سوال ۱)

۱.۱

Protocol A: $y = e_{k_1}[x \,||h(k_2 \,||x)]$

- محاسبه ی $x||h = e_{k1}^{-1}(y)$
- محاسبه ی $h' = H(k_2 \,||x)$
- اگر $h = h'$ شود پیام معتبر است؛ در غیر این صورت یا پیام یا MAC (و یا هر دو) در حین انتقال دست‌کاری شده‌اند.

۱.۲

Protocol B: $y = e_k[x \,||\, sig_{k_{pr}} h(x)]$

- محاسبه ی $x||s = e_{k1}^{-1}(y)$ و بدست آوردن مقدار s
- محاسبه ی $h' = H(x)$
- بررسی اعتبار امضا به کمک $ver_{k_{pub}}(s, H(x))$.

سوال ۲)

۲.۱

$$c_i = z_i \oplus \{x_1 x_2 \cdots x_n || H_1(x) H_2(x) \cdots H_m(x)\} \qquad i = 1,2,\cdots,n+m$$

با فرض این‌که x تعداد n بیت دارد، اسکار ابتدا مقدار زیر را محاسبه می‌کند:

$$z_i = x_i \oplus c_i \qquad i = 1,2,\cdots,n$$

با توجه به این‌که اسکار مقدار x را می‌داند، $H(x)$ را محاسبه می‌کند؛ با فرض این‌که $H(x)$ دارای m بیت خروجی است، اسکار مقدار زیر را محاسبه می‌کند:

$$z_{j+n} = H_j(x) \oplus c_{j+n} \qquad j = 1,2,\cdots,m$$

سپس اسکار مقدار $H(x')$ را بدست می‌آورد و در انتها مقادیر زیر را محاسبه می‌کند:

$$c'_i = z_i \oplus x'_i \qquad i = 1,2,\cdots,n$$

$$c'_{j+n} = z_{j+n} \oplus H_j(x') \qquad j = 1,2,\cdots,m$$

با توجه به این‌که مهاجم کلید را با استفاده از plaintext و ciphertext بدست می‌آورد، این حمله در صورت استفاده از OTP نیز قابل اجرا است. البته با توجه به این‌که کلید هر سری متفاوت است؛ مهاجم برای هر پیام رد و بدل شده باید همه‌ی مراحل بالا را انجام دهد.

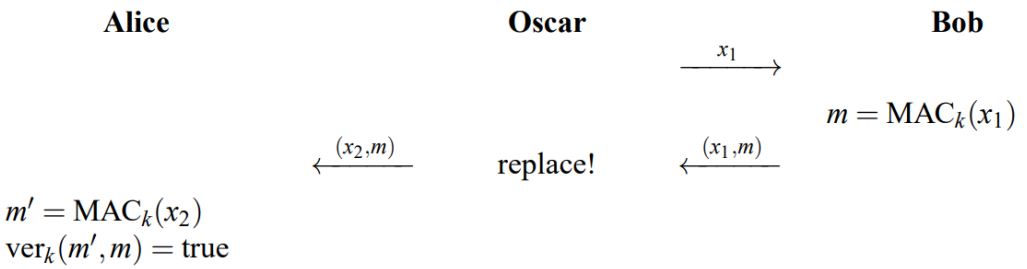
۲.۲

خیر، با توجه به این‌که اسکار مقادیر $z_1, z_2, \cdots, z_n$ را می‌تواند بازیابی کند ولی قادر به بازیابی رشته بیت $z_{n+1}, z_{n+2}, \cdots, z_{n+m}$ که برای رمز کردن $MAC_{k_2}(x)$ استفاده می‌شود، نیست. حتی اگر کل رشته بیت را بداند؛ باز هم به دلیل این‌که مقدار k_2 را نمی‌داند، قادر به محاسبه $MAC_{k_2}(x')$ نیست.

سوال ۳)

۳.۱

در این‌جا فرض می‌شود که اسکار می‌تواند باب را فریب دهد تا پیام x_1 را امضا کند.



۳.۲

برای ایجاد کالیژن، اسکار باید $\sqrt{2^n}$ مک را محاسبه کند. با توجه به این‌که اسکار مقدار کلید مخفی را نمی‌داند، باید به نحوی آلیس و یا باب را فریب دهد تا مقدار MAC را برای تعداد زیادی پیام محاسبه کند، که در عمل امکان‌پذیر نیست. از طرف دیگر، کالیژن در تابع هش توسط اسکار و بدون کمک آلیس و باب می‌تواند ایجاد شود، زیرا محاسبات بدون استفاده از کلید انجام می‌گیرد.

بنابراین یک تابع مک ۸۰ بیتی، امنیت 2^{80} را ایجاد می‌کند، زیرا حملات کالیژن قابل اجرا نیستند و یک تابع هش با اندازه خروجی یکسان، تنها امنیت در حدود 2^{40} را به وجود می‌آورد.

سوال ۴)

۴.۱

۱. کلید جلسه ها توسط یک عملیات خطی و معکوس کلید جلسه قبلی تولید می‌شوند.

۲. استفاده از توابع $hash$ ← وابستگی غیرخطی و غیرمعکوس کلید جلسه ها

۳. استفاده از کلید اصلی و کلید جلسه قبلی برای ایجاد کلید جلسه ی بعدی

۴.۲

روش b, c . چرا که کلید جلسه های قدیمی را نمی‌توان از کلید جلسه های اخیر استخراج کرد.

۴.۳

۱. هرجلسه، از آن‌جایی که PFS وجود ندارد.

۲. همه ی جلسه هایی که از کلید جلسه ی K_n استفاده می‌کنند و همه‌ی جلسه های بعدی.

۳. تنها جلسه ی اخیر. از آن‌جایی که از کلید اصلی که نا معلوم است، برای تولید کلیدهای بعدی استفاده می‌شود.

۴.۴

همه‌ی کلیدها قابل محاسبه هستند.

سوال ۵)

۵.۱

$$t = 10^6\ bit/sec$$

$$storage = t \cdot r = 2h \cdot 10^6 = 2 \cdot 3600 \cdot 10^6 = 7.2\ Gbit = 0.9\ GByte$$

بنابراین ذخیره‌سازی کمتر از ۱ گیگابایت را می‌توان با هزینه‌های کم انجام داد.

۵.۲

تعداد کلیدهایی که یک مهاجم در طی ۳۰ روز می‌تواند به آن‌ها دست پیدا کند، برابر است با:

$$keys = \frac{30\ days}{10\ min} = \frac{30 \cdot 24 \cdot 60}{10} = 4320$$

مدت زمان استخراج هر کلید برابر است با:

$$T_{key\ derive} = \frac{2h}{4320} = \frac{2 \cdot 3600}{4320} = 1.67\ sec$$

با توجه به این‌که توابع هش سریع هستند، بنابراین عملیات استخراج کلید می‌تواند به راحتی و با این سرعت انجام گیرد.

سوال ۶)

۶.۱

$$Oscar$$

$$O = 2^{16} \ \ mod\ 467 = 156$$

$$K_{AO} = A^{o} \ \ mod\ p = 394^{16} \ \ mod\ 467 = 243$$

$$K_{BO} = B^{o} \ \ mod\ p = 313^{16} \ \ mod\ 467 = 438$$

۶.۲

$$Alice$$

$$A = 2^{228} \ \ mod\ 467 = 394$$

$$K_{AO} = O^{a} \ \ mod\ p = 156^{228} \ \ mod\ 467 = 243$$

$$Bob$$

$$B = 2^{57} \ \ mod\ 467 = 313$$

$$K_{BO} = O^{b} \ \ mod\ p = 156^{57} \ \ mod\ 467 = 438$$

سوال ۷)

امضای CA صرفا کلید عمومی یک کاربر را پوشش می‌دهد. حتی اگر تمامی پارامترهای الگوریتم امضا اعلام شود، کلید خصوصی به دلیل مسئله لگاریتم گسسته همچنان غیرقابل محاسبه است. پس اسکار نمی‌تواند کلیدهای جلسه‌ای که قبل از تشخیص کلید و الگوریتم امضای CA استفاده شده‌است را محاسبه کند. ولی اسکار اکنون می‌تواند با ارائه گواهی‌های جعلی، خودش را به عنوان هر کاربر دلخواه معرفی کند.