به نام خدا تعالی

مبانی رمزنگاری و امنیت شبکه — تمرین سری هشتم

دکتر امیر محسنی — پوریا دادخواه — ۹۶۱۰۶۴۸۵

۱ـ (12.4)

$M = 0^n \Rightarrow VMAC(k, 0^n) = CBC(k, 0) \oplus k_1 = T_0 \Rightarrow CBC(k, 0) = k_1 \oplus T_0$ (*)

$M = 1^n \quad CBC(k, 1^n) = k_1 \oplus T_0 \quad (*) \Rightarrow T_1 = CBC(k, CBC(k, 1^n)) = T_2 \oplus k_1$ (**)

پیام سوم — اگر پیام:

$0^n \| (T_0 \oplus T_1)$ باشد $\Rightarrow VMAC(k, 0 \| (T_0 \oplus T_1)) = CBC(k, 0^n \| (T_0 \oplus T_1)) \oplus k_1$

$= T_0 \oplus T_1 \oplus T_1 \oplus k_1 = T_2$ ✓

(**)

بنابراین همان طور که دیدیم، مهاجم با داشتن T_2 در واقع MAC پیام درست شده را می‌فهمد ✓

۲ـ (12.6) a)

- E → A: ابتدا verification C، سپس Decrypt C → M

b)

- A → E: ابتدا از الزامات MAC، سپس Encrypt
- H → E (هم‌زمان در دو طرف): Decrypt سپس verification M
- E + A: دو عمل موازی با هم انجام

۳- (12.9)

کلید \ متن	00	01	11	10
1	0	*	1	*
2	*	0	*	1
3	*	1	*	0
4	1	*	0	*

a) برای Bob ۵ حالت دارد

b) تحت هر کلید، Alice و Bob، دو حالت از چهار حالت فوق می‌فرستند. بنابراین مهاجم به احتمال $\frac{1}{2}$ آن را درست حدس می‌زند.

$$\begin{cases} 00 \\ 01 \\ 10 \\ 11 \end{cases}$$

⇐ احتمال این که مهاجم جای A پیام بفرستد $\frac{1}{2}$ است

c) در این حالت مهاجم می‌داند حداکثر ۲ کلید متفاوت می‌توانسته استفاده شود، اما نمی‌داند کدام کلید $\begin{cases} 1 \text{ یا } 4 \\ 2 \text{ یا } 3 \end{cases}$

⇐ احتمال تعیین درست $\frac{1}{2}$ است.

به طور مثال اگر مهاجم 01 را ببیند یکی از حالات $\begin{cases} k_2 \text{ تحت } 0 \\ k_3 \text{ تحت } 1 \end{cases}$ بوده است.

⇐ در حالت اول باید 00 و در دومی باید 11 بفرستد در نتیجه، احتمال $\frac{1}{2}$ موفق می‌شود. □

۴- (13.5) از آن جایی که کلید تصادفی داریم (K)، وقتی فرستنده ۲ بار یک پیام، مقادیر فرستاده شده یکسان نخواهد بود. در نتیجه باید مطمئن شد که کلید k کاملاً تصادفی باشد که مهاجم نتواند آن را حدس بزند برای این می‌توان مثلاً با PRNG تولید کنیم.

ادامه صفحه بعد

← در مراحل اصلی الگوریتم حتی چیدست کلید با استفاده از قدرت قابل پیش بینی از

چند امضا می‌توانند الگوریتم DSA را شکنند ☒

۵- (13.6) a) خروجی الگوریتم فرینه q است:

$g^q \overset{p}{\equiv} 1 \Leftarrow \begin{cases} g \text{ از مرتبه } 1 \text{ است} \\ \text{یا} \\ q \text{ " " } q \end{cases}$ (زیرا q اول است)

حال از صورت سؤال می‌بینیم $O(g)=q \Leftarrow g$ نمی‌تواند برابر قدر g باشد.

$O(g)=q \Rightarrow g^q \overset{p}{\equiv} 1 \Rightarrow (g^\alpha)^q \overset{p}{\equiv} 1 \xrightarrow{q \text{ اول است}} 1<\alpha<q-1$

بنابراین تمام q تا $q-1$ هم می‌توانند برای g انتخاب شود. بدلیل اول بودن q

$\Rightarrow g = \begin{cases} \text{تمام اعضای } q \text{ تا } 1 \text{ تا } q-1 \\ q \text{ هیچ عدد دیگری وجود ندارد.} \end{cases}$

b) $g^q \overset{p}{\equiv} (h^{\frac{p-1}{q}})^q \overset{p}{\equiv} h^{p-1} \overset{p}{\equiv} 1$

← خروجی الگوریتم مرتبه ۲ است؛ ثابت الف g از مرتبه q است.

$\Rightarrow g = q^\alpha \bmod p \quad 1 \le \alpha \le q-1$

اگر $q=O(q)$

ادامه صفحه بعد

ج) درجه‌ی کلی برای انتخاب α ، به اندازه $\varphi(q)$ حالت داریم؛ زیرا ←

همانطور که در طرف دیدیم اگر q اول باشد ، اندازه $q-1$ انتخاب داریم

حال اگر اول نباشد $(\alpha, q) = d \quad ; d > 1$

$$(g^{\alpha})^{k'p} \equiv (g^{k q})^{k'p} = (g^{q})^{k k' p} \equiv 1^{k k' p} \equiv 1 \quad \begin{cases} \alpha = kd \\ q = k'd \end{cases}$$

پس q کوچکترین توان نخواهد بود و درنتیجه g از q کمتر است $o(g) = \varphi(q)$

← چون q اول است $p-1$ توان ممکن برای توان g ممکن است پس

احتمال پیدا شدن توان g و در پی آن g در هر بار برابر $\frac{q-1}{p-1}$ است ←

تعداد تکرار $= \left\lfloor \frac{p-1}{q-1} \right\rfloor = 155$ ▨

d)

تکرار برای یافتن g لازم داریم ← تعداد تکرار $= \frac{p-1}{q-1} = \frac{2^{512}}{2^{128}} = 2^{384}$

⇐ پس در این مورد الگوریتم ۷ مناسب نیست ▨

e) برای الگوریتم ۲ باید $h^{\frac{p-1}{q}} = 1$ باشد تا جواب درست در تکرار اول

یعنی $o(h) \mid \frac{q-1}{p}$ باشد. حال چون تعداد مقدار ممکن برای g ، $\varphi(q)$ است،

برای همه مقسوم علیه‌های $\frac{q-1}{p}$ باید q اویلر آن را حساب کنیم تا تعداد hها را بیابیم.

ادامه صفحه بعد

تعداد h = (تعداد d که $\frac{p-1}{q}$ را عاد می‌کند) = $\frac{p-1}{d}$

⇒ احتمال موفقیت الگوریتم P_2 در مرحله اول $= 1 - \frac{\frac{p-1}{q}}{p-1} = 1 - \frac{1}{q} = 0.998$

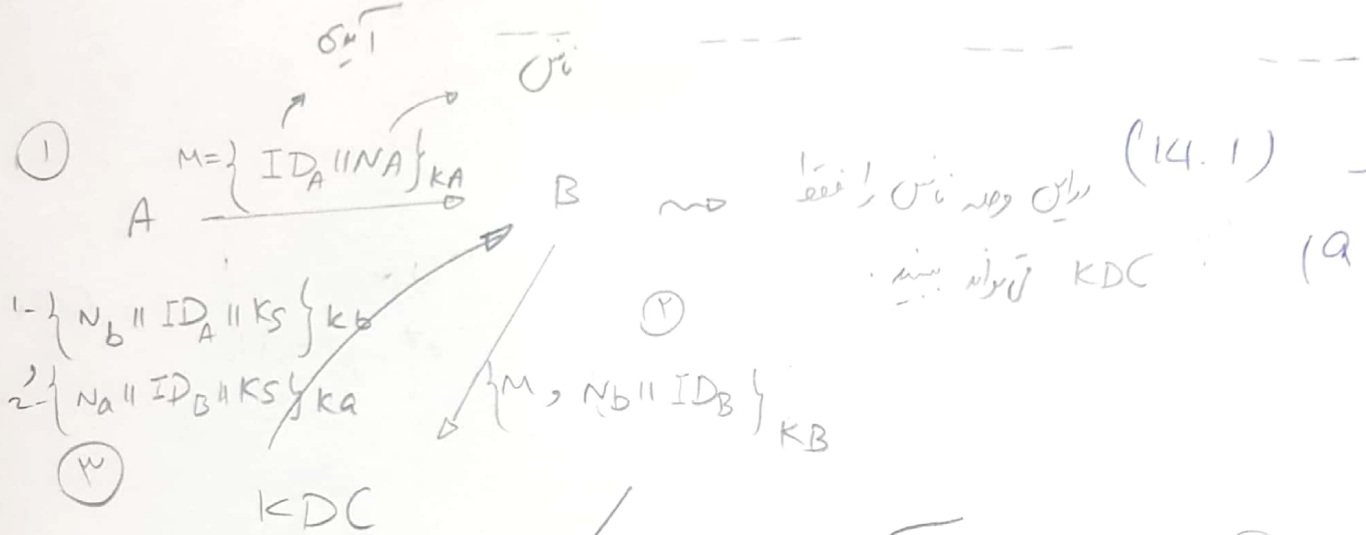
۶- (13.8)

الف) برای هر بیت پیام باید کلید متناظر فرستاده شده با آن را در نظر بگیریم. هر بیت پیام را کنار کلید متناظر قرار می‌دهیم و از آن یک هش (Hash) می‌گیریم. اگر Hash با محتویات متناظر پیام برابر بود ($V_i' \to V_i$ ، $1 \to V_i$) یعنی بیت مورد نظر درست ارسال شده است.

ب) چون V_i و V_i' مشخص هستند و Hash یک طرفه است، محاسبه Private key ممکن نیست ← در نتیجه اصل این روش امن است.

ج) این شیوه تنها برای گیرنده اول مناسب است زیرا با هر بار ارسال پیام نصف کلید به صورت خصوصی آشکار می‌شود و در ارسال‌های بعدی گیرنده واقعی می‌تواند پیام را بیابد زیرا برای هر بیت کلیدی وجود دارد و در گیرنده اول مشخص می‌شود؛ پس در مرحله‌ی بعد با کپی کردن امضا، اگر کلید تکراری باشد، که این بیت پیام همان پیام قبلی است و اگر کلید فرق کند حداقل می‌فهمیم این بیت با بیت متناظر قبلی متفاوت است.

d) برای هر سه پیام باید اعضای چندتایی تشکیل دهیم؛ با توجه به این که هر این الگوریتم یک تکرار است؛ این دسته نیاز به اعضای بزرگی دارد که پیاده سازی آن به صرفه نیست



در وضعه ③ با پیام اول، B آیدی A و نانس را باز می‌کند و از صحت K_S مطمئن می‌شود.

و پیام دیگری را برای A فرستاده می‌کند، A از صحت کلید و از اصالت B مطمئن می‌شود.

④

b) هر دو الگوریتم شبیه همدیگر اند؛ اما در الگوریتم 14.18، A در هر صورت به KDC پیام می‌دهد و کلید تولید می‌شود (حتی اگر B تأیید نکند) که این امر به صرفه نیست و سربار می‌اضافی دارد ولی در الگوریتم اول، B می‌تواند پیام A را در صورت داده، رد کند (قبل از ارتباط با KDC). بنابراین به صرفه تر است

۸- (14.4) در پوشه ارسالی، ۳ تصویر اول مربوط به root certificate کامپیوتر شخصی من است و ۲ تصویر دوم Intermediate Certificate سایت Digikala.com است.

(14.6) 9

(a

- می‌دانیم N_A با K_{AB} رمزنگاری می‌شود و A پیش از دریافت K_{AB} آن را رمزنگاری کرده و از B اطمینان می‌یابد سپس A از توزیع کلید K_{AB} اطمینان می‌یابد.
- همچنین N_A ارسال شده توسط A و رمزشده با K'_{AB} در پیام سوم به B می‌رسد. ✓
اگر B نتواند K_{AB} را N_A رمزنگاری کند. این کار فقط با K_{AB} ممکن است پس B هم از A اطمینان می‌یابد و به طور معادل K_{AB} توزیع شده است ✓
- N_A در پیام دوم می‌یابد و از تولید K'_{AB} بعد از ارسال N_A مطمئن می‌شود. بنابراین مطمئن می‌شود K_{AB} جدید باشد.
- بدیهتا چون B خودش توانست K'_{AB} را تولید کند، از جدید بودن آن اطمینان دارد ✓

(b

ترتیب داریم:

1. $A \xrightarrow{M=\{N_A, ID_A\}} B$
2. $B \xrightarrow{M} C$
3. $C \xrightarrow{ID_B, N_A} A$
4. A $\Rightarrow$ کلید K_{AB} را تولید می‌کند
5. $A \xrightarrow{E(K_{AB}, N_A \| K'_{AB})} C$
6. $C \xrightarrow{M'=\{N_A, K' \{K_{AB}} A$ که کلید تولید می‌کند
7. $C \xrightarrow{M'} A$

که چون C به K_{AB} دسترسی ندارد همان پیام M' را دوباره به A می‌فرستد

8. A گیج می‌کند، B در ارتباط است
9. $A \xrightarrow{E(K'_{AB}, N_A)} C$
10. C به K'_{AB} دسترسی ندارد و از A زیب دارد! ✓

ج) کافی است ثابت کنیم $\overline{B} \rightarrow \overline{A}$. ابتدا اجازه اصالت دوطرفه صورت گیرد

تا هنگام ارائه کار الگوریتم، هنگامی که A دنیایی از جانب ظاهر B روایت کرد

که فرستنده و گیرنده یکی نداند، می فهمند $\overline{B}$ آن را فرستاده است ✓