RSA Şifreleme

2190656027 Aygül Koç
2160656048 Hasan Özer

RSA (Rivest, Shamir, Adleman)

- Bu algoritma, adını Ron Rivest, Adi Shamir ve Leonard Adleman adlı geliştiricilerinin soyadlarının ilk harflerinden almıştır.
- RSA, çok büyük sayıları asal çarpanlarına ayırmanın zorluğuna dayanan asimetrik (açık anahtarlı) şifreleme algoritmasıdır.
- Herkese açık bir açık anahtar, sadece mesajın alıcısında bulunan ve paylaşılmayan bir gizli anahtar kullanılır.
- Kullanılacak bu iki farklı anahtar asal sayılar arasından seçilerek oluşturulur. Bir RSA kullanıcısı seçilen iki büyük asal sayının çarpımını üretir ve seçtiği başka bir değerle beraber ortak anahtar olarak ilan eder. Seçilmiş olan asal çarpanları ise saklar.

RSA (Rivest, Shamir, Adleman)

- Açık anahtarlı bir şifreleme sisteminde, şifreleme anahtarı herkese açıktır ve gizli tutulan (özel) şifre çözme anahtarından farklıdır. Bir RSA kullanıcısı, bir yardımcı değerle(public exponent **e**) birlikte iki büyük asal sayının çarpımını içeren bir açık anahtar oluşturur ve yayınlar. Asal sayılar gizli tutulur, yayınlanmaz. Mesajlar, genel anahtar aracılığıyla herkes tarafından şifrelenebilir, ancak yalnızca gizli anahtara sahip olan biri tarafından çözülebilir.

AVANTAJLARI

- Asimetrik şifreleme algoritması olarak simetrik anahtar dağıtımı sorununu çözebilir.
- Asimetrik şifreleme algoritması olarak RSA Kimliklendirme, dijital imza gibi ihtiyaçlara yanıt verebilir.
- Herhangi bir simetrik şifreleme algoritmasına göre daha güvenlidir.
- Asimetrik şifreleme algoritması olarak RSA, simetrik şifreleme algoritmalarının çözemediği özgünlük ve gizlilik zayıflığı gibi konuları çözebilir.

DEZAVANTAJLARI

- İşlem gücü maliyeti yüksektir. (Özellikle anahtar oluşum esnasında)
- Eğer ki RSA ile şifrelenen veri bir ağ üzerinden transfer ediliyorsa bant genişliğinin fazlaca tüketilmesine sebebiyet verir.
- Şifrelenebilecek mesaj uzunluğu kısıtlıdır. Uygulanan PKCS#1 standardı versiyonuna göre ve anahtar boyutuna göre mesaj uzunluğu farklılık göstermektedir. Çözüm olarak mesajlar parçalar halinde şifrelenerek yollanabilir fakat bu durum büyük verilerde maliyetli olacağı için simetrik şifreleme ile kombine edilmiş olarak kullanılması daha uygundur.

PKCS#1

- Açık anahtarlı şifreleme için RSA algoritmasının uygulanmasına ilişkin temel tanımları ve önerileri sağlar.
- Çeşitli doldurma şemaları içerir. En güncel ve güvenilir doldurma şemasından OAEP(Optimal asymmetric encryption padding) olarak bahsedilir.
- En güncel versiyonu 2.2'dir. Detaylar hakkında rfc8017 dökümanı incelenebilir.

PKCS#1 Versiyon 1.5 Şeması

- Eski ve güvenli kabul edilmeyen fakat hala kullanılmaya devam edilen bir versiyon olan PKCS#1 v1.5 şeması :

ASN.1 (Abstract Syntax Notation One)

- Abstract Syntax Notation One (ASN.1), platformlar arası bir şekilde serileştirilebilen ve serileştirmesi kaldırılabilen veri yapılarını tanımlamak için standart bir arayüz tanımlama dilidir. Genel olarak telekomünikasyonda, bilgisayar ağlarında özellikle de kriptografide kullanılır.
- DER, XER, PER gibi formatlarda ASN.1 ile paketlenmiş veriler saklanır.
- RSA açık ve gizli anahtarlarındaki n,e,d,p,q... gibi değerler DER formatında onaltılık tabanda ASN.1 ile paketlenerek saklanırlar.

AÇIK VE GİZLİ ANAHTAR ÇİFTİ OLUŞTURMA

- **p** ve **q** gibi çok büyük iki asal sayı seçilir.
- Güvenlik amacıyla **p** ve **q** sayıları rastgele seçilmeli ve yakın uzunlukta olmalıdırlar. Bu sayılar asallık testi kullanılarak etkin bir şekilde bulunabilir.
- Bu iki asal sayının çarpımı **n** (modulus) = **p.q** hesaplanır.
- **p** ve **q** şeçilirken hedeflenen bit uzunluğu (modulus length) göz önünde bulundurulmalıdır.
- **n** özel ve ortak anahtarlar için mod değeri olarak kullanılacaktır.
- Bu sayıların totientı olan **ϕ(n)** = (**p**-1)(**q**-1) hesaplanır.
- 1 < **e** < **ϕ(n)** koşuluna uygun ve **ϕ(n)** ile en büyük ortak böleni 1 olan bir **e** sayısı seçilir.

AÇIK VE GİZLİ ANAHTAR ÇİFTİ OLUŞTURMA

- Bit uzunluğu kısa olan ve küçük Hamming Ağırlığına sahip **e** değerleri (yaygın olarak 0x10001 = 65,537) daha verimli şifreleme sağlarlar. Fakat küçük **e** değerleri (örneğin 3) bazı durumlarda güvenliği azaltabilir.
- **e**'nin mod **ϕ(n)**'e göre tersini alıp **d** sayısına ulaşılır.
- Diğer bir deyişle **d**.**e** ≡ 1 (mod**ϕ(n)**) koşulunu sağlayan **d** değeri hesaplanır. Modular aritmetik'te çarpımsal ters bulma işlemi yapılırken genelde **genişletilmiş öklid algoritması** kullanılır.

AÇIK VE GİZLİ ANAHTAR ÇİFTİ OLUŞTURMA

- Ayrıca **Çin Kalan Teoremi** ile daha verimli bir şekilde şifre çözme işlemi yapabilmek için;
  - $dP = (1/e) \pmod{p-1}$
  - $dQ = (1/e) \pmod{q-1}$
  - $qInv = (1/q) \pmod{p}$
    - (1/e) public exponent'in, (1/p) modulün tersi anlamında kullanılmıştır.

AÇIK VE GİZLİ ANAHTAR ÇİFTİ OLUŞTURMA

- Açık anahtar mod değeri olan **n** ve ortak üs olan **e**'den oluşur.
- Özel anahtar ise mod değeri olan **n**, açık üs olan **e**, özel üs olan **d**, -çin kalan teoremi ile daha verimli şifre çözümü için- **p**, **q**, **d (mod p-1)**,**d (mod q-1)**, **1/q mod p** değerlerini saklar.
- Açık ve gizli anahtar **ASN.1** formatı ile paketlenip **base64** ile kodlanır.

AÇIK VE GİZLİ ANAHTAR ÇİFTİ OLUŞTURMA

- Ubuntu 18.04 ortamında openssl kullanarak gizli ve açık anahtarlarımızı oluşturuyoruz.

```
syntaxbender@pc:~/sunum-rsa$ # 2048 bit anahtar boyutuna sahip bir rsa gizli anahtarı oluşturuyoruz.
syntaxbender@pc:~/sunum-rsa$ openssl genrsa -out private.pem 2048
Generating RSA private key, 2048 bit long modulus (2 primes)
.........+++++
..+++++
e is 65537 (0x010001)
syntaxbender@pc:~/sunum-rsa$ # gizli anahtardan açık anahtarı çıkaracağız
syntaxbender@pc:~/sunum-rsa$ openssl rsa -in private.pem -pubout -out public.pem
writing RSA key
```

- Resimde de göründüğü üzere public exponent değeri varsayılan olarak 65537 (0x10001) olarak tanımlandı. Bunun sebebi bit uzunluğu kısa olan ve küçük Hamming Ağırlığına sahip public exponent değerlerinin daha verimli şifreleme sağlamasıdır.

RSA GİZLİ ANAHTARI

- Gizli anahtarımızın içeriği aşağıdaki resimdekine benzer bir şekilde görünmekte;

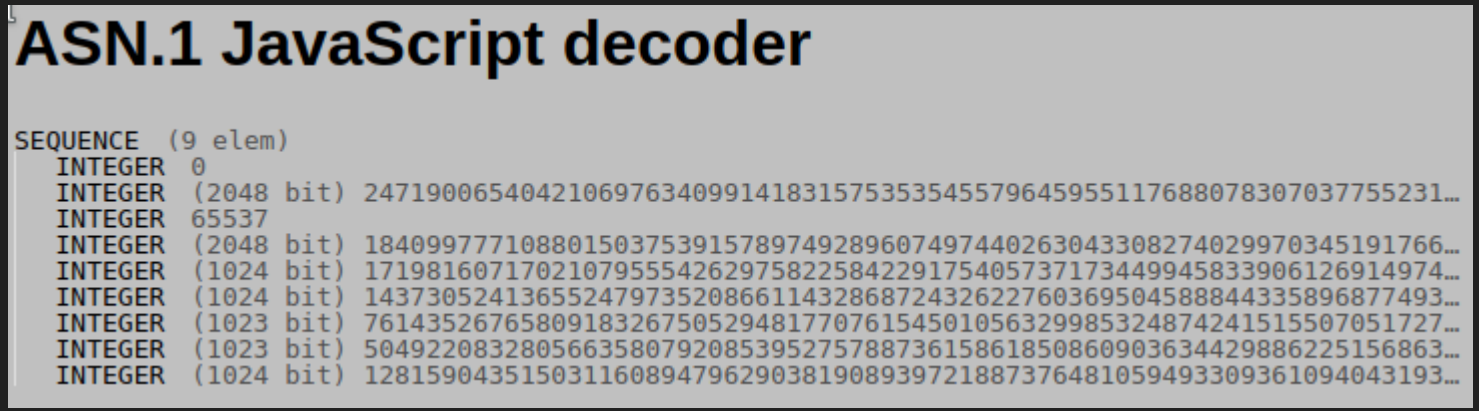
```
syntaxbender@pc:~/sunum-rsa$ cat private.pem
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAw8/vnt6hA75SSwFWoiFeXI8kEw5vFBBsjUVfy7gWxpW0gwpe
kSTnefH38eoEVo+mVe0O84S/cNJ4Cx7NyYSL6hn21h8pmjE9wlkAqRTBYtB3Bjb3
LUwZs88S10oQTaddgFaqTsNZ3526IqxUa/WX0bHpzvjTpcTt4mArjTBQC4sX8W0Z
o9ZWju7mn1LNwWQjEWbgCjev5MKWlIQMWlFyeLYkPWicuUHoeZkgeUN8oUn3M8W8
ualQI43GtZoLJB+oVfHFk9lRGL1Zw6IBQ6S823VrKvbfxKhyNlqBJASQJnk5v7iw
261/BEhp738e4OEaYFqZFEzASW4CLxoRgadhPQIDAQABAoIBAQCR1coXPqQZD6Rm
rGaDdeq7cPhjIPrlrJ3MUYDZTw/XcyPv/McVvrCPiAEsMSht96rgkB/0ikeylW5G
ifimP4udHR1Tt/3aExZ2KqdtRkIJgE1FWaaaG/WAj5butQCTdTUUInfRFXlYLDa3
IDX3+f/Kc4E5XuTyr0qksQ9+ph7XmL+x3pUWpxaIBXQyUzn1aNc2ZQNShvslxcYP
qyDAQYegaLC/EMSavVGwmQga4K8iGNLc1n6Wij7OGL9cxaFnrziO+MhxnS5s13Jm
Anir2WqrzaW6vg98jceSCOvRqXGXz3m4KzoJ0ND4ngIVMS6ToCoEXHeSKR+2OOg4
wSoYnSiBAoGBAPTo8bPOUvjAjgMr24iINCZ2im0zlrqXrA8Lk9kS8xxLycZ2JseD
5W64VpIEdrbJmJUXa2NtEvVWF4ch1+FqH3nx8/v1sJSuheRiVHN2+FgjF9YldFyS
hxdjenW6SnyYBk8f8q6Yz+dnLvvK1gfipPismZ6lGSvREFvZQ8+z0C7xAoGBAMyt
1ueNmFxKMsu5f3Nqnvl498aPCQcbzYMG/HiToHbJw1BygNiOpkn+94KCTnIkOcPC
vkFbOfx6UsETEPB6RdTMJYaDbvbKKooMLMJ6E9GV/UIROlT8VSxK59wyLXINGGoL
NsrhfwNt5rg3ABI8r7gRj2zs+AKH80A4ZUzMYu8NAoGAbG6WVN2MrsXSGmVYlWJ4
dG8XarHkkkiEzUR6+SkCqgyEEXJOSReEhTjoW03Ooa8v9xKfYS3xrt2nHRVG1258
8SjGtU9aFvoOHcfLTDPoUF15OD7s/e+CEEDi/yyzveYxJOcoM1VE91cf3is4B9W4
5MDGYw8FzfiKHaBb/aHHEoECgYBH5z4a5ExFB6/mj8jtU+R7QXrR30fNnn/I6vhg
p4P0lxatOTx4V0ylrBNpXruOL5m3Yr+RmCm7v4E44KjPwjSr9/eq1JtzyTu+eV50
B4zU0PtCbWjvmCGXyMzBN0Nt0JnmDwB1r1f+bWpjBrdYsMcOQ+T94BWUVgwnYGh9
3UNYNQKBgQC2gSlE/E9j8ex9OGxjrhHo6UEy59lOC0or3lC0cDzEuVzrok8i3UcP
176I0CFhtdfymwVE1kWfzHfejMesSarR9QQm6ZGhn5ubXTDAHXBwCGrmXQVoYsOb
OVdAwRIFM2W1HDjdgJhx0jByPj5Gfzu+9fIUFxJim6zr5M5N3AWufg==
-----END RSA PRIVATE KEY-----
```

RSA GİZLİ ANAHTARI

- Bir önceki slaytta gördüğümüz RSA gizli anahtarını incelersek;
  - Gördüğümüz anahtarda ilk gözümüze çarpan **base64** formatında kodlanmış bir veridir. Gizli anahtarın sakladığı değerler **ASN.1** ile paketlenir(**DER** formatında hexadecimal bir çıktısı vardır), **ASN.1** paketlemesi sonucu ortayan çıkan veri **base64** ile kodlanmıştır.
  - RSA anahtarlarında olduğu gibi Her **base64** ile kodlanan verinin **Unicode** karakter aralığında karşılığı olması gerekmez.

RSA GİZLİ ANAHTARI

- Gizli anahtarımızı bir **base64** decoder & **ASN.1** parser ile işlediğimizde aşağıdaki resimdekine benzer bir formatta çıktı almaktayız.



RSA GİZLİ ANAHTARI ASN.1 SÖZDİZİMİ

- Gizli anahtarımızın **ASN.1** parse işleminden sonraki halinde göz atmak gerekirse her bir satır aşağıdaki şekildedir.

```
RSAPrivateKey ::= SEQUENCE {
  version Version,                          Gelecekteki versiyonlar ile uyumluluk sağlanması için saklanan versiyon numarası.
  modulus INTEGER, -- n                     modulus, n = p.q
  publicExponent INTEGER, -- e              public exponent(ortak anahtar üssü)
  privateExponent INTEGER, -- d             private exponent (özel anahtar üssü)
  prime1 INTEGER, -- p                      prime1(p), n'nin asal çarpanıdır.
  prime2 INTEGER, -- q                      prime2(q), n'nin asal çarpanıdır.
  exponent1 INTEGER, -- d mod (p-1)         d mod (p - 1)
  exponent2 INTEGER, -- d mod (q-1)         d mod (q - 1)
  coefficient INTEGER -- (inverse of q) mod p   q'nun mod p'ye göre tersi
}
```

RSA AÇIK ANAHTARI

- Gizli anahtarımızın içeriği aşağıdaki resimdekine benzer bir şekilde görünmekte;

```
syntaxbender@pc:~/sunum-rsa$ cat public.pem
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAw8/vnt6hA75SSwFWoiFe
XI8kEw5vFBBsjUVfy7gWxpW0gwpekSTnefH38eoEVo+mVe0O84S/cNJ4Cx7NyYSL
6hn21h8pmjE9wlkAqRTBYtB3Bjb3LUwZs88S10oQTaddgFaqTsNZ3526IqxUa/WX
0bHpzvjTpcTt4mArjTBQC4sX8W0Zo9ZWju7mn1LNwWQjEWbgCjev5MKWlIQMWlFy
eLYkPWicuUHoeZkgeUN8oUn3M8W8ualQI43GtZoLJB+oVfHFk9lRGL1Zw6IBQ6S8
23VrKvbfxKhyNlqBJASQJnk5v7iw261/BEhp738e4OEaYFqZFEzASW4CLxoRgadh
PQIDAQAB
-----END PUBLIC KEY-----
```

RSA AÇIK ANAHTARI

- Bir önceki slaytta gördüğümüz RSA açık anahtarını incelersek;
  - Gizli anahtarda olduğu gibi açık anahtarlarda da anahtarın sakladığı veriler **ASN.1** ile paketlenmiş(**DER** formatında hexadecimal bir çıktısı vardır) **base64** formatında kodlanmıştır.
  - RSA anahtarlarında olduğu gibi her **base64** ile kodlanan verinin **Unicode** karakter aralığında karşılığı olması gerekmez.

RSA AÇIK ANAHTARI

- Açık anahtarımızı bir **base64** decoder & **ASN.1** parser ile işlediğimizde aşağıdaki resimdekine benzer bir formatta çıktı almaktayız.

RSA AÇIK ANAHTARI ASN.1 SÖZDİZİMİ

- Açık anahtarımızın **ASN.1** parse işleminden sonraki halinde göz atmak gerekirse her bir satır aşağıdaki ;

```
RSAPublicKey ::= SEQUENCE {
   modulus INTEGER, -- n                  modulus, n = p.q
   publicExponent INTEGER, -- e           public exponent(ortak anahtar üssü)
}
```

ŞİFRELEME

Şifrelemeden işlemine geçmeden önce bazı notlar;

- RSA'nın çeşitli sorunlarını ortadan kaldırmak için şifreleme öncesinde düz mesaj olan m'ye rastsallaştırılmış dolgu uygulanır. Bu dolgu güvensiz düz metin aralığında olmaktan korur ve sabit bir şifreli mesajı olmasını engeller. Dolgulama ve RSA'yı güvenli kılmak için tasarlanan **PKCS#1** standardının ilk versiyonlarının "**adaptif seçilmiş şifreli mesaj**" atağına karşı dayanıksızlığı ortaya çıkınca sonraki versiyonlar bu atağı engellemek için **OAEP** içermekteler.
- Şifrelenebilecek mesaj uzunluğu kısıtlıdır. Uygulanan **PKCS#1** standardı versiyonun uyguladığı dolgu algoritmasına göre ve anahtar boyutuna göre mesaj uzunluğu farklılık göstermektedir. Şifrelenecek mesaj dolgu fonksiyonundan geçtiğinde mesajın bit uzunluğu modulus bit uzunluğundan küçük olmalıdır. Uygulanacak dolgu fonksiyonuna göre maksimum mesaj uzunluğu değiştirmektedir.

ŞİFRELEME

- Dolgulama işlemine girmeden daha basit olarak şifreleme işleminden bahsedersek;
- Şifrelenecek olan mesajdaki her bir karakterin **UTF-8** kodlanmış bit sayı değeri onaltılık tabanda yazılır. Bu işlem sayesinde **UTF-8** kodlanmış 0-255 aralığında olan 8 bitlik her bir byte, onaltılık tabandaki bir rakama karşılık gelir ve geri dönüştürülebilir, parse edilebilirdir.

ŞİFRELEME

- Örnek olarak biz “Test” mesajını şifrelemek istiyoruz diyelim.
- **UTF-8** olarak kodlanan Test stringi;
  - Onluk tabanda; 84 101 115 116
  - Onaltılık tabanda; 54 65 73 74

  sayılarına denk gelir. Onaltılık tabanda yazılan tam sayının onluk tabandaki halini şifreleme esnasında kullanacağız. Yani bizim şifrelenecek mesaj için sayımız artık onaltılık tabanda “54657374”, onluk tabanda ise “1415934836” olarak **büyük bir tamsayı** halini alacak.

ŞİFRELEME

- Elde ettiğimiz onluk tabandaki sayıya “**m**” diyelim.
- Anahtar oluşumu esnasında belirlediğimiz ve ortak anahtarda sakladığımız **e** değeri ve **n** değeri ile aşağıdaki denkliği uygulamalıyız.
- **c ≡ m^e (mod n)** denkliğini sağlayan **c** değerine ulaşılır. **c** değeri bizim şifrelenmiş mesajımızı oluşturur.
- Son olarak **c** değerini hexadecimal tabana çevirip üzerine **base64** kodlanır. Elde ettiğimiz **base64** çıktısı mesajın şifrelenmiş halidir.

ŞİFRE ÇÖZME

- Dolgulama uygulanmamış bir şifreli RSA verisinin adım adım çözülmesi işleminden bahsedersek;
- İlk olarak base64 kodlanmış şifreli veriyi base64'ten onaltılık tabana, onaltılık tabanda yazdığımız büyük tam sayıyı ise onluk tabana çeviriyoruz. Ulaştığımız onluk tabandaki büyük tam sayıya **c** diyelim.
- Anahtar oluşumu esnasında belirlediğimiz ve gizli anahtarda sakladığımız **d** değeri ve **n** değeri ile aşağıdaki denkliği uygulamalıyız.
- $\mathbf{m \equiv c^d \pmod{n}}$ denkliğini sağlayan **m** değerine ulaşılır. **m** değeri bizim mesajımızın ondalık tabandaki halini oluşturur.
- Bulduğumuz ondalık tabandaki sayıyı onaltılık tabanda yazıp UTF-8 decode işlemi sonrası şifresiz mesaja ulaşmış oluyoruz.

ÇİN KALAN TEOREMİ İLE ŞİFRE ÇÖZME

- Bilinmeyen bir x tam sayısının belirli bir tam sayı grubu tarafından bölündüğünde ortaya çıkan kalanlar biliniyorsa, buradan x in alabileceği değer ve değerler bulunabilir.
- **CRT**(Chinese Remainder Theorem) algoritmasında **m = c^d (mod n)** denklemini daha verimli çözmemizde bize yardımcı olur.

  Bunun için öncelikle şu ön işlemleri yapmamız gerekir. Zaten bu değerler halihazırda şifrelemeyi çözmek üzere gizli anahtarımızda bulunmaktadır. (**dP** = exponent1, **dQ** = exponent2, **qInv** = coefficient)

- **dP** = (1/e) (mod p-1)
- **dQ** = (1/e) (mod q-1)
- **qInv**= (1/q) (mod p)
  - (1/e) public exponent'in, (1/p) modulün tersi anlamında kullanılmıştır.

ÇİN KALAN TEOREMİ İLE ŞİFRE ÇÖZME

- Gizlenmiş mesajı orjinal mesaja çevirmek için;
- m_1 adımı = c^{dP} (mod p)
- m_2 adımı = c^{dQ} (mod q)
- h = qInv . (m_1 - m_2) (mod p)
- m = m_2 + h.q
  - **p**, **q**, **dP**, **dQ**, **qInv** değerleri RSA gizli anahtarında saklanmaktadır.

Teorem

- $n_1, n_2, n_3, \ldots n_k$ aralarında asal tam sayılar olsun
- $a_1, a_2, a_3, \ldots a_k$ rastgele tam sayılar olmak üzere
- x = a_1 (mod n_1)

  x = a_1 (mod n_2)

  ...

  ...

  ...

  x = a_k (mod n_k)
- N= $n_1 . n_2 . n_3 . \ldots n_k$
- Her i (1, 2, 3, … k) sayısı için y_i = N/n_i
- Her i (1, 2, 3, … k) sayısı için z_i = ${y_i}^{-1}$ (mod n_i) (uzatılmış öklid algoritması kullanılarak çözülebilir.)
- Buradan x tam sayısı x = Σ i=1,k $a_i . y_i . z_i$

GÜVENLİK VE SALDIRI TÜRLERİ

- Düz(Dolgu uygulanmayan) RSA karşısındaki Saldırılar;
  - Küçük şifreleme üssü (örn. **e = 3**) kullanıldığında şifrelenen m mesajı da $m < n^{1/e}$ koşulunu doğruluyorsa ,m^e değeri **n**'den küçük bir değere karşılık gelir. Bu durumda şifreli mesajlar tam sayılardaki **e**'inci dereceden kökleri alınarak kolayca çözülebilirler.
  - Eğer bir mesaj aynı şifreleme üssü **e** ve farklı **n** değerleri ile birden çok kişiye gönderiliyorsa, bu mesaj Çin Kalan Teoremi ve logoritma kullanılarak kolayca çözülebilir.
  - RSA algoritması herhangi bir rastsallık içermediği için bazı düz mesajlar şifrelenip herhangi bir şifreli mesaja eşit olup olmadığı kontrol edilebilir. Dolayısıyla dolgu kullanılmayan RSA anlamsal güvenliğe sahip değildir.

GÜVENLİK VE SALDIRI TÜRLERİ

- Düz(Dolgu uygulanmayan) RSA karşısındaki Saldırılar;
  - RSA yönteminde iki şifreli mesajın çarpımı, kendilerine karşılık gelen düz mesajların çarpımının şifrelenmiş haline eşittir.
    - Yani, $m_1{}^e . m_2{}^e = (m_1 . m_2)^e$ (mod n) Bu çarpımsal özelliğinden dolayı RSA'e seçilmiş şifreli mesaj atağı gerçekleştirilebilir.
    - Örneğin, bir şifreli mesaj olan $c = m^e$ (modn) 'nin çözümünü elde etmek isteyen bir saldırgan rastgele bir r değeri seçip, özel anahtara sahip kişiye şüpheli gözükmeyen $c' = c . r^e$ (mod n) mesajını gönderip deşifre etmesini isteyebilir.
    - Çarpımsal özellikten dolayı c', $m.r$(mod n)'in şifreli halidir. Eğer saldırgan atakta başarılı olursa $m.r$(mod n) değerini elde eder ve r^{-1} ile çarpıp kolayca m değerini bulur.