به نام خدا

دانشکده‌ی علوم ریاضی

مقدمه‌ای بر رمزنگاری آذر ۱۳۹۲

جلسه‌ی ۱۸: ساختن توابع چکیده ساز و طراحی سیستم CCA امن

مدرس: دکتر شهرام خزائی *نگارنده: لعیا قدرتی*

در این جلسه سعی داریم که بعضی از روش های ساختن یک تابع چکیده ساز [۱]مقاوم در برابر برخورد را بررسی کنیم. ساختار های Merkle-Damgard و اسفنجی [۲] دو روش کلی برای طراحی توابع چکیده ساز مقاوم در برابر برخورد با طول ورودی دلخواه هستند.که در هر دو روش برای ساخت توابع چکیده ساز از تابعی فشرده ساز [۳] با طول ورودی ثابت استفاده می شود.
در واقع این ساختارها روشی برای تبدیل یک تابع فشرده ساز مقاوم در برابر برخورد با طول ورودی ثابت، به تابع چکیده ساز مقاوم در برابر برخوردی است که می تواند روی ورودی با هر طولی عمل کند.
در جلسه ی گذشته ساختار Merkle-Damgard بررسی شد. در این جلسه تعدادی از مدهای توابع فشرده ساز بررسی می شود و سپس ساختار اسفنجی را معرفی می کنیم.

۱ ساختن توابع فشرده ساز

برای ساختن یک تابع فشرده ساز، ایده اولیه که ممکن است به ذهن برسد استفاده از رمزهای قالبی[۴] است. برای مثال می توانیم قرار دهیم:

$$f(h,m) = E_h(m)$$

برای چنین تابعی به راحتی با در دست داشتن E^{-1} می توانیم برخورد بسازیم. درواقع (h_1, m_1) و (h_2, m_2) را می یابیم که مقدارf برایشان برابر باشد.

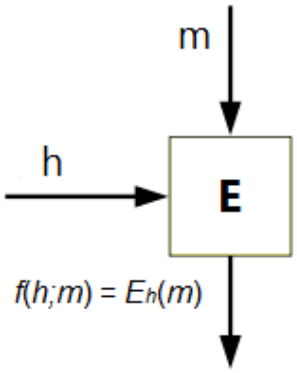
$$m_1 = E_{1^n}^{-1}(0^n)$$

،

$$m_2 = E_{1^n}^{-1}(0^n)$$

که E یک رمز قالبی با کلید h است. اما با تغییر کوچکی در ایده قبل تابع مقاوم در برابر برخورد خواهد شد.به توابعی که در ادامه می آید دقت کنید.

[۱] Hash Function
[۲] Sponge Construction
[۳] Compress Function
[۴] block cipher



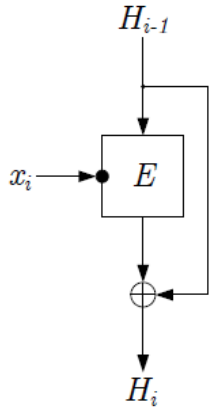
شکل ۱ : استفاده از رمز قالبی

۲ تابع فشرده ساز Oseas-Meyer-Matyas

این تابع به این صورت است که ابتدا ورودی به صورت تعدادی بلاک تجزیه می شود و هر بلاک x_i به عنوان کلید رمز قالبی E در نظر گرفته می شود.و در مرحله i خروجی مرحله ی قبل، $H_{i-۱}$ ،به عنوان ورودی به E_{x_i} داده می شود و سپس دوباره با $H_{i-۱}$ xor می شود.در واقع داریم:

$$H_i = E_{x_i}(H_{i-۱}) \oplus H_{i-۱}$$

و $H_۰ = IV$.



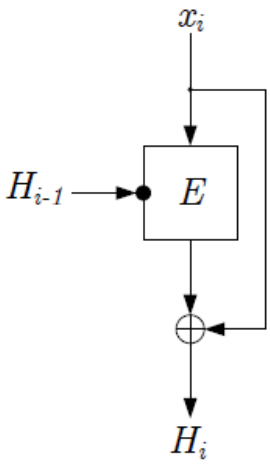
شکل ۲ : Oseas-Meyer-Matyas

۳ تابع فشرده ساز Davies- Meyer

این تابع فشرده ساز شباهت زیادی با تابع قبلی دارد، تنها تفاوت این است که اینبار هر بلاک ورودی، x_i، به عنوان ورودی به رمز قالبی $E_{H_{i-1}}$ داده می شود که H_{i-1} خروجی مرحله ی قبل است. لذا داریم:

$$H_i = E_{H_{i-1}}(x_i) \oplus x_i$$

و $H_0 = IV$.



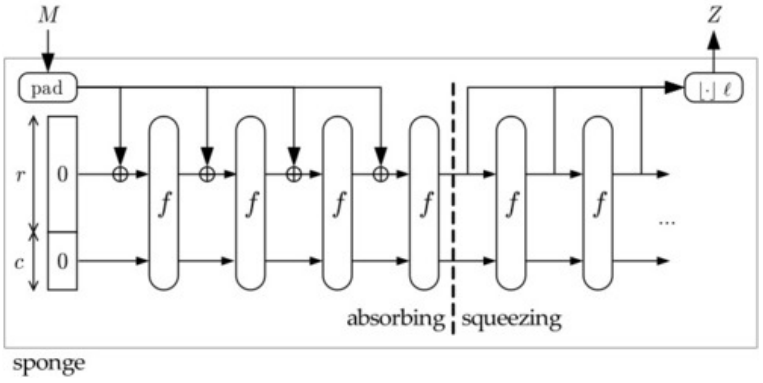
شکل ۳: Davies- Meyer

۴ ساختار اسفنجی

ساختار اسفنجی [۵] یک ساختار ساده ی تکرار شونده است که برای ساختن یک تابع F با اندازه ی ورودی و خروجی دلخواه استفاده می شود و از یک جایگشت f که روی b بیت عمل می کند ساخته می شود. این ساختار روی $b = r + c$ بیت عمل می کند که به این r بیت bitrate و به c بیت دیگر ظرفیت [۶] گفته می شود.
در ابتدا ورودی با ۱ و ۰ پد می شود تا طول آن مضرب r شود و به بلاک های با طول r تجزیه می شود.سپس b بیت ، برابر صفر مقدار دهی می شوند و ساختار ما در دو مرحله ی زیر انجام می پذیرد:

- در مرحله ی جذب ، بلاک با طول r با r بیت اولیه XOR می شود.تا r بیت جدید به همراه c بیت صفر اولیه به عنوان ورودی تابع f داده شوند. وقتی برای تمام بلاک ها عمل بالا صورت پذیرفت وارد مرحله ی بعدی فشرده سازی می شویم.

[۵] sponge construction
[۶] capacity



شکل ۴: ساختار اسفنجی

- در مرحله ی فشرده سازی ، r بیت اولیه خروجی مرحله ی قبل به همراه اعمال تابع f به تعداد دلخواه روی آن ، به عنوان خروجی داده می شود. نکته: c بیت های هر مرحله به عنوان خروجی فشرده سازی استفاده نمی شوند.

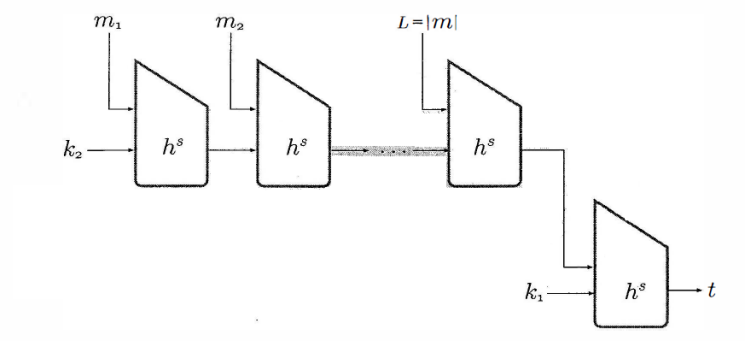
تا به این جا بعضی از کدهای اصالت سنجی مبتنی بر رمزهای قالبی را دیده ایم.یک روش دیگر برای ساختن این کدها استفاده از توابع چکیده ساز است.برای مثال NMAC و HMAC دو ساختار هستند که با استفاده از Merkle-Damgrad ساخته می شوند که در قسمت بعد آنها را بررسی می کنیم. یکی از کاربردهای توابع درهم ساز استفاده از آنها در ساختن کدهای اصالت سنجی است. به اختصار دو نوع MAC مختلف NMAC و HMAC را که از توابع درهم ساز استفاده می کنند را بررسی می کنیم.

۵ Nested MAC(NMAC)

فرض کنید که تابع H تابع چکیده ساز ساخته شده توسط ساختار Merkle-Damgard است که روی تابع فشرده ساز h اعمال شده است. حال اگر s یک کلید ثابت و غیرمحرمانه باشد، ورژن کلید-محرمانه ی h_s به صورت مقابل تعریف می شود: $h^s_k(x) = h^s(k||x)$.
H^s_k را تابع چکیده سازی که از قرار دادن $IV = K$ و با استفاده از Merkle-Damgard ساخته می شود، تعریف می کنیم. بنابراین برای مثال در مرحله ی اول خروجی $z_1 = h^s(k||x_{x_1}) = h^s_k(x)$ بدست می آید. بنابراین HMAC به این صورت کار می کند که ابتدا $H^s_{k_2}$ روی ورودی پد شده اعمال شده و سپس خروجی بعنوان ورودی به تابع فشرده ساز $h^s_{k_1}$ با کلید محرمانه ی k_1 داده می شود.در آخر خواهیم داشت: $t = h^s_{k_1}(H^s_{k_2}(m))$.

۶ HMAC

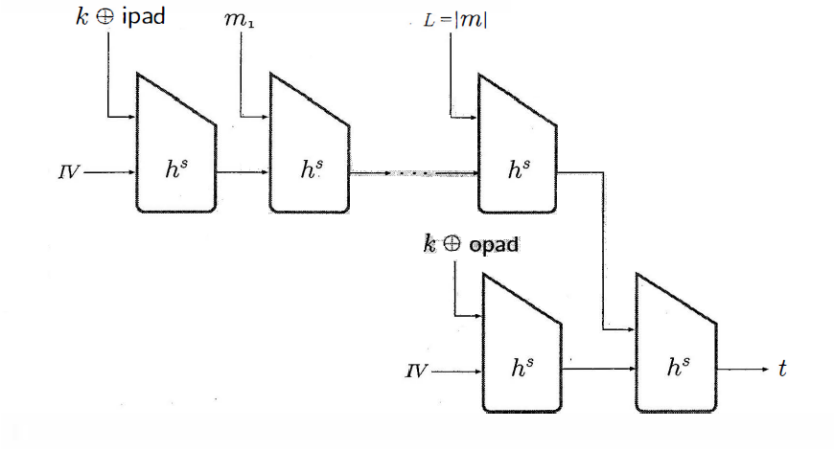
یکی از اشکالات HMAC این است که IV که در H استفاده می شود را تغییر می دهد که این می تواند اشکالاتی را در مرحله ی اجرا بوجود بیاورد.HMAC این مشکل را حل می کند و نیز برخلاف NMAC به جای دو کلید از یک کلید بهره می گیرد.
H و h را مانند قسمت قبل در نظر بگیرید.فرض کنید که IV یک مقدار ثابت باشد همچنین فرض کنید که $opad$ و $ipad$ دو مقدار ثابت بصورت مقابل باشند: $opad = ox\text{۵}c\text{۵}c\text{۵}c../\text{۵}c$ ، $ipad = ox\text{۳۶۳۶}../\text{۳۶}$



شکل ۵: NMAC

اولین مرحله، محاسبه ی $H^s_{IV}((k \oplus ipad)||m)$ است. بعد از آن خروجی کلی به صورت مقابل محاسبه می شود:

$$HMAC_k(m) = H^s_{IV}((k \oplus opad)||H^s_{IV}((k \oplus ipad)||m)) = H^s_{k_1}(H^s_{k_2}(m))$$



شکل ۶: HMAC

امروزه HMAC به صورت یک استاندارد در آمده و بصورت گسترده ای مورد استفاده قرار می گیرد.علاوه بر اینکه به سادگی قابل پیاده سازی است دارای اثبات برای امنیت نیز هست.

۷ رمزنگاری احراز اصالت شده

در جلسات گذشته آزمایش مربوط به سیستم CCA امن را تعریف کردیم. در ادامه ی این جلسه سعی داریم یک سیستم امن دارای امنیت متن رمزی انتخابی [۷] بسازیم.
ابتدا آزمایش مربوط به این نوع امنیت را یادآوری می کنیم.

[۷] CCA-Secure

۱. چالشگر یک کلید k تولید می کند: $k \leftarrow \mathsf{Gen}()$

۲. مهاجم دارای دسترسی اوراکلی پیام‌های $m_0, m_1 \in \mathcal{M}$ را انتخاب کرده و به چالشگر می‌دهد:
$m_0, m_1 \leftarrow \mathcal{A}^{Enc_k(.),Dec_k(.)}(1^n)$

۳. چالشگر بیت تصادفی b را تولید می‌کند: $b \leftarrow \{0, 1\}$

۴. چالشگر متن رمزی c، رمزشده m_b تحت کلید k، را محاسبه می‌کند و به مهاجم می‌دهد: $c \leftarrow \mathsf{Enc}_k(m_b)$

۵. مهاجم $\mathcal{A}$ بیت $\hat{b}$ را بیرون می‌دهد: $\hat{b} \leftarrow \mathcal{A}^{Enc_k(.),Dec_k(.)}(c)$
دقت شود که مهاجم مجاز به درخواست رمزگشایی متن رمزی c نیست.

خروجی آزمایش که با $\mathsf{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}$ نشان داده می‌شود برابر یک در نظر گرفته می‌شود اگر $\hat{b} = b$ و صفر است اگر $\hat{b} \neq b$.
روش به این صورت است که دو کلید محرمانه بین فرستنده و گیرنده به اشتراک گذاشته می شود.یکی از کلیدها مربوط به ساختار CPA امن و دیگری مربوط به کد اصالت سنجی است. فرض کنید که Π_1 یک سیستم رمز CPA امن و Π_2 یک کد اصالت سنجی جعل ناپذیر باشد.آنگاه اگر $c = Enc_{k_1}(m)$ و $t = MAC_{k_2}(c)$ باشد یک سیستم رمزگذاری با $< c, t > \leftarrow \mathsf{Enc}_k(m_b)$ را به عنوان خروجی دهد آنگاه این سیستم CCA امن است. گیرنده بعد از دریافت $< c, t >$ چک می کند که آیا تگ t روی c معتبر است یا خیر و سپس اقدام به رمزگشایی می کند.
نکته در این روش این است که هیچ مهاجم چندجمله ای نمی تواند یک متن رمزی معتبر را بدون دانستن کلیدها تولید کند و بنابراین ساختار دارای امنیت مورد انتظار است.
بطور شهودی نیز بنظر می رسد اگر سیستمی CCA امن باشد حمله کننده نباید قابلیت ساختن یک متن رمزی معتبر را داشته باشد. بنابراین آزمایش جامعیت متن رمزی را بصورت زیر تعریف می کنیم:
آزمایش جامعیت متن رمزی (CI) [۸]:

۱. $k \leftarrow \mathsf{Gen}(1^n)$

۲. $c \leftarrow \mathcal{A}^{Enc_k(.)}(1^n)$
اگر Q مجموعه ی متن های رمزی دریافتی توسط A باشد آنگاه $c \notin Q$

۳. اگر $Dec_k(c) \neq \perp$ خروجی آزمایش ۱ و در غیر این صورت ۰ است.

تعریف ۱ *یک سیستم دارای جامعیت متن رمزی است اگر برای هر مهاجم چندجمله ای احتمالی A تابع ناچیز negl وجود داشته باشد که:*

$$\Pr\{PrivK^{CI}_{A,\Pi}(1^n) = 1\} \leq negl(n)$$

برای ساختن یک سیستم CCA امن این کافی نیست که تنها جامعیت متن رمزی داشته باشد، درواقع شرایط کافی به صورت زیر است:

سیستم CPA امن باشد و جامعیت متن رمزی داشته باشد. $\Longleftarrow$ CCA امن است.

[۸] Ciphertext Integrity

دانشکده‌ی علوم ریاضی

مقدمه‌ی بر رمزنگاری ۲۰ اردیبهشت ۹۳

جلسه۲۳: تابع چکیده‌ساز و نحوه ساختن کد اصالت‌سنجی پیام با آن

استاد: دکتر میر امید حاجی میر صادقی *نگارندگان: مصطفی کریمی و امیر عرفان عشرتی‌فر*

۱ تابع چکیده‌ساز

تابع چکیده‌ساز تابعی است که پیامی را به طول دلخواه فشرده می‌کند. برای کاربردهای رمزنگاری، یک تابع چکیده‌ساز مطلوب دارای ویژگی‌های زیر است:

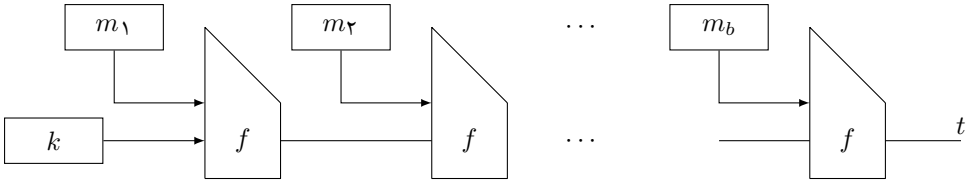
- مقاوم در مقابل برخورد
- مقاوم در مقابل پیش تصویر
- مقاوم در مقابل پیش تصویر دوم

۲ نحوه ساختن کد اصالت‌سنجی پیام با تابع چکیده‌ساز

در جلسات قبلی با تابع چکیده‌ساز، ویژگی‌های آن و نحوه ساخت آن در جلسات قبلی آشنا شدیم. در این جلسه با بعضی از روش‌های ساخت کد اصالت‌سنجی پیام با استفاده از توابع چکیده‌ساز به روش مرکل دمگارد آشنا می‌شویم. در ادامه با طرح‌های مختلف که همه بر اساس ساختار مرکل دمگارد هستند مواجه می‌شویم. به صورت مرحله به مرحله ضعف‌های ساختارهای قبلی رابر طرف کرده تا درنهایت به یک کد اصالت‌سنجی امن برسیم. از ۵ طرح پیشرو، فقط ۲ مورد از آنها قابل قبول هستند.

۱.۲ طرح پیشنهادی اول

اولین تلاش برای بدست آوردن کد اصالت‌سنجی پیام، استفاده از ساختار مرکل-دمگارد و جایگزینی IV با کلید است. در این طرح، از طول سیگنال استفاده نمی‌کنیم. در ابتدا با استفاده از لایه‌گذاری، پیام دلخواه را به b بلوک تبدیل کرده و به صورت زیر به یک کد اصالت‌سنجی پیام می‌رسیم.



با استفاده از حمله گسترش پیام به راحتی می‌توان به این طرح حمله کرد. همچنین معمولا در توابع چکیده‌ساز دوست داریم مقدار اولیه ثابت باشد تا بتوانیم از یک تابع چکیده‌ساز ثابت استفاده کنیم و نیازی به تغییر تابع برای استفاده مجدد نداشته باشیم. در این طرح چون از کلید به عنوان مقدار اولیه استفاده می‌کنیم ثابت نیست. ولی به طور کلی ضعف این طرح، نداشتن امنیت کافی است.

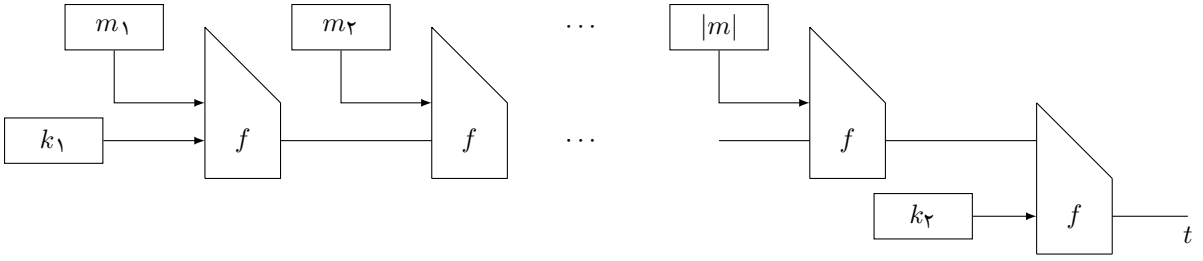
۲.۲ طرح پیشنهادی دوم

در ادامه تلاشمان برای جلوگیری از حمله گسترش متن، طول پیام را در انتها اضافه می‌کنیم. ولی دوباره این طرح هم در مقابل حمله گسترش متن مقاوم نیست.

این مدل به هیچ وجه مطلوب نیست. زیرا اولا امنیت ندارد. ثانیا در تابع چکیده ساز دوست داریم قسمت IV یک عدد ثابت باشد نه متغیر.

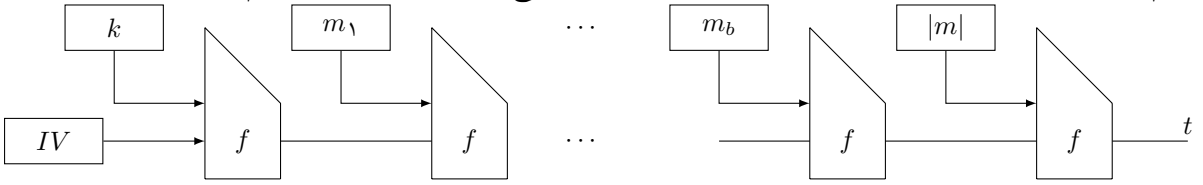
۳.۲ طرح پیشنهادی سوم

در طرح قبلی دیدیم که فقط با اضافه کردن طول پیام نمی‌توان در برابر حمله مقاوم بود. در این طرح با استفاده از دو کلید k_1 و k_2 می‌توان امنیت کافی را بدست می‌آورد. این طرح $NMAC$ نامیده می‌شود.



۴.۲ طرح پیشنهادی چهارم

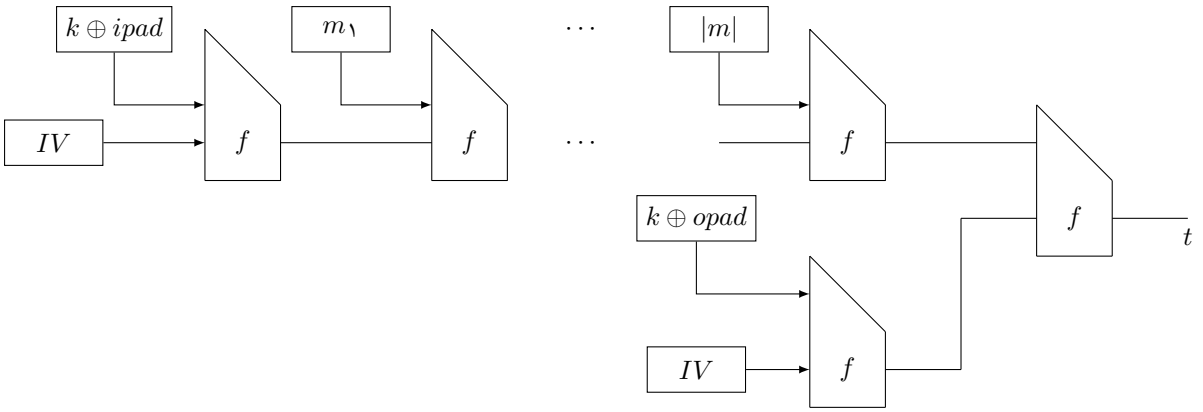
با استفاده از طرح قبلی توانستیم امنیت لازم را بدست آوریم. ولی همانطور که در توضیحات طرح اول گفته شد. دوست داریم مقدار اولیه، مقدار ثابتی باشد. برای رسیدن به این هدف طرح زیر را پیاده‌سازی می‌کنیم.



ولی با تغییراتی که در این طرح دادیم دوباره با حمله گسترش متن، می‌توان به این طرح حمله کرد.

۵.۲ طرح پیشنهادی پنجم

این طرح هر دو ویژگی مطلوب، امنیت و مفدار اولیه ثابت، را دارا است. این مدل $HMAC$ نامیده می‌شود.



که کلیدها به صورت زیر بدست می آیند:

$$k_1 = k \oplus ipad = k \oplus 0X363636..... \tag{1}$$

$$k_2 = k \oplus opad = k \oplus 0X5c5c5c..... \tag{2}$$

۳ امنیت جعل ناپذیری قوی

همانطور که در جلسات قبل خواندیم، در امنیت جعل‌ناپذیری ضعیف، به سیستم کد اصالت‌سنجی پیام، نمی‌توان پیام تکراری با برچسب جدید، داد. ولی در امنیت جعل‌ناپذیری قوی، سیستم کد اصالت سنجی پیام باید امنیت قوی داشته باشد که حتی با پرسمان پیام تکراری با برچسب جدید، سیستم شکسته نشود. آزمایش آن به صورت زیر است:

- الگوریتم تولید کلید: $k \leftarrow Gen(1^n)$
- حمله کننده A که ورودی 1^n دریافت می‌کند و دسترسی اوراکلی به $Mac_k(.)$ دارد در نهایت $\langle m, t \rangle$ را به عنوان خروجی بدهد. $\langle m, t \rangle$ خروجی نباید هم پیام و هم برچسب تکراری باشد. مشکلی ندارد اگر پیام تکراری باشد ولی برچسب تکراری نباشد.
- خروجی آزمایش برابر ۱ است اگر$Verfy_k(m, t) =$ ۱ در غیر این صورت برابر ۰ است.

تعریف ۱ *Mac دارای امنیت جعل‌ناپذیری قوی است هرگاه به ازای هر حمله کننده A چندجمله‌ای تصادفی غیریکنواخت تابع ناچیز $\epsilon(n)$ وجود داشته باشد که:*

$$\Pr[SMacForge_{A,\pi}(n) = 1] \leq \epsilon(n) \tag{3}$$

۴ امنیت متن رمزی انتخاب‌شده

همانطور که در جلسات قبل مشاهده کردیم، آزمایش امنیت متن رمزی انتخابی از قوی‌ترین امنیت‌ها در بین تمام امنیت‌هایی است که مطالعه کردیم. آزمایش مربوطه به صورت زیر است:

- الگوریتم تولید کلید: $K \leftarrow Gen(1^n)$
- حمله کننده A ورودی 1^n را دریافت می‌کند. با استفاده از دسترسی اوراکلی به رمزگشا و رمزکننده دو پیام m_1 و m_0 را که طول یکسانی دارند را به عنوان خروجی بیرون می‌دهد.
- بیت b به صورت کاملا تصادفی از $\{0, 1\}$ انتخاب می‌شود.
- پیام m_b را با استفاده از $Enc_k(m_b)$ به رمز c تبدیل کرده و به حمله کننده می‌دهیم.
- حمله کننده با دریافت متن رمز شده c به عنوان خروجی و همچنین با استفاده از قابلیت دسترسی اوراکلی به رمز کننده و رمزگشا بیت b' را به عنوان خروجی می‌دهد. حمله کننده حق پرسمان متن رمز شده c را از رمزگشا ندارد.
- جواب آزمایش ۱ است اگر $b = b'$ باشد و برابر ۰ است اگر $b \neq b'$ باشد.

تعریف ۲ سیستم رمز متقارن دارای امنیت متن رمز انتخاب‌شده است اگر به ازای همه‌ی حمله کننده‌های چندجمله‌ای تصادفی غیریکنواخت، تابع ناچیز $\epsilon(n)$ وجود داشته باشد که :

$$\mathbb{P}[PrivK^{CCA}_{A,\pi}(n) = 1] \leq \epsilon(n) + \frac{1}{2} \tag{4}$$

۵ امنیت جامعیت رمزی

امنیت جامعیت رمزی، قوی‌ترین نوع امنیت است. این امنیت این قابلیت را می‌دهد که حمله کننده نتواند متن رمز شده قابل قبولی تولید کند. آزمایش به صورت زیر است:

- با استفاده ار الگوریتم تولید کلید $Gen(1^n)$ کلید k را تولید می‌کنیم.
- حمله کنندهA، ورودی 1^n را دریافت می کند و با استفاده از قابلیت دسترسی اوراکلی به رمزکننده و رمزگشا، متن رمزشده c را به عنوان خروجی می‌دهد.
- خروجی آزمایش ۱ است اگر وتنها اگر ۲ شرط زیر برقرار باشد:
  - اگر $Dec_k(c) \neq \perp$ باشد.
  - متن رمزی c را قبلا از Enc دریافت نکرده باشیم.

تعریف ۳ سیستم رمز متقارن دارای امنیت جامعیت رمزی است اگر به ازای همه‌ی حمله کننده‌های چندجمله‌ای تصادفی غیریکنواخت، تابع ناچیز $\epsilon(n)$ وجود داشته باشد که :

$$\mathbb{P}[PrivK^{CI}_{A,\pi}(n) = 1] \leq \epsilon(n) \tag{5}$$

۶ طراحی سیستمی با امنیت متن رمزی انتخابی و جامعیت رمزی

در این قسمت با استفاده از آنچه تاکنون آموخته‌ایم می خواهیم سیستم رمزی متقارن با امنیت متن رمزی انتخابی و جامعیت رمزی طراحی کنیم. فرض می‌کنیم که سیستم رمز متقارن π_E دارای امنیت CPA است. همچنین MAC داریم که دارای امنیت جعل‌ناپذیری قوی است. با ترکیب این دو سیستم می‌خواهیم یک سیستم که دارای امنیت متن رمزی انتخابی و جامعیت رمزی است، طراحی کنیم. ۳ روش عمده برای انجام این کار وجود دارد که فقط یکی از آنها مطلوب است.

۱.۶ روش اول

در این روش از هر دو سیستم به صورت موازی استفاده می‌کنیم. به این روش $Mac\ and\ Encrypt$ گفته می‌شود. به صورت جداگانه هر کدام از الگوریتم‌ها را پیاده سازی می‌کنیم:

- *با استفاده از سیستم رمز متقارن:* $c \leftarrow Enc_{k_1}(m)$
- *با استفاده از MAC:* $t \leftarrow Mac_{k_2}(m)$

در نتیجه، خروجی این روش متن رمزشده $c' = \langle c, t \rangle$ می‌باشد. این روش امن نیست زیرا در الگوریتم Mac ممکن است اطلاعاتی از متن اصلی وجود داشته باشد و حمله کننده به راحتی می‌تواند به آن دسترسی داشته باشد.

۲.۶ روش دوم

در این روش، از الگوریتم MAC قبل از سیستم رمز متقارن استفاده می‌کنیم. به این روش $Mac\ then\ Encrypt$ گفته می‌شود. روش به صورت زیر است:

- *با استفاده از Mac:* $t \leftarrow Mac_{k_2}(m)$
- *با استفاده از سیستم رمز متقارن:* $c \leftarrow Enc_{k_1}(\langle m, t \rangle)$

این روش هم امنیت مطلوب ما را ندارد.

۳.۶ روش سوم

در نهایت به روش مطلوب می‌رسیم. در این روش از الگوریتم Mac بعد از سیستم رمز متقارن استفاده می‌کنیم. به این روش $Encrypt\ then\ Mac$ گفته می‌شود. روش به صورت زیر است:

- *با استفاده از سیستم رمز متقارن:* $c \leftarrow Enc_{k_1}(m)$
- *با استفاده از Mac:* $t \leftarrow Mac_{k_2}(c)$

خروجی این روش $c' = \langle c, t \rangle$ است. می‌توان اثبات کرد که این روش امنیت دلخواه ما را دارا است.(اثبات امنیت به خواننده واگذار می‌شود.)