به نام خدا

«امنیت داده و شبکه»

دکتر امینی

تمرین دوم

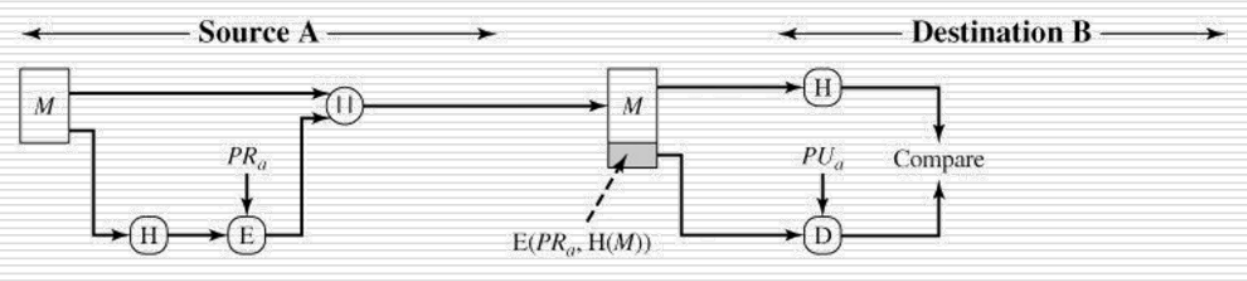
علیرضا دهقانپور فراشاه

۹۸۱۰۱۵۵۵

سوال یک

الف) کد احراز صحت پیام یا MAC به این علت که با یک کلید عمومی میان دو طرف ایجاد می‌شود خاصیت عدم انکارپذیری را ندارد و دو طرف با همان کلید مشترک قابلیت ایجاد MAC را دارند.

ب) در شکل زیر hash پیام به کمک کلید خصوصی فرستنده رمز می‌شود و سمت گیرنده با رمزگشایی MAC به کمک کلید عمومی فرستنده و مقایسه‌ی hash پیام با مقدار رمزگشایی شده می‌تواند از اصالت فرستنده مطمئن شود.



ج) رمزهای نامتقارن به دو هدف ایجاد شدند. یک ایجاد امضای دیجیتال و دیگری حل مشکل توزیع کلید عمومی بین فرستنده و گیرنده. همچنین برای ایجاد ارتباط میان n نفر به n(n-1)/2 کلید نیاز است در روش متقارن.

مقایسه رمز متقارن و نامتقارن:

- کاربردی: از این نظر رمز نامتقارن بیشتر به عنوان راه حلی برای مسئله‌ی توزیع کلید متقارن استفاده می‌شود. می‌توان با آن امضای دیجیتال داشت و مشکل مدیریت کلیدها را نیز ندارد.
- سرعت: رمز متقارن از عملیات‌های جابجایی و جایگزینی استفاده می‌کرد ولی رمز نامتقارن بر اساس عملیات توان‌رسانی است که سنگین‌تر است و بنابراین کندتر از رمزهای متقارن است.
- امنیت: از نظر امنیت هر دو روش امن هستند و امنیت آن‌ها وابسته به طول کلید و توابع مورد استفاده در آن‌ها است.

د) در مورد اول هم محرمانگی را داریم هم صحت پیام زیرا مهاجم پیام را نمی‌تواند ببیند و با تغییر آن نمی‌تواند MAC را تغییر دهد.

در مورد دوم اما محرمانگی پیام حفظ می‌شود ولی صحت آن خیر زیرا مهاجم می‌تواند پیام رمز شده را تغییر دهد و از آن hash بگیرد و کنار پیام خراب شده قرار دهد. دیگر گیرنده نمی‌تواند تشخیص دهد که پیام رمزشده دستکاری شده‌است یا خیر.

سوال دو

الف) خیر مهاجم با داشتن تابع H به راحتی می‌تواند یک بلاک داده جدید به پیام اضافه کند و c را با بلاک جدید به تابع h بدهد و خروجی چکیده ساز را خروجی تابع h قرار دهد. یعنی این روش در مقابل حمله‌ی افزایش طول ضعف دارد.

$$C_{n+1} = H(k||m_1, m_2, ..., m_{n+1}) = h(C_n, m_{n+1})$$

ب) خیر امن نخواهد بود زیرا اگر مهاجم بتواند یک تصادم برای m بیابد یعنی یک پیام دیگری مانند 'm وجود دارد که hash آن‌ها در در مرحله‌ی آخر یکی می‌شود. سپس چون خروجی بلاک یکی مانده به آخر که مربوط یکی شده است hash آن به همراه کلید نیز یکسان خواهد بود یعنی c برای دو پیام m و 'm یکسان خواهد بود. یعنی مهاجم با حدس تصادم می‌تواند پیام 'm را به جای m بدهد و خروجی MAC آن‌ها یکسان خواهد بود.

$$H(m) = H(m')$$

$$H(m||k) = h(H(m), k) = h(H(m'), k) = H(m'||k)$$

سوال سه

الف)

- محرمانگی: این روش محرمانگی را حفظ می‌کند زیرا پیام با کلید مشترک رمزشده است و همچنین در قسمت دوم متن ارسالی نیز پیام به همراه رمزشده‌اش به همراه کلید به یک تابع چکیده‌ساز داده شده‌است که برگشت ناپذیر است. بنابراین بدون داشتن کلید نمی‌توان به متن پیام دست یافت.
- صحت پیام: این ویژگی را نیز دارا است. زیرا اگر متن رمز شده تغییر کند دیگر خروجی بخش دوم که حاصل اجرای تابع چکیده‌ساز بر روی متن رمز شده به همراه متن اصلی و کلید است دیگر سازگار نخواهد بود. احتمال اینکه بتوان خروجی تابع چکیده‌ساز و متن رمزشده را به گونه‌ای تغییر داد که سازگار باشند بسیار کم است بنابراین در این مورد نیز امن است.
- عدم انکارپذیری: این ویژگی را به دلیل وجود کلید مشترک نخواهد داشت.

ب) اگر فرض کنیم h در برابر تصادم مقاوم نباشد، یعنی می‌توان مقداری را یافت که خروجی آن با خروجی حاصل از پیام و رمزشده‌ی پیام و کلید یکسان است ولی نمی‌توان از این مقدار جدید استفاده کرد زیرا ما در بخش اول پیام فقط بخشی از ورودی تابع چکیده‌ساز که متن رمزشده است را می‌دهیم. یعنی با فرض اینکه تابع چکیده‌ساز در برابر تصادم مقاوم نیست نیز نمی‌توان متن رمزی پیدا کرد که خروجی تابع چکیده‌ساز برای این متن رمز و رمزگشایی شده‌ی آن و کلید یکسان باشد.

سوال چهارم

اگر فرض کنید 1n و 2n نسبت به هم اول نباشند یعنی یک عامل مشترک دارند که چون هر دو حاصل ضرب دو عدد اول هستند یعنی یکی از عوامل اول آن‌ها مشترک است.

$$n_1 = p_1 \times q_1$$
$$n_2 = p_2 \times q_1$$
$$gcd(n_1, n_2) \neq 1 \implies p_1 = p_2 \text{ or } q_1 = q_2$$

بدون کاستن از کلیت مسئله فرض می‌کنیم عامل مشترک 1p باشد بنابراین می‌توانیم با محاسبه‌ی gcd میان 1n و 2n با مرتبه‌ی زمانی log به مقدار 1p برسیم.

حال با تقسیم 1n و 2n به 1p می‌توانیم 2p و 2q را بدست آوریم و از آن‌ها خواهیم داشت:

$$\phi(n_1) = (p_1 - 1)(q_1 - 1)$$
$$\phi(n_2) = (p_1 - 1)(q_2 - 1)$$

با داشتن این دو مقدار و 1e و 2e یافتن 1d و 2d ممکن است.

$$d_1 = e_1^{-1} \mod \phi(n_1)$$
$$d_2 = e_2^{-1} \mod \phi(n_2)$$

سوال پنجم

بله امکان جعل امضا وجود دارد. در روابط زیر سمت چپ نیز mod n دارد ولی نوشته نشده‌است یعنی تمامی روابط در mod n است. داریم:

$$\left.\begin{array}{l} s_1' = m_1^d \mod n \\ s_2' = m_2^d \mod n \end{array}\right\} \implies \left\{\begin{array}{l} {s_1'}^e = m_1^{de} \mod n \\ {s_2'}^e = m_2^{de} \mod n \end{array}\right. \implies \left\{\begin{array}{l} {s_1'}^e = m_1^{de} \mod n = m_1 \\ {s_2'}^e = m_2^{de} \mod n = m_2 \end{array}\right.$$

پس بنابراین اگر 2m1m را پیام قرار دهیم و2s1's را به عنوان امضا می‌توانیم امضای بانک را جعل کنیم زیرا:

$$\begin{aligned} (s_1' s_2')^e \mod n = \\ &= (m_1 m_2)^{de} \mod n = \\ &= m_1^{de} m_2^{de} \mod n = \\ &= (m_1^{de} \mod n)(m_2^{de} \mod n) = \\ &= m_1 m_2 \end{aligned}$$

بنابراین این پیام توسط گیرنده تایید می‌شود.

سوال ششم
طبق تعارف صورت سوال داریم:

$$gcd(m', N) = 1$$
$$C' = m'^{e} \mod N$$
$$C'' = C'c \mod N$$

حال d را معکوس 'm تعریف می‌کنیم و ''C را می‌دهیم به ماشین تا رمزگشایی کند. خروجی ماشین را e می‌نامیم.
خواهیم داشت:

$$d = m'^{-1} \mod N \quad (۱)$$
$$C'^{d} \mod N = m'^{de} \mod N = m' \mod N \quad (۲)$$
$$e = (C'c)^{d} \mod N = C'^{d}c^{d} \mod N \quad (۳)$$

حال اگر ed را به پیمانه‌ی N محاسبه کنیم طبق ۲ و ۱ خواهیم داشت:

$$\begin{aligned} ed \mod N &= (c^{d} \mod N)(C'^{d} \mod N)(m'^{-1} \mod N) = \\ &= (c^{d} \mod N)(m' \mod N)(m'^{-1} \mod N) = \\ &= (c^{d} \mod N) \\ &= m \end{aligned}$$

سوال هفتم

ابتدا هر سه عضو بر عدد α و q توافق می‌کنند. سپس هر کدام برای خود مقادیر X_A, X_B, X_C را به صورت تصادفی مانند روش دیفی-هلمن عادی تولید می‌کنند.

سپس به صورت چرخشی ابتدا A به B مقدار $\alpha^{X_A} \ mod \ q$ و B به C مقدار $\alpha^{X_B} \ mod \ q$ و C به A مقدار $\alpha^{X_C} \ mod \ q$ را می‌فرستد.

حال A مقدار $Y_{CA} = \alpha^{X_C \times X_A} \ mod \ q$ را محاسبه می‌کند. به همین ترتیب B مقدار $Y_{AB} = \alpha^{X_A \times X_B} \ mod \ q$ و C مقدار $Y_{BC} = \alpha^{X_B \times X_C} \ mod \ q$ را محاسبه می‌کنند.

در این مرحله A به B مقدار Y_{CA} و B به C مقدار Y_{AB} و C به A مقدار Y_{BC} را ارسال می‌کند.

حال A با داشتن Y_{BC} می‌تواند مقدار ${Y_{BC}}^{X_A} \ mod \ q = \alpha^{X_A \times X_B \times X_C} \ mod \ q$ را می‌کند و این مقدار می‌شود کلید مشترک سه نفر.

سپس B نیز به همین ترتیب با داشتن Y_{CA} می‌تواند مقدار ${Y_{CA}}^{X_B} \ mod \ q = \alpha^{X_A \times X_B \times X_C} \ mod \ q$ را می‌کند.

در نهایت C نیز مقدار ${Y_{AB}}^{X_C} \ mod \ q = \alpha^{X_A \times X_B \times X_C} \ mod \ q$ را محاسبه می‌کند.

در نهایت $\alpha^{X_A \times X_B \times X_C} \ mod \ q$ کلید مشترک هر سه نفر است.

سوال هشتم

الف)

رمز الجمل بر پایه‌ی سختی مسئله‌ی لگاریتم گسسته است ولی RSA بر پایه‌ی سختی تجزیه اعداد بزرگ است. از RSA برای رمزگذاری و رمزگشایی، امضای رقمی و توزیع کلید استفاده می‌شود ولی از الجمل برای رمزگذاری و رمزگشایی و توزیع کلید استفاده می‌شود.

مهم‌ترین برتری الجمل نسبت به RSA وجود عامل تصادفی در رمزنگاری است که سبب می‌شود یک پیام ثابت در هر بار رمزنگاری به یک عبارت متفاوت رمز شود.

به طور کلی رمز RSA سریع‌تر از الجمل است زیرا محاسبات در RSA حجم کمتری دارد. همچنین برای داشتن یک سطح برابر امنیتی معمولا طول کلید RSA باید بیشتر از الجمل باشد.

معمولا برای زمانی که حجم داده ارسالی کم باشد مانند توزیع کلید از الجمل استفاده می‌شود و زمانی که داده بزرگ باشد از رمز RSA استفاده می‌شود.

ب)

این روش در مقابل حمله‌ی Chosen Cipher ضعف دارد. فرض کنید (2C1, C) رمزشده‌ی پیام m باشد و بخواهیم آن را رمزگشایی کنیم. پارامترهای عمومی α, q هستند و کلید خصوصی X_A و کلید عمومی α^{X_A} است.

داریم:

$$(C_1, C_2) = (\alpha^r \mod q, m \times \alpha^{X_A r} \mod q)$$

حال r' و m' را تصادفی انتخاب می‌کنیم و پیام $(C1', C2') = (C1 * \alpha^{r'} mod\ q, C2 * m' * \alpha^{X_A r'})$ را به سیستم رمزگشایی می‌دهیم. داریم:

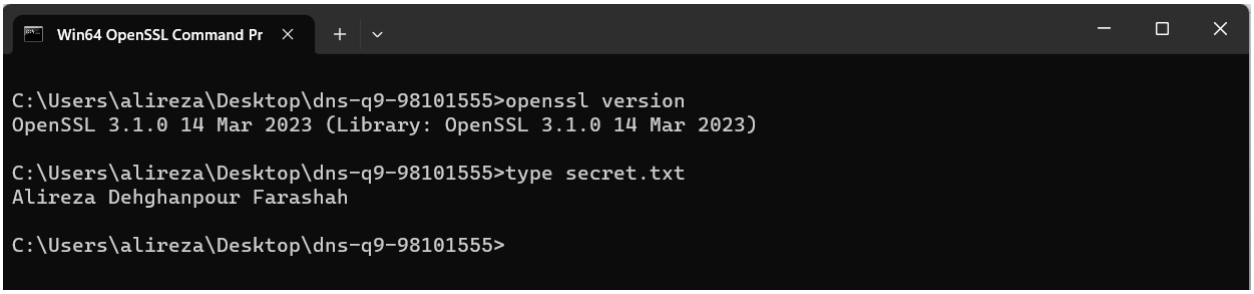
$$\begin{aligned}(C_1', C_2') = \\ &= (\alpha^r \alpha^{r'} \mod q, mm' \times \alpha^{X_A r} \alpha^{X_A r'} \mod q) = \\ &= (\alpha^{r+r'} \mod q, mm' \times \alpha^{X_A (r+r')} \mod q)\end{aligned}$$

بنابراین خروجی سیستم رمزگشایی برابر $(m * m')\ mod\ q$ خواهد شد و اگر خروجی را در $m'^{-1}\ mod\ q$ ضرب کنیم به $m\ mod\ q$ که پیام است خواهیم رسید.

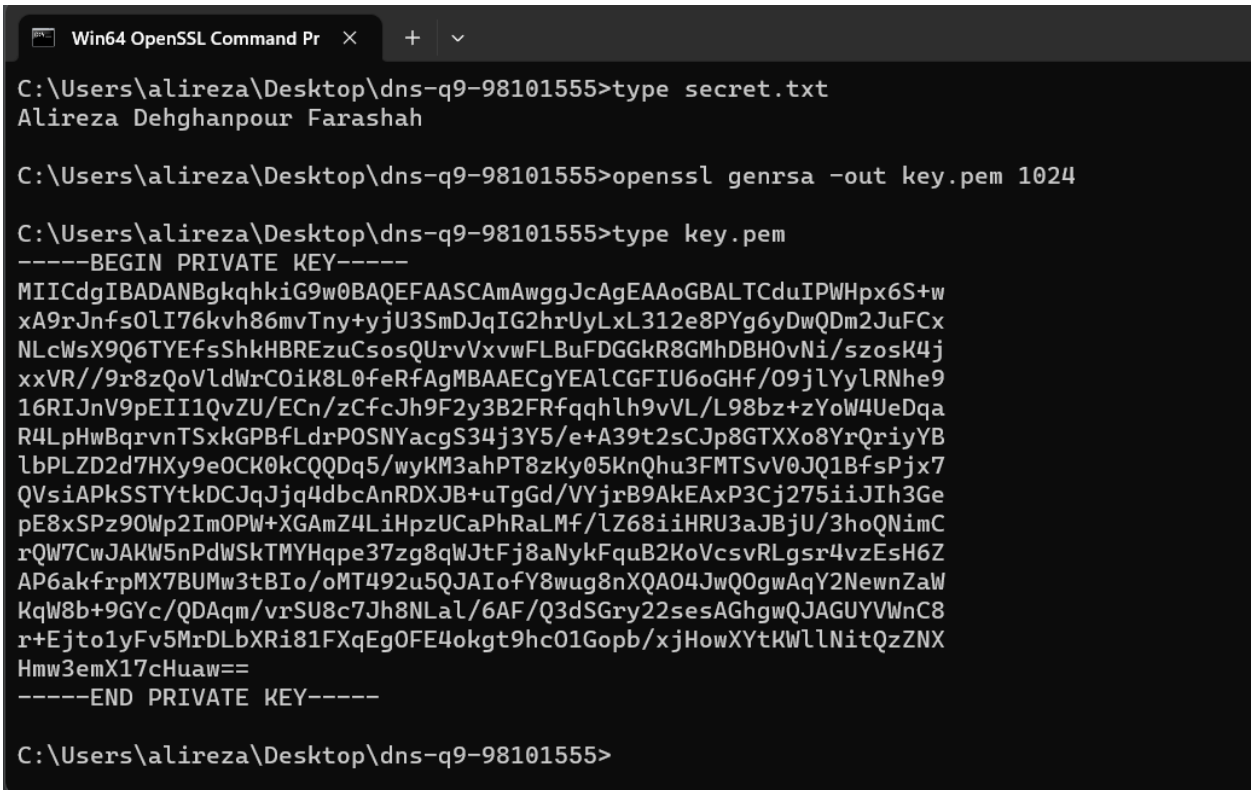
بنابراین این رمزگذاری در برابر Chosen Cipher آسیب‌پذیر است.

سوال نهم

در ابتدا از نصب شدن openssl مطمئن می‌شویم و سپس محتویات فایل secret.txt را مشاهده می‌کنیم.

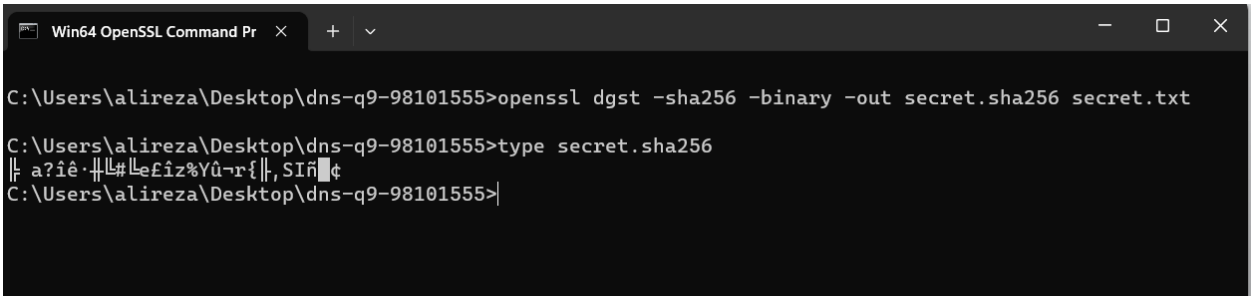


حال کلید خصوصی را تولید می‌کنیم.

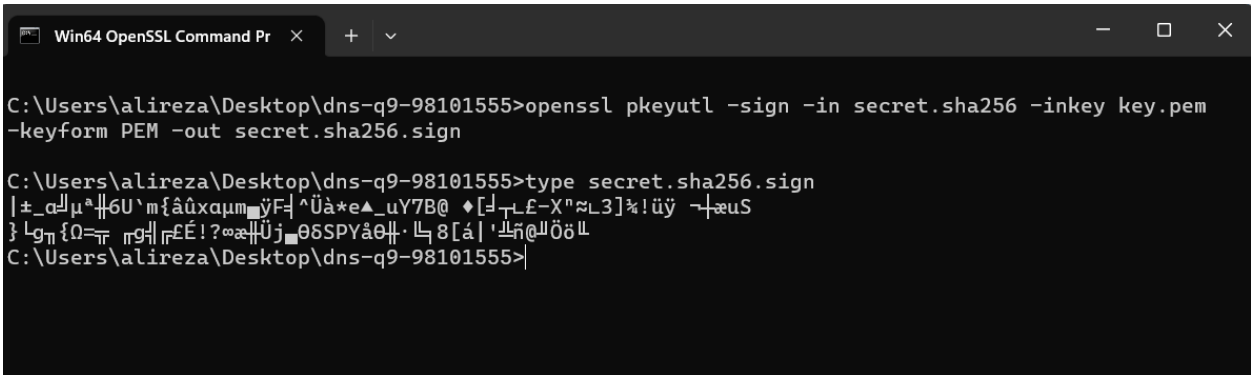


سپس کلید عمومی را تولید می‌کنیم.

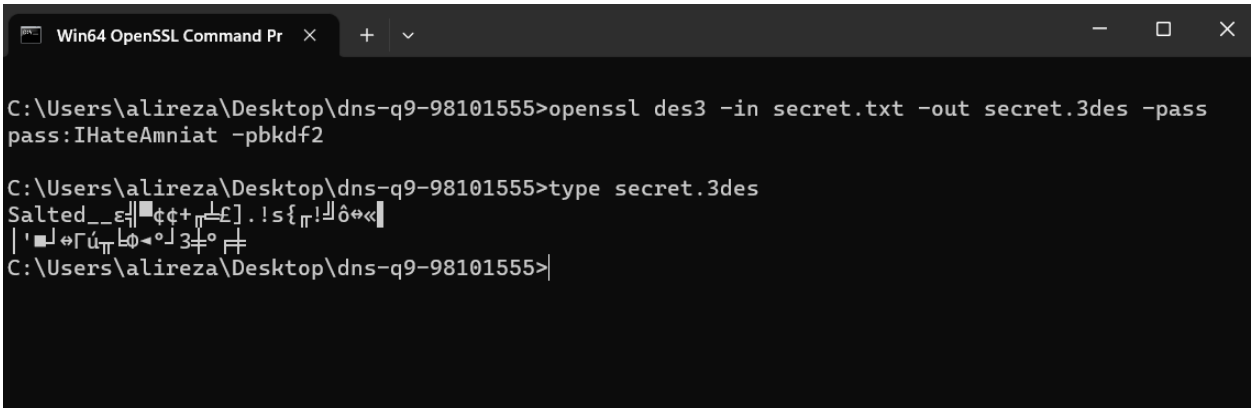
حال باید از فایل secret.txt با الگوریتم SHA-256 چکیده بگیریم.



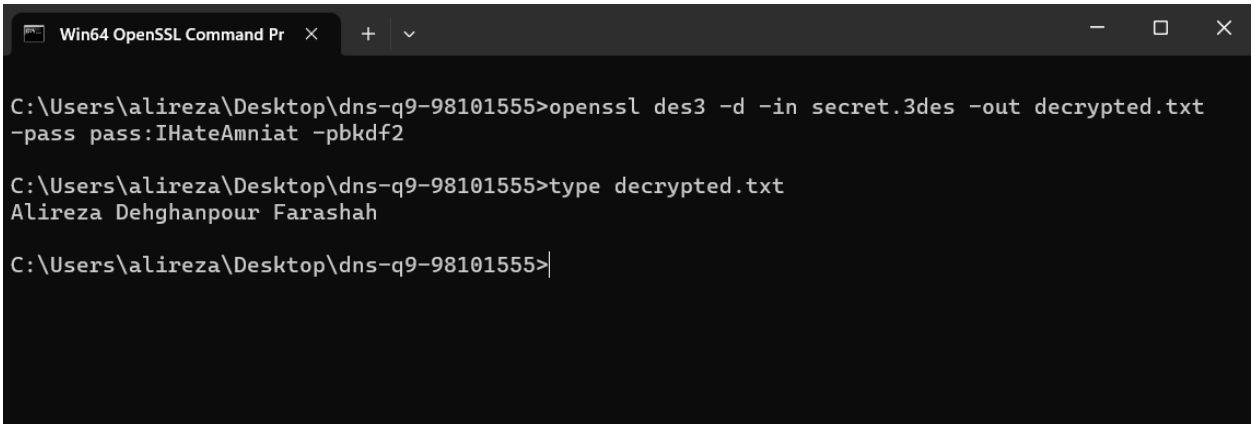
حال باید با کلید خصوصی آن را امضا کنیم. برای امضا دو دستور rsautl و pkeyutl وجود دارد که در داکیومنت openssl آورده شده است که دستور اول در حال منسوخ شدن است. از دستور دوم برای اینکار استفاده می‌کنیم.



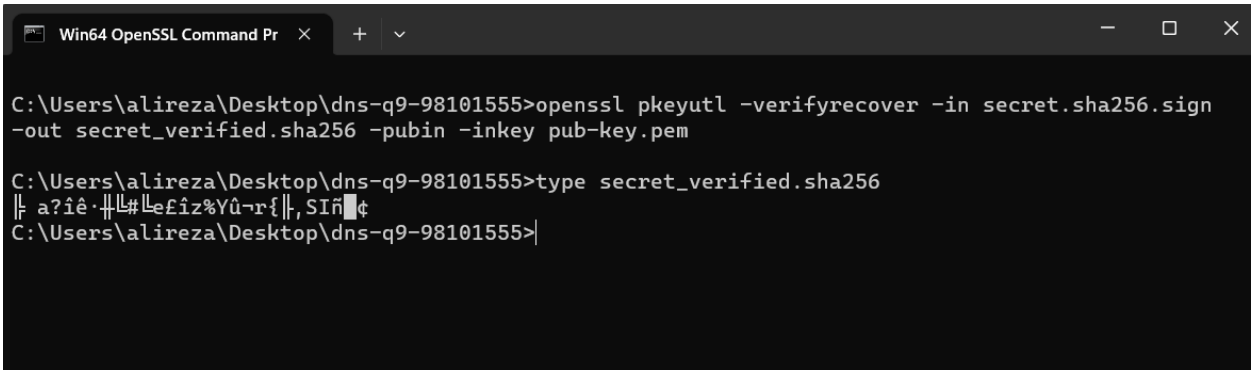
حال با الگوریتم DES3 فایل را رمز می‌کنیم.



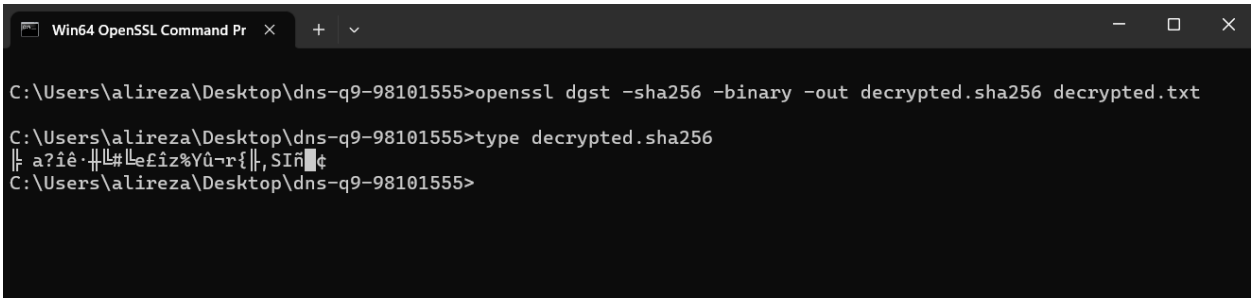
حال فایل رمز شده را رمزگشایی می‌کنیم.



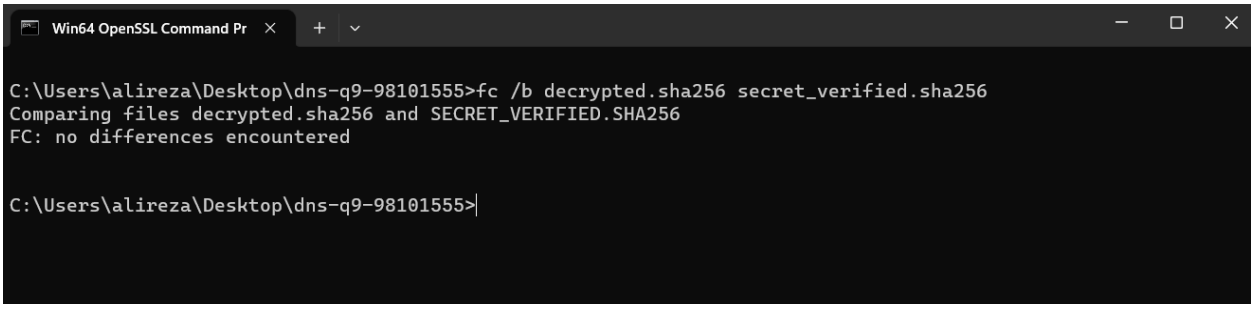
حال باید با کلید عمومی امضا را verify کنیم.



حال از فایل رمزگشایی شده چکیده می‌گیریم.



حال در مود باینری دو فایل 256decrypted.sha و 256secret_verified.sha را مقایسه می‌کنیم و متوجه می‌شویم که یکسان هستند.



منابع

https://pagefault.blog/2019/04/22/how-to-sign-and-verify-using-openssl/

https://opensource.com/article/19/6/cryptography-basics-openssl-part-2

https://www.openssl.org/docs/man3.0/man1/openssl-rsautl.html

https://www.openssl.org/docs/man3.0/man1/openssl-pkeyutl.html