

Encryption and Decryption

**Mohammadrafie Fattahinia
Mahdi Asadian**

از کجا به کجا

در چند دهه اخیر به دلیل پیشرفت تکنولوژی و رواج استفاده از سیستم های کامپیوتری و در پی آن گسترش شبکه های ارتباطاتی و اینترنتی ، یکی از مهمترین دغدغه های کاربران و ارائه دهندگان این حوزه ، افزایش امنیت در انتقال داده ها و حفظ حریم شخصی بوده و خواهد بود . و اینگونه شد که مفاهیمی مانند امنیت شبکه ، رمزگذاری و رمزگاری ، هک و نفوذ و دیگر مفاهیم مربوط به امنیت این سیستم ها به وجود امد.

Canva

رمزگذاری ، قدیمی تر از تصور

شاید خیلی ها فکر کنند که مفهوم رمزگذاری در همین چند دهه اخیر به دلیل گسترش دستگاه های ارتباطی معنا پیدا کرده است در حالی که واقعیت جور دیگریست.

در واقع این مفهوم به هزاران سال پیش باز میگردد . از همان زمان که انسان ها به جای اینکه حرفشان را مستقیم بزنند ، از شکل ها و رمز هایی استفاده کردند که جز کسانی که از آنها مطلع اند، از این حرف ها با خبر نشوند.

در ادامه انسان ها با ساختن دستگاه ها و متدهای رمزگذاری مختلف توانستند امنیت پیام ها ، فرمول ها ، داده ها و خود را در برابر بیگانگان افزایش دهند.

به عنوان مثال در یونان باستان ، از دستگاهی به نام اسکایتیل (scytale) برای رمزگذاری و رمز نگاری استفاده میکردند. سازوکار این دستگاه این گونه است که نواری از جنس پارچه یا چرم را به دور استوانه ای به قطری مشخص می پیچیدند و بعد از نوشتن متن، آنرا باز میکردند، اینگونه حروف به هم میریخت و فقط با پیچیدن دوباره‌ی آن به استوانه ای با همان قطر ، متن درست پدیدار میشد.

این دستگاه تنها مثال کوچکی از چندین هزار دستگاه و متدها رمزگاریست.
یکی دیگه از روش‌های قدیمی و معروف رمزگذاری، رمزگذاری سزار است به این‌گونه که هر حرف الفبا را به حرفی که چند جایگاه جلوتر یا عقب تر قرار دارد تبدیل میکردند و این‌گونه پیام رمزگذاری میشد.

مثلی دیگر از این دستگاه‌ها که در بحبوحه جنگ جهانی اتفاق افتاد، دستگاه رمزگذاری نازی‌ها به اسم ائیگما بود که برای انتقال دستورات و پیام‌ها استفاده میشد. با شکسته شدن رمز این دستگاه توسط آلن تورینگ و تیمش، مسیر جنگ به سود متفقین تغییر کرد.



Scytale



Enigma

امروز چه خبر

امروزه به لطف پیشرفت قدرت پردازشی کامپیوتر ها ، الگوریتم ها و سازوکار های بسیار پیچیده تر و ایمن تری در این حوزه استفاده میشود که شکستن آن به راحتی امکان پذیر نیست . البته باید در نظر داشت که دنیای نفوذ و رمزگاری از این قاعده استثناء نیست و پیشرفت های بسیار محسوسی داشته است
در دنیای امروز از انواع مختلفی رمزگذاری و رمزگاری استفاده میشود مانند :

(رمزگذاری متقارن) Symmetric Encryption

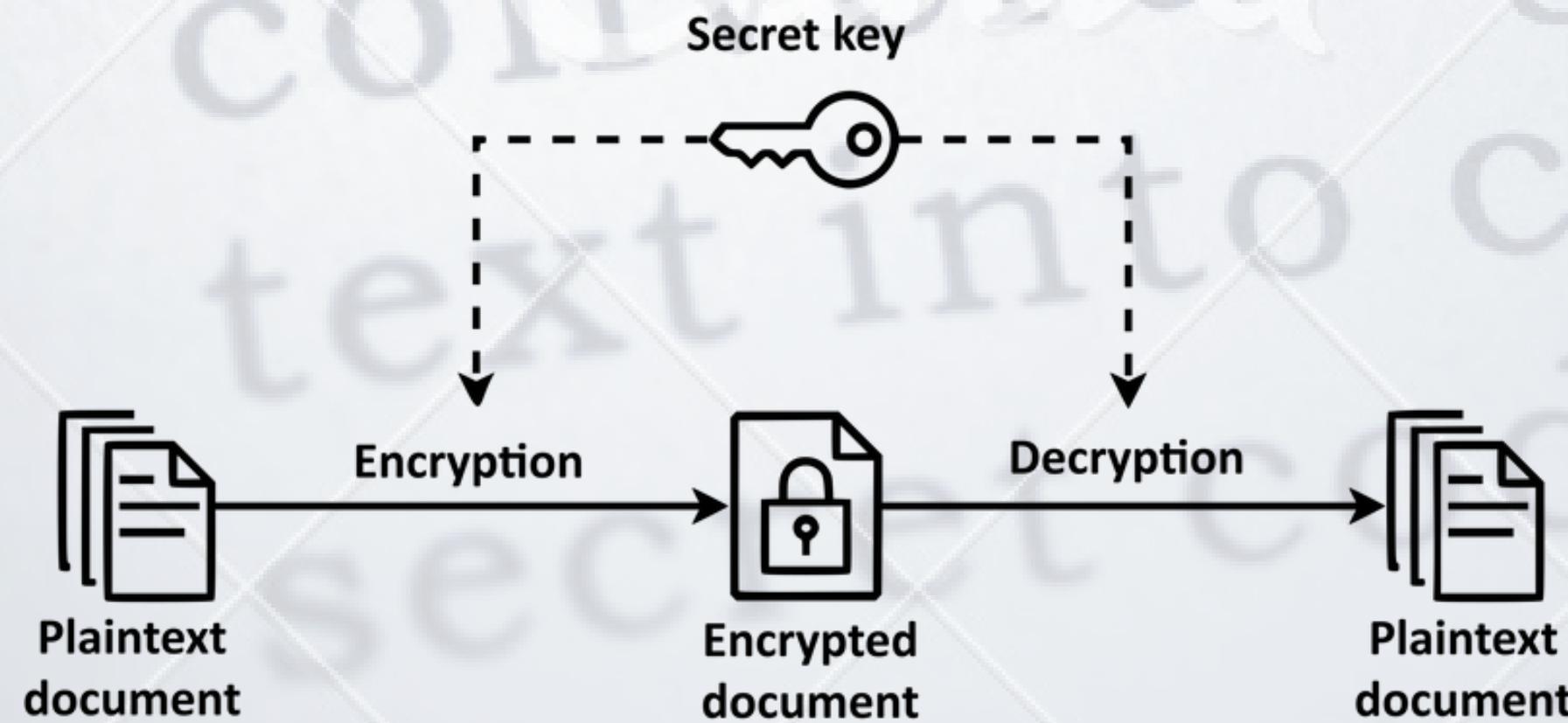
(رمزگذاری نامتقارن) Asymmetric Encryption

و انواعی دیگر از متدها و پروتکلها

رمزگذاری متقارن

این نوع سیستم ساز و کار پیچیده ای ندارد . سیستم مش این گونه است که داده ها با یکسری الگوریتم های خاص رمزگذاری شده و با تنها کلیدی که از آن به جا مانده است باز میشود . AES و Triple DES نمونه هایی از آن هستند.

چون توضیح این موضوع از اهداف این فایل نیست بیشتر از این به آن نمیپردازیم



رمزگذاری نا متقارن

در دنیای امروز ارتباطات به حدی گسترده شده که کار کردن طرفین فقط با یک کلید بسیار سخت به نظر میرسد.

هرچند رمزگذاری متقارن از کاربردهای خاص خود برخورد دار است اما گسترده‌گی شبکه و ارتباطات در دنیای امروزی به حدی زیاد شده که عملاً کار با یک کلید برای رمزگذاری و ارسال داده‌ها به آن طرف دنیا، امکان پذیر نیست.

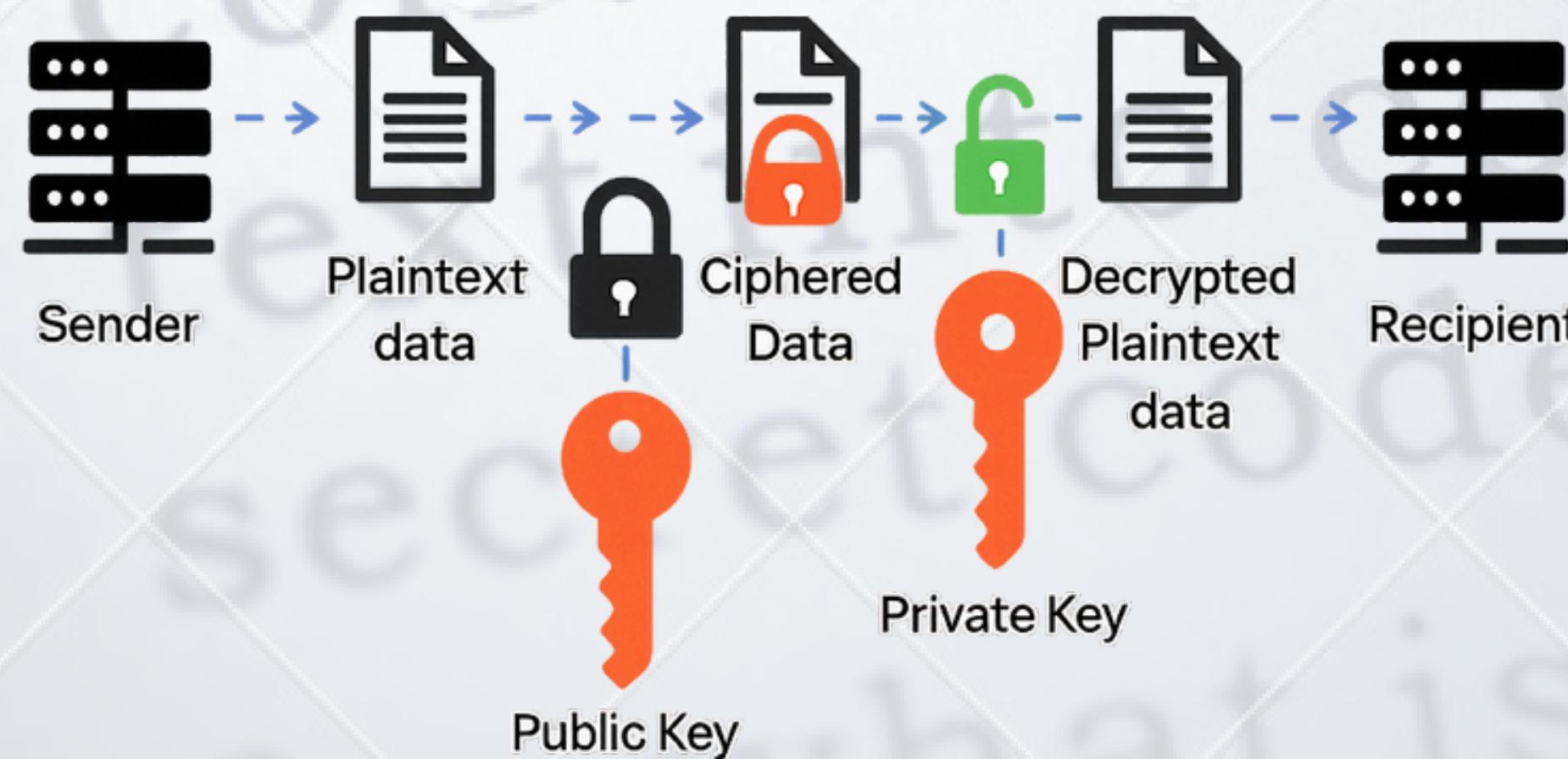
اینجا بود که متخصصین این حوزه چاره‌ای برای آن اندیشیدند و متدهایی مخصوص برای حل این مشکل بنا گذاشتند. این متدها پایه‌ای برای رمزگذاری عمومی و امضاهای دیجیتال شدند.

در این متدها، دیگر فقط با یک کلید سروکار نداریم بلکه با یک جفت کلید سروکله می‌زنیم که این دو کلید هم کار می‌کنند. اگر با یک کلید داده‌ای را رمزگذاری کنید فقط با کلید دیگر می‌توان آنرا باز کرد. ECC، RSA، DSA و از این نوع سازوکار پیروی می‌کنند که جلوتر RSA را بیشتر بررسی خواهیم کرد.

کلید عمومی-خصوصی

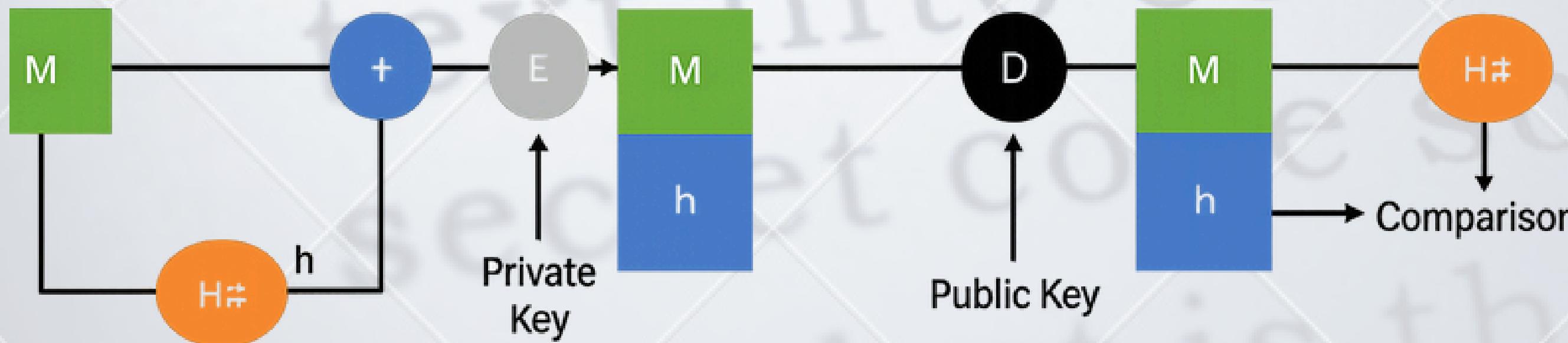
این ساز و کار متقارن پیچیده تر است. همانگونه که اشاره شد این ساز و کار از الگوریتم ها و فرمول های ریاضیاتی پیشرفته ای برای تولید دو عدد (کلید) که برعکس هم کار میکنند استفاده میکند.

یکی از این کلید ها به صورت عمومی منتشر شده و کلید دیگر به صورت خصوصی دست کاربر اصلی میماند. هر کس که بخواهد داده ای برای این کاربر بفرستد ، ابتدا با کلید عمومی مربوط به کاربر که منتشر شده است داده را رمزگذاری کرده و برای کاربر اصلی میفرستد و این داده رمزگذاری شده فقط و فقط با کلید خصوصی در دست کاربر اصلی باز میشود. البته توجه داشته باشید که منظور از کاربر صرفا شخص نیست و میتواند سازمان ها و سرور های مختلفی باشند.



امضای دیجیتال

ساز و کار امضا دیجیتال نیز از کلید عمومی-خصوصی ولی با تفاوت های کوچک استفاده میکند. به این گونه که ابتدا هش اسناد را محاسبه کرده و کنار آن قرار میدهند. سپس با کلید خصوصی آن را قفل میکنند. دریافت کننده بعد از باز کردن آن با کلید عمومی مربوط به فرستنده متوجه اصلی بودن آن میشود زیرا تنها زمانی این اسناد با کلید عمومی فرستنده باز میشود که فقط با کلید خصوصی آن فرستنده قفل شده باشد. همچنین دریافت کننده با محاسبه هش اسناد و مقایسه هی آن با هش اصلی ، متوجه دست نخورده بودن آن اسناد میشود.



سیستم RSA

این سیستم یک نمونه از رمزگذاری و رمزنگاری عمومیست که در سال 1977 توسط رون ریوست، آدی شامیر و لئونارد ادلمن معرفی شد. نام این سیستم نیز از حروف اول نام های آنها گرفته شده است.

این سیستم بر اساس حدس دو عدد اول بزرگ (1024 یا 2048 بیتی) و ضرب آنها کار میکند. دلیل آن این است که پیدا کردن ضرب دو عدد اول بزرگ، با الگوریتم ها و ازمون های شناخته شده نسبتا ساده و دست یافتنیست اما تجزیه حاصل ضرب آن دو به اعداد اول مخصوصا در مرتبه ای به آن بزرگی برای کامپیوتر های فعلی عمل ناشدندی است. برای همین یک طرفه بودن عملیات است که عمل شکستن رمز آن کاری به شدت دشوار و یا حتی ناشدنیست. البته برخی متدها برای شکستن آن پیاده سازی شده که جلوتر به آن خواهیم پرداخت.

بعد از پیدا کردن دو عدد اول بزرگ و انجام یکسری عملیات ها و الگوریتم ها، داده ما رمزگذاری و کلید های ما به دست می آیند.

مراحل رمزگذاری RSA

1. (p, q) دو عدد اول بزرگ به دست اوردن

2. $n = p * q$ محاسبه‌ی

3. $\phi(n) = (p - 1) * (q - 1)$ محاسبه‌ی

4. $\text{gcd}(\phi(n), e) = 1$ پیدا کردن e به گونه‌ای که

5. (d) $\phi(n)$ پیمانه‌ای e نسبت به پیدا کردن معکوس

6. $\text{cipher} = \text{message}^e \pmod{n}$

7. $\text{message} = \text{cipher}^d \pmod{n}$

e=>public key and d=>private key

به دست اوردن عدد اول بزرگ

همانطور که میدانید هزاران سال این سوال که ایا رابطه ای منطقی در توزیع اعداد اول وجود دارد یا نه بی پاسخ مانده است. حدس هایی من باب این موضوع بیان شده اما هیچکدام به اثبات نرسیده است.

پس هیچ راهی برای حدس مستقیم اعداد اول وجود ندارد و باید غیر مستقیم به آن ها رسید. در روش غیر مستقیم، ابتدا عدد فرد و بزرگی را حدس میزنیم و سپس با ازمون ها و الگوریتم های ریاضی اول بودن یا نبودن آنرا پیشیبینی میکنیم. بعضی از این ازمون ها مانند AKS مطلقا اول بودن یا نبودن آن عدد را مشخص میکنند اما بسیار هزینه‌ی زمانی و پردازشی زیادی دارند. برخی دیگر از این ازمون ها اول بودن یا نبودن اعداد را با درصد خطای بسیار کم پیشیبینی میکنند اما هزینه‌ی کم این ازمون ها، آن را نسبت به خطای آنها ارزنده تر و استفاده از آن هارا عملی تر میکنند. در ادامه یکی از این ازمون ها را بررسی خواهیم کرد.

Miller-Rabin Test

این ازمون در ابتدا یک ازمون قطعی برپایه‌ی فرضیه‌ی اثبات نشده‌ی ریمان تعمیم یافته بود که در سال 1976 توسط میلر معرفی شد. حدوداً چهار سال بعد مایکل راین شکلی امروزی از این قضیه را ارائه داد که هرچند نتیجه‌ی آن قطعی نبود ولی مستقل از حدس ریمان عمل می‌کرد. اینگونه بود که شکل امروزی این قضیه پدید امد. در ادامه به شرح آن خواهیم پرداخت.

از آنجا که اثبات این قضیه نیازمند اشنایی با یکسری از مفاهیم ریاضیاتی مانند قضیه میدان‌ها، قضیه کوچک فرما و دیگر مفاهیم ریاضیاتی است، از اثبات آن در این فایل می‌گذریم. برای اطلاع بیشتر از اثبات این روش به Number Theory and Cryptography نوشته‌ی Neal Koblitz مراجعه کنید.

فرض کنید n که عددی فرد است را حدس زدیم.
در مرحله‌ی اول $1 - n$ را به صورت ضرب یک عدد فرد در دو به توان s مینویسیم :

$$n - 1 = 2^s * d \rightarrow \text{فرد}$$

Python Code:

```
s = 0  
d = n - 1  
while d % 2 == 0:  
    d //= 2  
    s += 1
```

$$1 < a < n - 1$$

$$x = a^d \pmod{n}$$

در مرحله‌ی بعد یک عدد ماین ۱ تا $n - 1$ به عنوان پایه انتخاب می‌کنیم:
سپس باید شرط اولیه‌ای را چک کنیم:

اگر x برابر با $1 \pmod{n}$ شد ازمون موفق بوده و عدد ما **احتمالاً** اول است.

$$x = x^2 \pmod{n}$$

اگر این اتفاق نیوفتاد برای $1 - s$ بار مرحله‌ی رو به رو را تکرار می‌کنیم:
در هر مرحله:

اگر x برابر با $1 \pmod{n}$ شد ازمون موفق بوده و عدد ما **احتمالاً** اول است.

در غیر این صورت اگر در $1 - s$ مرحله شرط بالا برقرار نشد ازمون ناموفق بوده و عدد ما **حتماً** مرکب است.

اگر n مرکب باشد تقریباً سه چهارم پایه‌ها در بازه‌ی مشخص شده می‌توانند مرکب بودن آن را برملا کنند پس احتمال خطا در هر بار تست یک پایه، 0.25 است. در نتیجه اگر ازمون را با k بار تکرار با پایه‌های مختلف تست کنیم، خطای ازمون به $P = (0.25)^k$ کاهش پیدا می‌کند. در نتیجه هر بار ازمون را با پایه‌های بیشتری تست کنیم، احتمال خطا کمتر می‌شود.

از آنجا که انتخاب پایه را چهل بار تکرار کردیم ، خطای ما به $P = (0.25)^{40}$ کاهش پیدا میکند

```
k = 40
for _ in range(k):
    a = random.randint(2, n - 2)
    x = pow(a, d, n)

    if x == 1 or x == n - 1:
        continue

    for _ in range(s - 1):
        x = pow(x, 2, n)
        if x == n - 1:
            break
    else:
        return False

return True
```

انتخاب پایه $x = a^d \pmod{n}$

شرط اولیه

ایجاد شرط ثانویه و چک کردن $x = x^2 \pmod{n}$ برای ۱ - s بار

بعد از پیدا کردن p و q و گذراندن ازمون میلر رایین ، نوبت به محاسبه $\phi(n)$ و پیدا کردن کلید ها (e, d) میرسد:

$$\phi(n) = (p - 1) * (q - 1)$$

حال باید e را به گونه ای انتخاب کنیم که $\gcd(\phi(n), e) = 1$.
ابتدا به صورت پیشفرض عدد 65537 را تست میکنیم که در کتابخانه های مختلف به عنوان عدد استاندارد برای e استفاده میکند.

دلیلش این است که این عدد یک عدد فرماست و به فرم $(s=2^n+1)$ نوشته میشود. این فرم در اعداد دودویی و در کامپیوتر به صورت 100...001...1 در می آید که محاسبات و هزینه هی زمانی و پردازشی را به ویژه در اعداد بزرگ کاهش میدهد.
اگر این عدد در شرط ما صدق نکرد ، سراغ اعداد کوچکتر فرما میرویم و اگر انها هم صدق نکردند ، به سراغ عددی که در شرط صدق کند میرویم.

ابتدا تست اعداد فرما و عدد استاندارد 65537

```
for candidate in [65537, 257, 17, 5, 3]: → 65537
    if candidate < phi_n and math.gcd(candidate, phi_n) == 1:
        return candidate

while True:
    candidate = random.randint(7, phi_n - 1)
    if math.gcd(candidate, phi_n) == 1:
        return candidate
```

پیدا کردن اولین عددی که صدق کند
درصورتی که اعداد فرما جواب ندهند

حال بعد از پیدا کردن e نوبت به پیدا کردن معکوس پیمانه ای آن در پیمانه $\phi(n)$ یعنی d میرسیم برای پیدا کردن d از روش اقلیدوس معکوس استفاده میکنیم.

روش اقلیدوس معمولی برای پیدا کردن \gcd دو عدد استفاده میشود و در این روش انقدر طرفین ب m را از هم باقی مانده میگیریم تا به صفر برسیم. اخرين باقى مانده اى غير صفر ما در واقع ب m ماست برای مثال $\gcd(36, 56)$ را محاسبه میکنیم:

$$1. 56 = 36 * 1 + 20$$

$$2. 36 = 20 * 1 + 16$$

$$3. 20 = 16 * 1 + 4$$

$$4. 16 = 4 * 4 + 0$$

آخرین باقى مانده غير صفر و جواب ما

حال به سراغ تعریف معکوس پیمانه ای من رویم :

$$d \mid e^*d \pmod{\phi(n)} = 1$$

معکوس پیمانه ای e در پیمانه $\phi(n)$ به صورت رو به رو تعریف میشود:

حال که با الگوریتم اقلیدوس و تعریف معکوس پیمانه ای اشنا شدیم ، سراغ معکوس آن میرویم از آنجا که می توان ب م هر دو عدد را به صورت ترکیبی خطی از آن دو عدد نوشت :

$$\gcd(e, \phi(n)) = e^*d + \phi(n)*k$$

اگر از رابطه i بالا از طرفین همنهشتی به پیمانه $\phi(n)$ بگیریم دقیقا به رابطه i اول صفحه میرسیم.

از آنجا که $1 = e^*d + \phi(n)*k$ پس کافی است که جواب معادله i را به دست اوریم.

پس ابتدا به روش اقلیدوسی به یک که ب م e و $\phi(n)$ است میرسیم سپس یک را از انتهای بازنویسی میکنیم تا به ابتدا بررسیم :

فرض کنیم معکوس پیمانه ای 17 در پیمانه ی 3120 را میخواهیم به دست اوریم:

$$9 = 3120 - 17 * 183 \quad \leftarrow \quad 1. 3120 = 17 * 183 + 9$$

$$8 = 17 - 9 * 1 \quad \leftarrow \quad 2. 17 = 9 * 1 + 8$$

$$1 = 9 - 8 * 1 \quad \leftarrow \quad 3. 9 = 8 * 1 + 1 \quad \rightarrow \quad \text{ب} ۲۳$$

$$4. 8 = 8 * 1 + 0$$

حال به صورت عکس یک را ساخته و اعداد را جایگذاری میکنیم:

$$1. 1 = 9 - 8 * 1$$

$$2. 1 = 9 - (17 - 9 * 1) * 1 = 2 * 9 - 17$$

$$3. 1 = 2 * (3120 - 17 * 183) - 17$$

$$4. 1 = 2 * 3120 - 367 * 13$$

معکوس پیمانه ای ما که در پیمانه ی 3120 همان 2753 است

حال که کلید عمومی و کلید خصوصی e و d ساخته شد و میتوانیم با فرمول های زیر ، پیام خود را رمزگذاری و رمزنگاری کنیم:

$$\text{cipher} = \text{message}^e \pmod{n}$$

$$\text{message} = \text{cipher}^d \pmod{n}$$

از آنجا که اثبات این بخش نیازمند درک مفاهیم ریاضیاتی مانند قضیه ی کوچک فرما ، گزاره ی اویلر ، قضیه میدان ها ، قضیه بزو و دیگر مفاهیم ریاضیاتی است، از آن میگذریم.

برای اطلاع بیشتر از اثبات این روش به Cryptography نوشته ی Neal Koblitz مراجعه کنید.

منابع

1. Number Theory and Cryptography
2. Introduction to Algorithms(CLRS)

و همچین استفاده از ویکیپدیا و چت بات‌ها برای پیدا کردن اطلاعات عمومی و تاریخی این فایل

توجه شود که تمام کدهای این فایل از سورس کدهای پروژه استخراج شده است