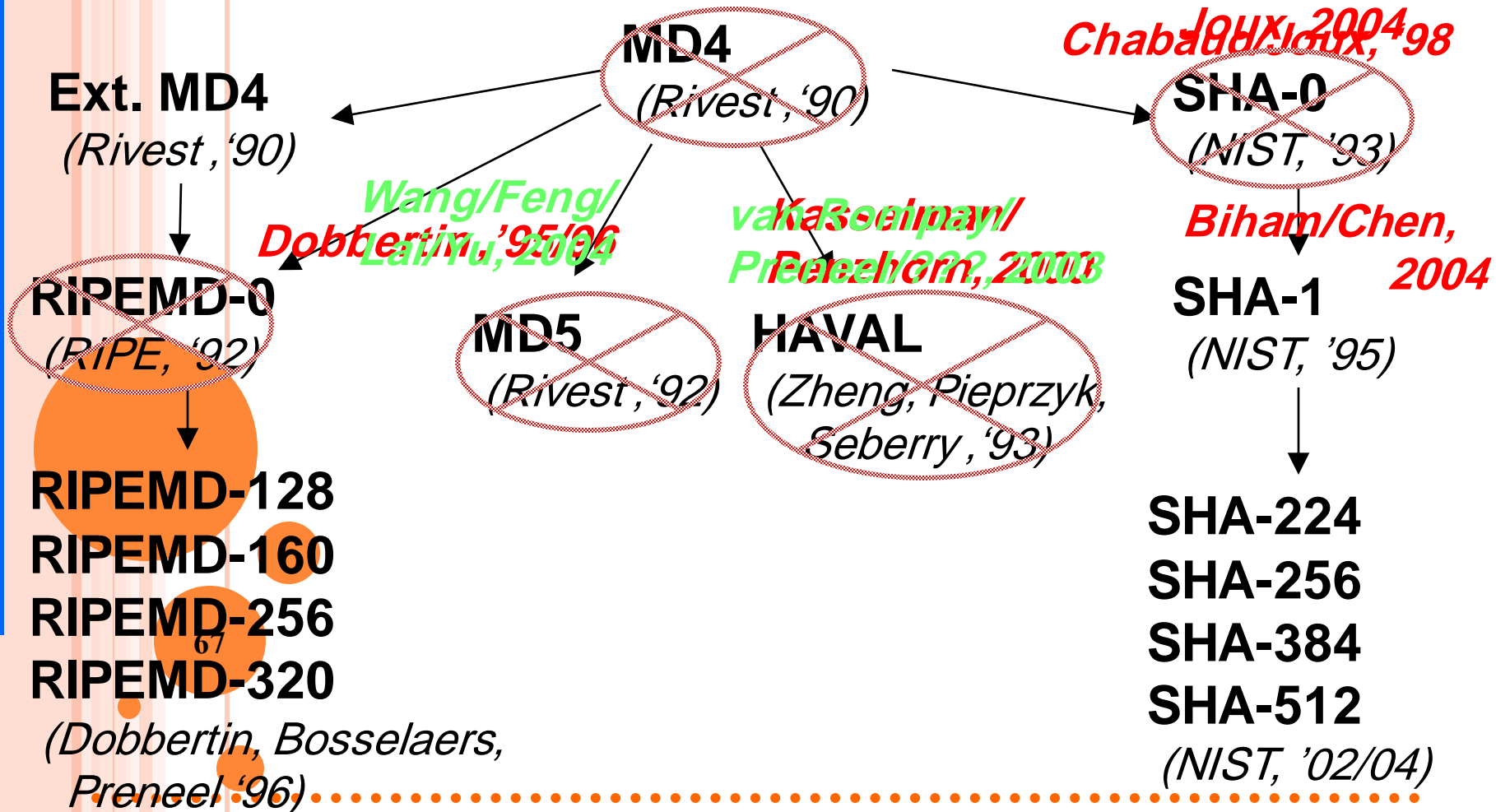


گذشته

1989	MD2
1990	MD4
1991	MD5
1992	HAVAL
1993	SHA0, RIPEMD
1994	
1995	SHA1
1996	MD4 شکسته شد
1997	حمله تنویری روی SHA-0
1998	
1999	
2000	
2001	
2002	SHA-256, 384, 512
2003	
2004	SHA-224
2005	MD5 و SHA0 شکسته شدند و حمله تنویری روی SHA1
2006	



گذشته



اکنون : مسابقه الگوریتم استاندارد جدید

• در سال 2007 موسسه NIST فراخوان طرح جدید را در قالب مسابقه طراحی الگوریتم درهم ساز پیشرفته با نام SHA-3 ارائه کرد.

- 64 الگوریتم ثبت نام کردند.
- 51 الگوریتم برای دور اول پذیرفته شدند.
- 14 الگوریتم به دور دوم راه پیدا کردند.
- 5 الگوریتم به عنوان کاندیداهای نهایی معرفی شدند.
- .؟؟؟؟



روند انتخاب استاندارد جدید

- ✓ 01/23/07 Draft submission criteria published
- ✓ 11/02/07 Federal Register announcement of SHA-3 Competition
- ✓ 08/31/08 Preliminary submissions due
- ✓ 10/31/08 Submissions due – 64 received
- ✓ 12/09/08 Announced 51 First round candidates
- ✓ 02/25/09 First SHA-3 Candidate Conference, Leuven Belgium
- ✓ 07/24/09 Announced 14 second round candidates
- ✓ 09/15/09 Tweaks accepted, second round began
- ✓ 08/23/10 – 08/24/10 Second SHA-3 Candidate Conference, UCSB
- ✓ 4Q10 Announce finalist candidates
- ✓ 1Q11 Final tweaks of candidates
- 1Q12 Last SHA-3 Candidate Conference
- 2Q12 Announce winner
- 4Q12 FIPS package to Secretary of Commerce



کاندیداهاي نهاي استاندارد جديد درهم سازي

- **BLAKE**
 - ❖ Swiss, HAFA
- **Grøstl**
 - ❖ European, WideP MD
- **JH**
 - ❖ Singapore, novel construction
- **Keccak**
 - ❖ European, SpongeSkein
- **SKEIN**
 - ❖ US, WideP MD (more or less)



HMAC

- HMAC یک الگوریتم احراز هویت پیام است
- HMAC اساساً روشی برای ترکیب کردن کلید مخفی با الگوریتمهای درهم ساز فعلی میباشد.
- برای تولید چکیده پیغام، از توابع درهم استفاده شده است
 - در مقابل استفاده از رمزهای قطعه ای
 - بدلیل مزایای عملی توابع درهم ساز
- HMAC جزو ملزومات پیاده سازی امنیت IP میباشد.
- HMAC به طور گسترده استفاده میشود (مثلاً SSL)



HMAC: اهداف طراحی

- استفاده از توابع درهم ساز بدون تغییر آنها
- پشتیبانی از توابع درهم ساز متنوع
- حفظ کارایی و سرعت تابع درهم ساز به کار گرفته شده
- استفاده ساده از کلید
- طراحی روشن و بدون ابهام
- طول ثابت
- علاوه بر امکان اصالت و صحت پیام امکان احراز هویت را نیز در خود دارد. حتی امکان انکار ناپذیری را نیز تا حدی فراهم میکند.
- توجه کنید که چکیده سازی فقط اصالت پیام را فراهم میکند.
- HMAC بطور گسترده در اجرای امضای دیجیتال بکار میرود



HMAC: الگوریتم

H: تابع درهم ساز به کار گرفته شده

- M : پیام ورودی

- K : کلید مخفی

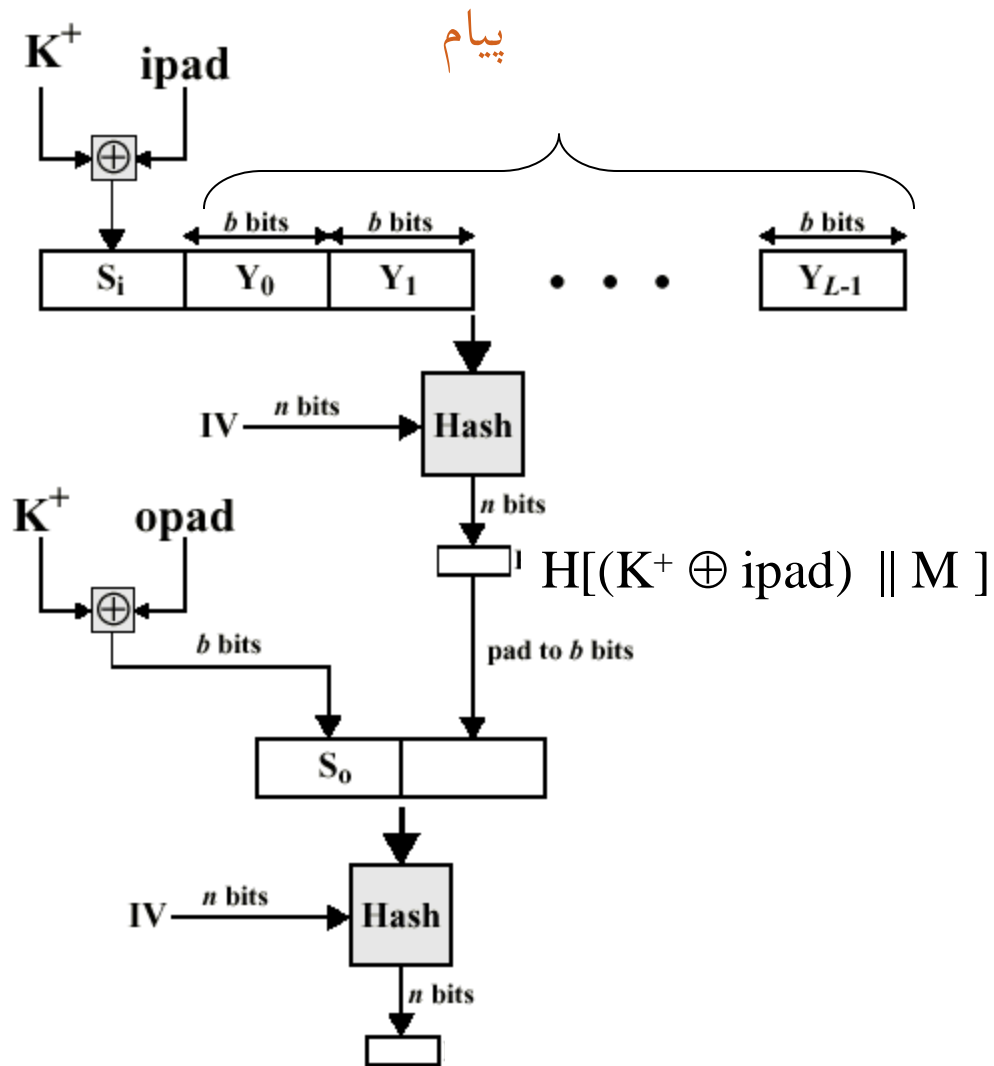
- K^+ : کلید مخفی که یک دنباله صفر به آن اضافه شده است

- $ipad$: تکرار رشته ۰۰۱۱۰۱۱۰ بصورت هگز میشود 36

- $opad$: تکرار رشته ۰۱۰۱۱۰۱۰ بصورت هگز میشود 5c

$$HMAC_K = H[(K^+ \oplus opad) || H[(K^+ \oplus ipad) || M]]$$





$$H[(K^+ \oplus \text{opad}) \parallel H[(K^+ \oplus \text{ipad}) \parallel M]]$$



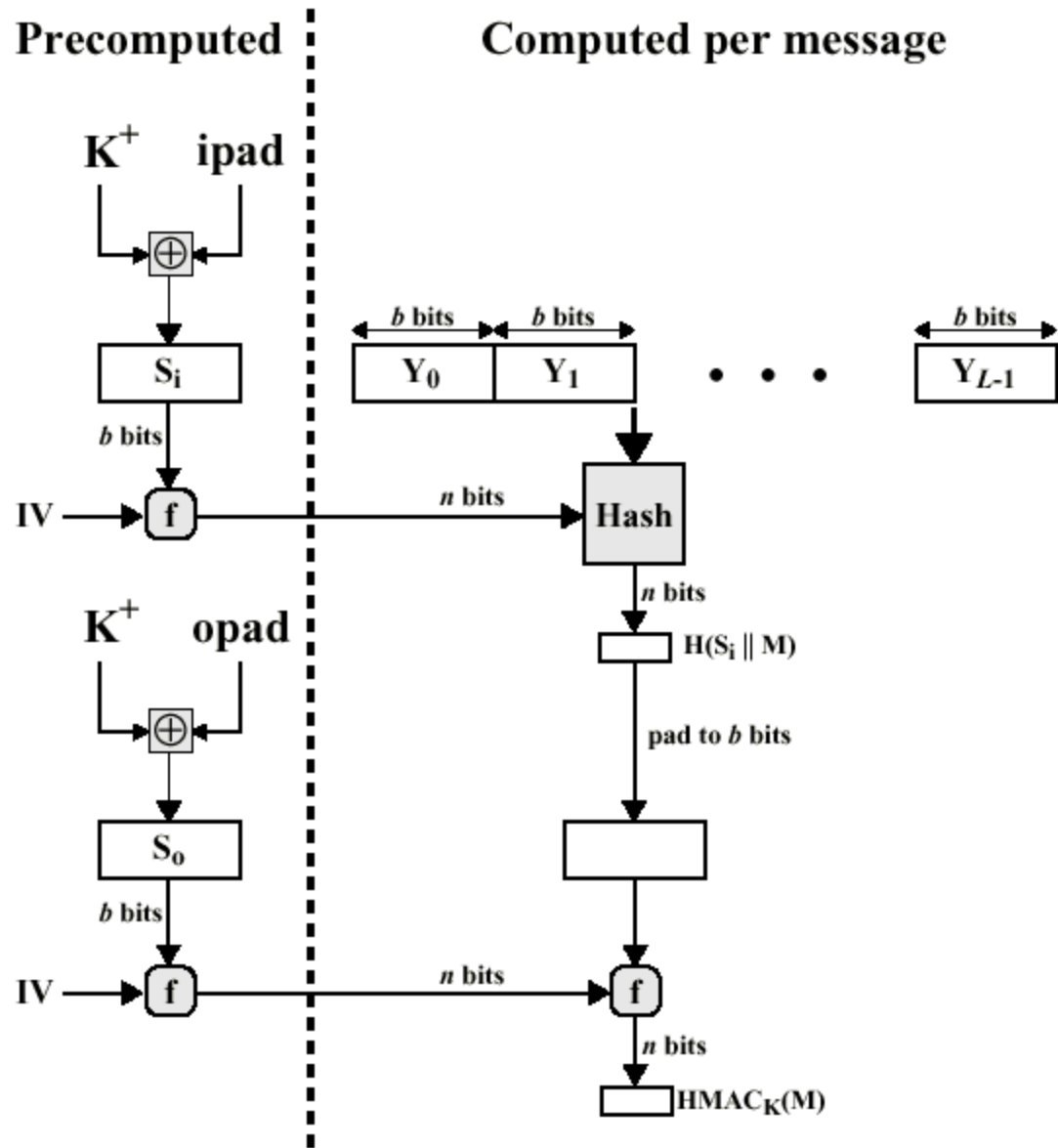


Figure 9.11 Efficient Implementation of HMAC

HMAC: امنیت

- ارتباط دقیق بین امنیت تابع در هم ساز با امنیت HMAC اثبات شده است.
- مقاومت HMAC در برابر حمله روز تولد از تابع در هم ساز به کار گرفته شده، بیشتر است.
- استفاده از MD5 در هنگام نیاز به سرعت بیشتر مجاز است.



MAC

- کدهای MAC مبتنی بر تولید چکیده هستند
- با توافق طرفین بر سر یک کلید سری
- کلید سری فقط برای تولید کد MAC است و برای رمزنگاری پیام نمی باشد
- کدهای MAC فقط برای بررسی صحت پیام است
- این نوع کدها بسیار سریع، کوتاه و با طول ثابت اند و روی بیشتر کارت های شبکه موجود می باشد
- چون کلید متقارن بین طرفین مورد توافق واقع شده قابل انکار نیست



HASHED MESSAGE AUTHENTICATION (HMAC)

○ دو متغیر با مقادیر ثابت

ipad = 363636....

opad = 5c5c5c....

○ عملکرد

$T_1 = \text{Key} \oplus \text{ipad}$

$T_2 = \text{CONCAT}(T_1, \text{Message})$

$H_1 = \text{Hash}(T_2)$

$T_3 = \text{Key} \oplus \text{opad}$

$T_4 = \text{CONCAT}(T_3, H_1)$

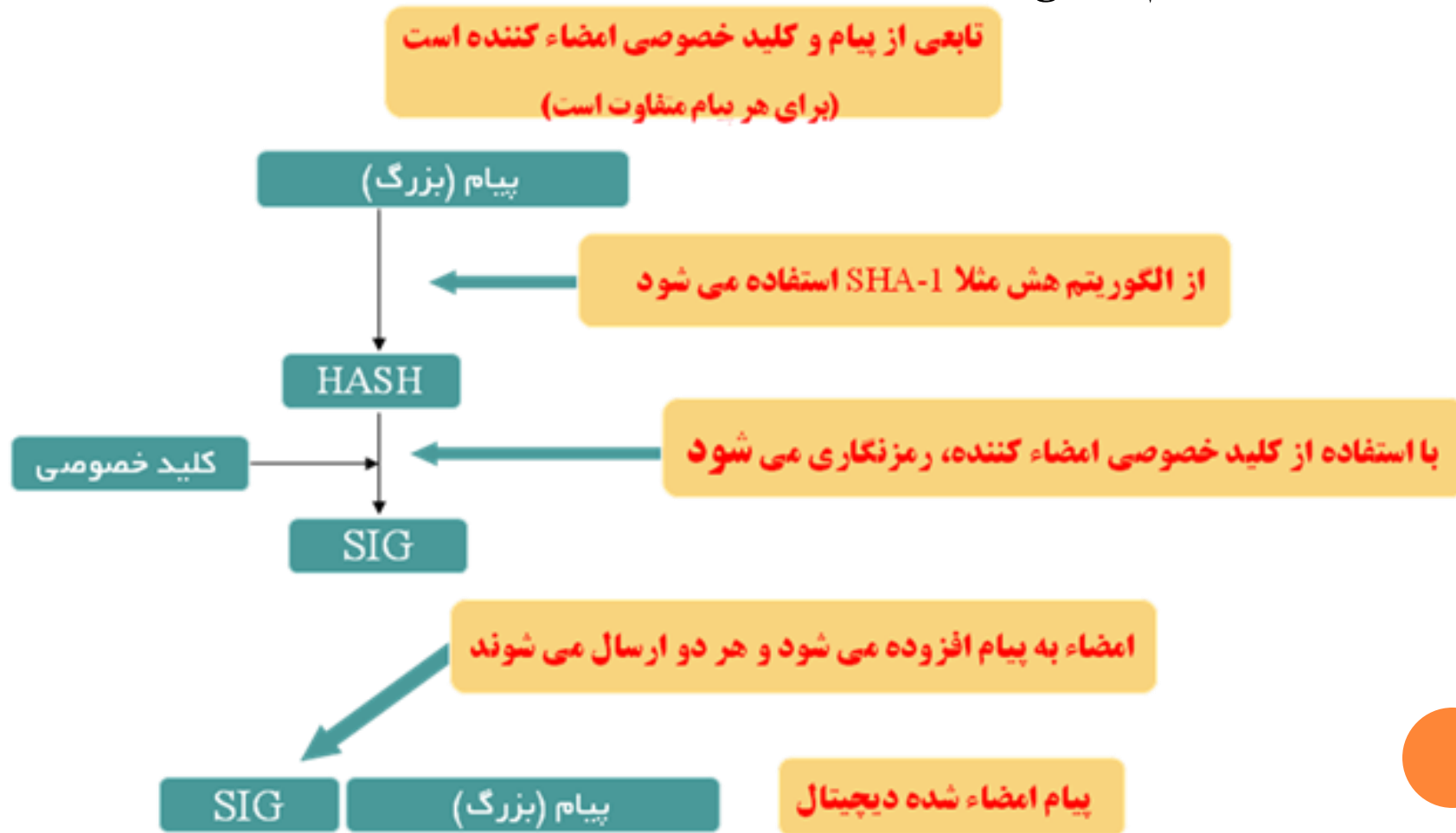
$H_2 = \text{Hash}(T_4)$

Return H_2



امضای دیجیتال

- امضای دیجیتال مبتنی بر الگوریتم‌های رمزنگاری نامتقارن و الگوریتم‌های Hashing است.

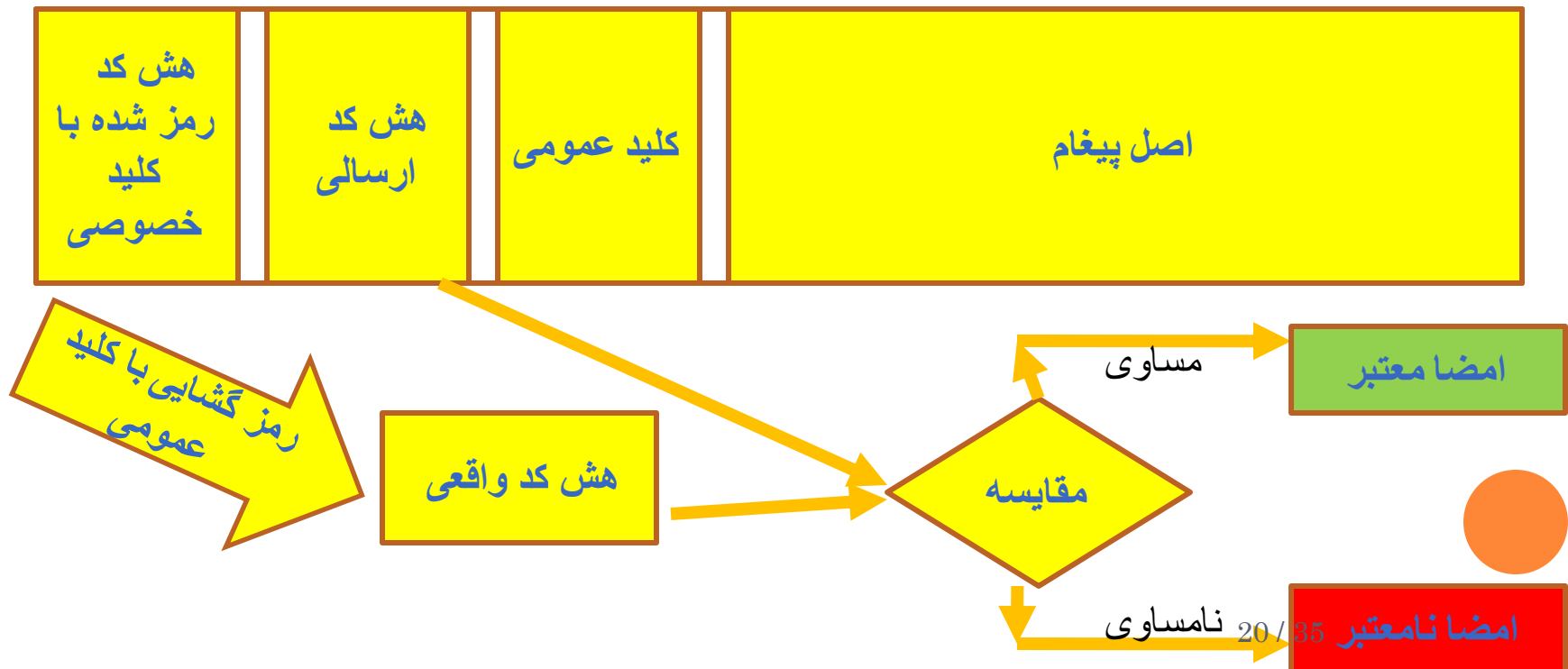


روال امضا کردن

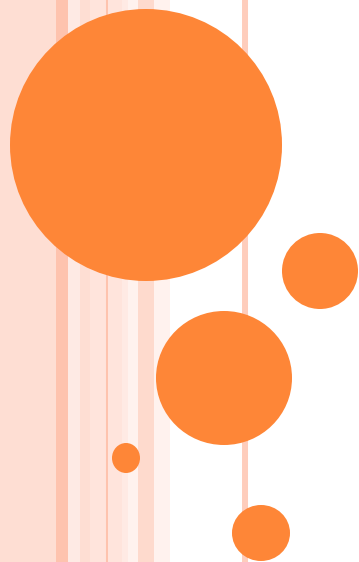
- پیش از ارسال داده‌ها، آنها را با استفاده از الگوریتم‌های Hashing به یک کد فشرده Hash تبدیل می‌کنند. مقادیر هش شده همگی طول یکسانی دارند و در صورت تغییر در اطلاعات ورودی Hash Code جدیدی تولید می‌شود. این الگوریتم‌ها همگی یک طرفه هستند، یعنی پس از کد شدن اطلاعات نمی‌توان از روی این کدها اطلاعات اصلی را به دست آورد.
- در جریان ارسال اطلاعات کد Hash به دست آمده از الگوریتم محاسباتی توسط کلید خصوصی به حالت رمز تبدیل می‌شود و همراه با کلید عمومی به انتهای داده‌ها اضافه شده و برای گیرنده ارسال می‌شود به علاوه کد Hash واقعی داده‌ها نیز محاسبه شده و در انتها این دو کد باهم مقایسه می‌شوند.

روال چک کردن امضا

- اگر این Hash کدهای ارسالی و واقعی همخوانی داشتند بیانگر این است که داده های ارسال شده دستکاری نشده‌اند و قابلیت اعتماد دارند اما در صورتی که یکسان نباشند به معنای دستکاری در اطلاعات است و این اطلاعات دیگر قابل اطمینان نیستند.



مفهوم شکستن چکیده سازی و حمله برخورد یا مواجهه



مفهوم شکستن چکیده سازی و حمله برخورد یا مواجهه

○ هنگامی که از شکستن یک روش چکیده سازی سخن میگوییم منظور این است که بتوانیم دو متن $P1, P2$ متفاوت پیدا کنیم که که چکیده آنها یکی باشد،

$$P1\#P2 \quad \& \quad \text{Hash}(P1)=\text{Hash}(P2)$$

معمولا چنین موردی را یک مواجهه یا برخورد collision گویند،
حمله مواجهه یا حمله برخورد بر علیه یک روش چکیده سازی عبارت است از پیدا کردن دو متن $P1, P2$ بشرح فوق،



مفهوم شکستن چکیده سازی و حمله برخورد یا مواجهه

- اگر یک حمله موفق مواجهه یا حمله برخورد بر علیه یک روش چکیده سازی صورت بگیرد نتیجه اش این است که نشان میدهد روش چکیده سازی معتبر نیست، چرا؟
- چون ما با استفاده از چکیده سازی اعتبار سنجی و اصالت پیام را میسنجیم،
- مثلاً فرض کنید یک پیام داشته باشیم $P1 = \text{"123 \$"}$
- و $P2 = \text{"405 \$"}$
- و $\text{Hash}(P1) = \text{Hash}(P2)$
- آنگاه یک اخلالگر میتواند پیام $P1$ را به $P2$ تغییر دهد و متقلبانه اصالت آنرا اثبات و مبلغ بیشتری وصول کند !!



مفهوم شکستن چکیده سازی و حمله برخورد یا مواجهه

- بعلاوه مرسوم است که برای نگهداری password ها در پایگاه داده بجای اینکه خود password ها را نگهداریم، hash آنها را نگه میداریم،
- در واقع در اکثر پایگاههای داده در فیلدها نوع داده password قابل تعریف است،
- و در زبانهای برنامه نویسی هم معمولاً ابزارهای پردازش password مبتنی بر hash هستند،
- در اکثر موارد امکان انتخاب این هست که از md5 یا SHA استفاده کنیم،
- در اکثر موارد روش md5 استفاده میشود،
- ولذا در اکثر موارد آنچه در پایگاه داده نگهداری میشود md5(password) است.



مفهوم شکستن چکیده سازی و حمله برخورد یا مواجهه

- ولذا در اکثر موارد آنچه در پایگاه داده نگهداری میشود $\text{md5}(\text{password})$ است.
- اینکار باعث میشود اگر کسی بتواند به پایگاه داده دسترسی پیدا کند نتواند به password ها دسترسی پیدا کند،
- در حالیکه این تصور چندان درست نیست، چرا؟؟
- جالب است بدانید افراد خیری!!! با زحمت زیاد توانسته اند پایگاههای داده رایگان تحت وب درست کنند که در آن جدول تقابل $\text{hash}(P)$ و P فراهم شده است،
- به اینها جدول رنگین کمان rainbow tables میگویند،
- شما $\text{Hash}(P)$ را که از پایگاه داده بدست آورده اید به آنها میدهید و آنها هم برای اینکه بشما کمک کنند و کارتان راه بیافتد!!! P را بشما میدهند،
- این سایتها را معمولا سایتهای $\text{hash reverse search}$ یا جستجوی برعکس میگویند، و اکثرا ادعا میکنند که به کسانی که پسوردشان را گم کرده اند کمک میکنند،
- مثلاً سایت <http://www.cmd5.org/> دارای 7,800,000,000,000 رکورد است، امتحان کنید پشیمان نمیشوید!!!



مفهوم شکستن چکیده سازی و حمله برخورد یا مواجهه

- ضمنا این خارجی ها فقط حروف انگلیسی اعداد و علائم را در پایگاه داده اشان نگه میدارند
- اگر از حروف فارسی و َ ، ِ ، ُ ، ً ، ٌ ، ٍ ، ۚ ، ۛ ، ۞ در پسورد استفاده کنید احتمالا امن تر هستند،
- ولی نه برای همیشه چون اینها هم در فونتهای unicode هستند!!!
- ضمنا یکی از کارهای خیر همین راه اندازی سایت کمک به پسورد گم کردگان فارسی است،
- استفاده از جدول تقابل P و hash(P) موضوعی بسیار پر کاربرد در هک و دسترسی غیر مجاز است،
- نمونه ای از آنرا میتوانیم همین الان ببینیم!!



مفهوم شکستن چکیده سازی و حمله برخورد یا مواجهه

- همچنین ما از Hash در امضای دیجیتال استفاده میکنیم،
- با اخلاف در Hash میتوان امضای نادرستی را بصورت امضای اصل نشان دهیم،
- شاید جدیدترین نتیجه اخلاف در Hash اثر مخرب آن در صدور گواهیهای دیجیتال Https و SSL باشد،
- اگر چنین حادثه ای رخ دهد براحتی میتوان سایت نادرستی را با یک امضای جعلی https بعنوان سایت اصلی و معتبر تبلیس کنیم، توجه کنید که تقریبا تمام امنیت شکننده پرداختهای الکترونیک و وبسایتهای بانکی بر اساس همین https و SSL صورت میگیرد.



مفهوم شکستن چکیده سازی و حمله برخورد یا مواجهه

- میدانید که یکی از مشهورترین روشهای سرقت اطلاعات، پخش ویروس، پسورد دزدی، ثبت و جاسوسی از عملیات کاربران سیستمهای کامپیوتری انتشار بدافزارها با استفاده از نرم افزارهای معمول و شناخته شده است.
- مثلاً فرض کنید یک keylogger را در نرم افزار word بتوانیم bind و پنهان کنیم.
- به این روش و قتی کسی از این word تقلبی استفاده میکند، تمام اطلاعات و عملیات او از طریق keylogger برای هکر سازنده keylogger ارسال میشود،
- برای جلوگیری از هر نوع اختلال در نرم افزارهای اصیل، شرکتهای معتبر، مثل microsoft برای هر نرم افزار Hash آنرا نیز منتشر میکنند، (معمولاً MD5)
- با این حساب اگر کسی نرم افزار اصلی را دستکاری و چیزی به آن اضافه کند از طریق تطبیق hash عدم اصالت آن آشکار میشود،



مفهوم شکستن چکیده سازی و حمله برخورد یا مواجهه

- حالا فرض کنید بشود راهی پیدا کرد که بتوان نرم افزاری دستکاری کرد بطوریکه hash آن تغییر کند!!
- مثلاً word را به word+keylogger تغییر دهیم ولی
$$\text{Hash}(\text{word}) = \text{Hash}(\text{word} + \text{keylogger})$$
- به این روش کاربر با بررسی hash تصور میکند نرم افزار اصیل و سالمی را استفاده میکند در حالیکه نرم افزار همراه با یک جاسوس افزار یا بد افزار همراه است،
- در اینجا دو نمونه پیام متفاوت را ارائه میدهم که دارای hash یکسان هستند،



مفهوم شکستن چکیده سازی و حمله برخورد یا مواجهه

d131dd02c5e6eec4693d9a0698aff95c 2fcab58712467eab4004583eb8fb7f89
55ad340609f4b30283e488832571415a 085125e8f7cdc99fd91dbdf280373c5b
d8823e3156348f5bae6dacd436c919c6 dd53e2b487da03fd02396306d248cda0
e99f33420f577ee8ce54b67080a80d1e c69821bcb6a8839396f9652b6ff72a70

And

d131dd02c5e6eec4693d9a0698aff95c 2fcab50712467eab4004583eb8fb7f89
55ad340609f4b30283e4888325f1415a 085125e8f7cdc99fd91dbd7280373c5b
d8823e3156348f5bae6dacd436c919c6 dd53e23487da03fd02396306d248cda0
e99f33420f577ee8ce54b67080280d1e c69821bcb6a8839396f965ab6ff72a70

Each of these blocks has MD5 hash

79054025255fb1a26e4bc422aef54eb4



مفهوم شکستن چکیده سازی و حمله برخورد یا مواجهه

○ کار به همینجا ختم نمیشود،

○ دو مورد پیامی که در بالا دیدیم پیامهای بامعنی نبودند، آیا میتوان دو پیام با معنی و مشخص پیدا کرد که دارای hash یکسان باشند؟

○ مثلاً آیا میتوان دو برنامه اجرایی پیدا کرد که دارای hash یکسان باشند؟

○ پاسخ مثبت است!! با استفاده از الگوریتمی بنام Diaz میتوان برنامه های متفاوتی تولید کرد که دارای md5 یکسان باشند،

○ دو برنامه زیر

○ Windows version:

- hello.exe. MD5 Sum: cdc47d670159eef60916ca03a9d4a007
- erase.exe. MD5 Sum: cdc47d670159eef60916ca03a9d4a007

○ Linux version (i386):

- hello. MD5 Sum: da5c61e1edc0f18337e46418e48c1290
- erase. MD5 Sum: da5c61e1edc0f18337e46418e48c1290



مفهوم شکستن چکیده سازی و حمله برخورد یا مواجهه

```
C:\TEMP> md5sum hello.exe
Cdc47d670159eef60916ca03a9d4a007
C:\TEMP> .\hello.exe
Hello, world! (press enter to quit)
C:\TEMP>
```

```
C:\TEMP> md5sum erase.exe
Cdc47d670159eef60916ca03a9d4a007
C:\TEMP> .\erase.exe This program is evil!!!
Erasing hard drive...1Gb...2Gb... just kidding!
Nothing was erased. (press enter to quit)
C:\TEMP>
```



مفهوم شکستن چکیده سازی و روش مقابله نمک زدن، PASSWORD در SALTING ها

- خاطرتان هست که گفتیم اگر $\text{hash}(\text{password})$ را داشته باشیم معمولاً پیدا کردن password امکانپذیر است،
- روشی برای مقابله با این حمله وجود دارد، salting یا نمک زدن،
- در روش نمک زدن بجای نگهداری $\text{hash}(\text{password})$ در پایگاه داده یک مقدار ثابت salt = نمک مخصوص به password اضافه میکنیم،
- و $\text{hash}(\text{salt} + \text{password})$ را نگه میداریم،
- با این روش حداقل مطمئن هستیم اگر کسی به hash دسترسی پیدا کرد از روی آن نتواند به password دسترسی داشته باشد، چرا؟؟
- متأسفانه یا خوشبختانه این روش هم چندان امن نیست.
- در واقع روشی بنام salted hash cracking هست که براحتی مسئله را حل میکند!!



روش شکستن SALTED PASSWORDS

- با توجه به تکنیک پیچیده این روش، بطور خلاصه بیان میکنیم که الگوریتمی بنام wang در سال ۲۰۰۹ ارائه شده است که تقریباً میتواند پیچیدگی شکستن یک md5 را به 2^{29} تقلیل دهد!!
- پیچیدگی اصلی md5 برابر بود با 2^{128} که بنظر محکم و امن است، ولی با روشهای تحلیل بلوکی پیچیدگی به 2^{29} کاهش پیدا میکند،
- نرم افزارهای زیادی هستند که زحمت اینکار را میکشند، مثلاً نرم افزار زیر،



SALTED PASSWORDS روش شکستن

- Launch **MD5 Salted Hash Kracker** .
- Enter the MD5 hash.
- Then enter the Salt data and specify the Salt position either in the beginning [md5(salt+pass)] or at the end [md5(pass+salt)]
- For normal MD5 hash cracking, leave the Salt field empty.
- Next click on 'Start Crack' button to begin the Recovery operation
- During the operation, you will see all statistics being displayed on the screen.
- On success, message box is displayed and recovered password is automatically copied to clipboard.
- At the end, you can generate detailed report in **HTML/XML/Text** format by clicking on 'Report' button and then select the type of file from the drop down box of 'Save File Dialog'.



روش شکستن SALTED PASSWORDS

MD5 Salted Hash Cracker - www.SecurityXploded.com


MD5 Salted Hash Cracker
Salted MD5 Hash Password Cracker and Recovery Software


[Show Help](#) [About](#)

MD5 Hash:

Salt Text:

Salt Position: ☐ Beginning (Salt+Pass) ☒ At the End (Pass+Salt)

Password Dictionary File: 

 **Start Crack**


Cracking Status

Password Count:

Total Elapsed Time:

Cracking Speed:

Current Status:



Download More Password Tools from SecurityXploded

 **Report**

روش شکستن SALTED PASSWORDS



کاربرد شکستن HASH در ویروس نویسی و سلاحهای سایبری

در اینجا قصد نداریم وارد مبحث سلاحهای سایبری شویم، ولی هر طوریکه سلاحهای سایبری را تعریف کنیم ویروس مخرب و مشهور ضد ایرانی flame بعنوان یک سلاح سایبری محسوب میشود، این ویروس با استفاده از جعل امضای اصالت microsoft خود را بعنوان یک update سیستم windows معرفی میکند، در حالیکه واقعا دارای امضای واقعی microsoft نمیشود،



کاربرد شکستن HASH در ویروس نویسی و سلاحهای سایبری

توجه داریم که امضای دیجیتال معمول در شرکت microsoft مبتنی بر hash با روش md5 است،

با بررسیهای انجام شده معلوم شده است که ساخت یک امضای تقلبی معادل با امضای واقعی حدود \$200000 هزینه دارد،

بر همین اساس حدس این است که یک دولت (کشور) مجری ساخت flame بوده است،

شاید همین الان یک نرم افزار مخرب روی کامپیوتر شما در حال کار است و دارای امضای تصدیق اصالت (البته جعلی) شرکت microsoft میباشد.

البته stuxnet و duqu هم از همین ابزار تبلیس و جعل امضاهاى دیجیتال شرکتهاى معتبر استفاده کرده اند،