سوال ۱)

پارادوکس تولد در رمزنگاری به احتمال بالای وقوع تصادم‌ها (یعنی دو ورودی متفاوت که هش خروجی یکسان تولید می‌کنند) اشاره دارد، زمانی که طول خروجی هش به اندازه کافی بزرگ نباشد. برای یک تابع هش با n بیت، احتمال یافتن یک تصادم به طور قابل توجهی افزایش می‌یابد وقتی که $2^{n/2}$ هش محاسبه شده باشد.

برای توابع هش، داشتن طول خروجی بزرگ‌تر (مثلاً ۱۶۰ بیت) برای به حداقل رساندن ریسک تصادم‌ها حیاتی است و اطمینان حاصل می‌کند که تابع هش در برابر چنین حملاتی مقاوم است. یک تابع هش ۱۶۰ بیتی نیاز به 2^{80} هش برای داشتن احتمال قابل توجهی از تصادم دارد که از نظر محاسباتی غیر ممکن است.

چرا طول خروجی‌های کوتاه‌تر (به عنوان مثال، ۸۰ بیت) برای MAC ها کافی است؟

۱. محرمانه بودن کلید:

امنیت MAC بر اساس کلید مخفی k استوار است. بدون دانستن کلید، یک مهاجم نمی‌تواند به سادگی MAC های معتبر تولید کند، حتی اگر بتواند جفت‌های زیادی از پیام و MAC را مشاهده کند.

۲. پیچیدگی حمله:

مهاجم (اسکار) که قصد جعل یک MAC معتبر را دارد، باید یک جفت پیام و MAC معتبر $(x, MAC_k(x))$ را پیدا کند. با یک MAC ۸۰ بیتی، مهاجم نیاز به انجام 2^{80} عملیات برای پیدا کردن یک جفت معتبر دارد که از نظر محاسباتی غیر ممکن است.

۳. مقاومت در برابر حمله Brute Force:

یک MAC ۸۰ بیتی همچنان در برابر حمله جست و جوی فراگیر مقاوم است، زیرا تلاش برای حدس زدن MAC صحیح نیازمند 2^{80} تلاش است. با توجه به ماهیت نمایی عملیات‌های مورد نیاز، این سطح از تلاش برای مهاجمان غیر عملی است.

تحلیل حمله:

هدف اسکار:

اسکار نیاز دارد تا یک MAC معتبر برای یک پیام x بدون دانستن کلید k جعل کند.

مراحل حمله:

۱. Brute Force Attack : اسکار می‌تواند هر کلید ممکن k را امتحان کند تا MAC صحیح را برای یک پیام داده شده پیدا کند. با این حال، با یک MAC ۸۰ بیتی، این نیازمند 2^{80} تلاش است که غیر عملی است.

۲. Collision Attack : اسکار می‌تواند تلاش کند تا دو پیام متفاوت x و x' پیدا کند که $MAC_k(x) = MAC_k(x')$ به دلیل محرمانه بودن کلید و غیر عملی بودن انجام 2^{80} عملیات، این حمله نیز عملی نیست.

سوال 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.

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)

در صورت‌مسئله مقادیر زیر داده شده‌اند:

- $p = 61$ و $\alpha = 18$ برای الگوریتم Diffie-hellman
- $p' = 467$، $d' = 127$، $\alpha' = 2$ و $\beta = 2^{127} \bmod 467 = 132$ برای الگوریتم Elgamal Signature
- سه کلید خصوصی برای کاربران:

  $a = 11$ برای **Alice**، $B = 22$ برای **Bob**، $C = 33$ برای **Charley**

- شناسه کاربران:

$ID(C) = 3, ID(B) = 2, ID(A) = 1$

۱. محاسبه کلیدهای عمومی

با استفاده از رابطه $X = \alpha^x \bmod p$ کلیدهای عمومی به‌دست می‌آیند:

- برای Alice: $A = 18^{11} \bmod 61 = 10$
- برای Bob: $B = 18^{11} \bmod 61 = 39$
- برای Charley: $C = 18^{11} \bmod 61 = 24$

۲. محاسبه مقادیر x_i

برای هر کاربر مقدار x_i با استفاده از رابطه روبرو محاسبه می‌شود: $x_i = 4 * b_i + ID(i)$

- $x_A = 4 \times 11 + 1 = 45$
- $x_B = 4 \times 22 + 2 = 90$
- $x_C = 4 \times 33 + 3 = 135$

۳. امضای دیجیتال با الگوریتم Elgamal

برای هر کاربر، امضا (r,s) با استفاده از پارامترهای زیر محاسبه می‌شود:

- برای Alice $kE = 213$
- برای Bob $kE = 215$
- برای Charley $kE = 217$

فرمول‌های امضا:

- $r = \alpha^{kE} \bmod p$
- $s = (x - d \cdot r)\, k_E^{-1} \bmod p-1$

محاسبات:

- برای Alice: $s = 112, r = 29$
- برای Bob: $s = 218, r = 116$
- برای Charley: $s = 165, r = 464$

۵. تشکیل گواهی‌ها

گواهی هر کاربر به‌صورت زیر است:

- **Alice**: $CertA = (A, ID(A), (r, s)) = (10, 1, (29, 112))$

- **Bob**: CertB = (B, ID(B), (r, s)) = (39, 2, (116, 218))
- **Charley**: CertC = (C, ID(C), (r, s)) = (24, 3, (464, 165))

۶. تأیید گواهی‌ها

برای تأیید صحت امضای دیجیتال، باید شرط زیر برقرار باشد.

$\alpha^{x'} \bmod p \equiv \beta^{r} \cdot r^{s} \bmod p$

- برای Alice:

$(2^{45} \bmod 467) = (132^{29} \cdot 29^{112} \bmod 467) = 80$

- برای Bob:

$(2^{90} \bmod 467) = (132^{116} \cdot 116^{218} \bmod 467) = 329$

- برای Charley:

$(2^{135} \bmod 467) = (132^{464} \cdot 464^{165} \bmod 467) = 168$

۷. محاسبه session key ها

طبق فرمول $A^{b} \bmod p$ محاسبه می‌شود.

- کلید مشترک Alice و Bob:

$10^{22} \bmod 61 = 39^{11} \bmod 61 = 19$

- کلید مشترک Bob و Charley:

$39^{33} \bmod 61 = 24^{22} \bmod 61 = 27$

- کلید مشترک Alice و Charley:

$10^{33} \bmod 61 = 24^{11} \bmod 61 = 37$

سوال 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

۲.

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

۳.

۱. هرجلسه، از آن‌جایی که PSF 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.

همه‌ی کلیدها قابل محاسبه هستند.

سوال ۵)

۱ .فرایند صدور گواهی(Certificate Issuing)

باب برای دریافت گواهی از مرکز صدور گواهی (CA) اطلاعات زیر را ارسال می‌کند:

- شناسه باب ID(B)
- کلید عمومی باب B

اقدام اسکار:

- اسکار که کنترل کامل بر ارتباطات باب دارد، کلید عمومی CA را با کلید عمومی خودش جایگزین می‌کند.
- وقتی باب گواهی دریافت می‌کند، به دلیل نداشتن روشی برای تأیید اعتبار کلید عمومی، تصور می‌کند که کلید عمومی دریافتی همان کلید معتبر CA است.
- اسکار گواهی تقلبی صادر می‌کند که کلید عمومی B را تأیید می‌کند اما امضا شده با کلید خصوصی $k_{pr,Oscar}$ خود اسکار است.

۲ .اجرای پروتکل(Protocol Execution)

برای برقراری یک نشست امن بین باب و اسکار، اسکار باید طوری رفتار کند که باب فکر کند در حال ارتباط با آلیس است.

مراحل اقدام اسکار:

- اسکار با استفاده از پروتکل تبادل کلید دیفی-هلمن، گواهی جعلی خود را برای باب ارسال می‌کند.

- اسکار ابتدا یک جفت کلید خصوصی و عمومی ($k_{pr,Oscar}$, $k_{pub,Oscar}$) ایجاد می‌کند.
- مقدار B را به جای کلید عمومی آلیس ارسال می‌کند.
- باب مقدار A (کلید عمومی اسکار که به‌عنوان آلیس جعل شده) را دریافت می‌کند و فکر می‌کند که این کلید مربوط به آلیس است.
- با این کار، اسکار می‌تواند کلید جلسه k_{AB} را به اشتراک بگذارد و به اطلاعات رمزنگاری‌شده دسترسی پیدا کند.

در نتیجه اسکار با جایگزینی کلید عمومی CA و جعل هویت آلیس، می‌تواند گواهی‌ها را دستکاری کرده و یک نشست امن بین خود و باب برقرار کند، بدون اینکه باب متوجه تقلب شود.