BCTF warmup 50

转载

weixin_30955341 于 2015-03-30 13:19:00 发布 48 收藏
文章标签： python
原文链接：http://www.cnblogs.com/wangaohui/p/4377234.html

这是一道关于RSA的解密题；首先，我们要明白，通常是公钥加密、私钥解密，私钥签名、公钥验证。这个题目中给出的是一个公钥和一段密文。

刚开始一直以为和验证签名有关，费劲脑汁也想不出来怎么办。下面介绍些思路。

首先，利用openssl分析公钥的格式，获得modulus和expoent。

方法一：利用openssl asn1parse来分析公钥格式

```
root@bt:~/Desktop# openssl asn1parse -in publickey.pub
    0:d=0  hl=4 l= 546 cons: SEQUENCE
    4:d=1  hl=2 l=  13 cons: SEQUENCE
    6:d=2  hl=2 l=   9 prim: OBJECT            :rsaEncryption
   17:d=2  hl=2 l=   0 prim: NULL
   19:d=1  hl=4 l= 527 prim: BIT STRING
```

那么在偏移为19处就是证书的expoent和modulus的信息。

```
root@bt:~/Desktop# openssl asn1parse -in publickey.pub -i -strparse 19
    0:d=0  hl=4 l= 522 cons: SEQUENCE
    4:d=1  hl=4 l= 257 prim:  INTEGER
:0367198D6B5614E95813ADD8F22A4717BC72BE1EABD933D1B86944FDB75B8ED230BE62D7D1B69D222095C128C86F82012ECB116191F
D9D018A6D02F84DB27BC51A21307DC86F4BF771C691C143E5ABE549B5BD2D6EB1A21FD6270E7E1B48FE0611FBB2E1B0B3524E6F4DE8B
4E4A345DA44A13DE825B72608DB6C7C4A40B78266E6C87BBFDEF6B48381D49C4507A58BCD47B76D64B45908B158BD7EBC4DACB0B1CFD
6C2C19574F40EB2EFD0E9E10DC7005CAD39BCAF52B9EAC3873368D69031C5E724684A44F068EFD1D3DC096D9B5D6411E58BDEE43E46B
99A0D0494B9DB28195AF901AFF130D4A6E203DAD08DA57FA7E40262A5BADB2A323EDA28B44696AB305D
  265:d=1  hl=4 l= 257 prim:  INTEGER
:F3959D978E02EB9F06DEF3F335D8F8AFD7609951DDAC60B714B6C22AF0FA912F210B34206BD24A9601C78DF4A0275F107FD3AB552D9
5057EB934E71BDDCD7045C24B18587B8C8FCF5ADD4C5D83F0C77C94DC9C50CBE438E2B67BAFD31633B6AAF1781D90C3AD6F03D037B33
21801B23546D483E67E26067F7B22347DDBC0C2D592CE814CBF5DFCCC141437F14E0B3990F88061E5F0BAE5F01E3FA70DB0E9605E7CF
D575E9C81EFEEC529C33FD9037A20FD8ACD513AC9637768313E63F9838AE3511CDD0A9A2B516F2148C8D475A360A06359449739EECD2
51ABB42B014573E439F2FA4573557B25699FFC11E631CE8EE975A86E7E272BCF5F76A93450348FE3F
```

则第二行的十六进制就是modulus，第三行的十六进制则是expoent。观察到这里，expoent几乎和modulus一样长。

方法二：利用openssl rsa来获得modulus和expoent。

```
root@bt:~/Desktop# openssl rsa -in publickey.pub -pubin -modulus -text
Modulus (2050 bit):
    03:67:19:8d:6b:56:14:e9:58:13:ad:d8:f2:2a:47:
    17:bc:72:be:1e:ab:d9:33:d1:b8:69:44:fd:b7:5b:
    8e:d2:30:be:62:d7:d1:b6:9d:22:20:95:c1:28:c8:
    6f:82:01:2e:cb:11:61:91:fd:9d:01:8a:6d:02:f8:
    4d:b2:7b:c5:1a:21:30:7d:c8:6f:4b:f7:71:c6:91:
    c1:43:e5:ab:e5:49:b5:bd:2d:6e:b1:a2:1f:d6:27:
    0e:7e:1b:48:fe:06:11:fb:b2:e1:b0:b3:52:4e:6f:
    4d:e8:b4:e4:a3:45:da:44:a1:3d:e8:25:b7:26:08:
    db:6c:7c:4a:40:b7:82:66:e6:c8:7b:bf:de:f6:b4:
    83:81:d4:9c:45:07:a5:8b:cd:47:b7:6d:64:b4:59:
    08:b1:58:bd:7e:bc:4d:ac:b0:b1:cf:d6:c2:c1:95:
    74:f4:0e:b2:ef:d0:e9:e1:0d:c7:00:5c:ad:39:bc:
    af:52:b9:ea:c3:87:33:68:d6:90:31:c5:e7:24:68:
    4a:44:f0:68:ef:d1:d3:dc:09:6d:9b:5d:64:11:e5:
    8b:de:e4:3e:46:b9:9a:0d:04:94:b9:db:28:19:5a:
    f9:01:af:f1:30:d4:a6:e2:03:da:d0:8d:a5:7f:a7:
    e4:02:62:a5:ba:db:2a:32:3e:da:28:b4:46:96:ab:
    30:5d
Exponent:
    00:f3:95:9d:97:8e:02:eb:9f:06:de:f3:f3:35:d8:
    f8:af:d7:60:99:51:dd:ac:60:b7:14:b6:c2:2a:f0:
    fa:91:2f:21:0b:34:20:6b:d2:4a:96:01:c7:8d:f4:
    a0:27:5f:10:7f:d3:ab:55:2d:95:05:7e:b9:34:e7:
    1b:dd:cd:70:45:c2:4b:18:58:7b:8c:8f:cf:5a:dd:
    4c:5d:83:f0:c7:7c:94:dc:9c:50:cb:e4:38:e2:b6:
    7b:af:d3:16:33:b6:aa:f1:78:1d:90:c3:ad:6f:03:
    d0:37:b3:32:18:01:b2:35:46:d4:83:e6:7e:26:06:
    7f:7b:22:34:7d:db:c0:c2:d5:92:ce:81:4c:bf:5d:
    fc:cc:14:14:37:f1:4e:0b:39:90:f8:80:61:e5:f0:
    ba:e5:f0:1e:3f:a7:0d:b0:e9:60:5e:7c:fd:57:5e:
    9c:81:ef:ee:c5:29:c3:3f:d9:03:7a:20:fd:8a:cd:
    51:3a:c9:63:77:68:31:3e:63:f9:83:8a:e3:51:1c:
    dd:0a:9a:2b:51:6f:21:48:c8:d4:75:a3:60:a0:63:
    59:44:97:39:ee:cd:25:1a:bb:42:b0:14:57:3e:43:
    9f:2f:a4:57:35:57:b2:56:99:ff:c1:1e:63:1c:e8:
    ee:97:5a:86:e7:e2:72:bc:f5:f7:6a:93:45:03:48:
    fe:3f
Modulus=367198D6B5614E95813ADD8F22A4717BC72BE1EABD933D1B86944FDB75B8ED230BE62D7D1B69D222095C128C86F82012ECB1
16191FD9D018A6D02F84DB27BC51A21307DC86F4BF771C691C143E5ABE549B5BD2D6EB1A21FD6270E7E1B48FE0611FBB2E1B0B3524E6
F4DE8B4E4A345DA44A13DE825B72608DB6C7C4A40B78266E6C87BBFDEF6B48381D49C4507A58BCD47B76D64B45908B158BD7EBC4DACB
0B1CFD6C2C19574F40EB2EFD0E9E10DC7005CAD39BCAF52B9EAC3873368D69031C5E724684A44F068EFD1D3DC096D9B5D6411E58BDEE
43E46B99A0D0494B9DB28195AF901AFF130D4A6E203DAD08DA57FA7E40262A5BADB2A323EDA28B44696AB305D
```

同样的，我们获得了modulus和expoent。

由于expoent很大，可以使用Wiener攻击来得到d，从而对信息解密。

求d的python脚本使用了https://github.com/pablocelayes/rsa-wiener-attack上的，稍微修改了下。

```
import ContinuedFractions, Arithmetic
import sys
sys.setrecursionlimit(1000000)
def hack_RSA(e,n):
    '''
    Finds d knowing (e,n)
    applying the Wiener continued fraction attack
    '''
    frac = ContinuedFractions.rational_to_contfrac(e, n)
    convergents = ContinuedFractions.convergents_from_contfrac(frac)

    for (k,d) in convergents:

        #check if d is actually the key
        if k!=0 and (e*d-1)%k == 0:
            phi = (e*d-1)//k
            s = n - phi + 1
            # check if the equation x^2 - s*x + n = 0
            # has integer roots
            discr = s*s - 4*n
            if(discr>=0):
                t = Arithmetic.is_perfect_square(discr)
                if t!=-1 and (s+t)%2==0:
                    print("Hacked!")
                    return d
if __name__ == "__main__":
    e =
0xF3959D978E02EB9F06DEF3F335D8F8AFD7609951DDAC60B714B6C22AF0FA912F210B34206BD24A9601C78DF4A0275F107FD3AB552D
95057EB934E71BDDCD7045C24B18587B8C8FCF5ADD4C5D83F0C77C94DC9C50CBE438E2B67BAFD31633B6AAF1781D90C3AD6F03D037B3
321801B23546D483E67E26067F7B22347DDBC0C2D592CE814CBF5DFCCC141437F14E0B3990F88061E5F0BAE5F01E3FA70DB0E9605E7C
FD575E9C81EFEEC529C33FD9037A20FD8ACD513AC9637768313E63F9838AE3511CDD0A9A2B516F2148C8D475A360A06359449739EECD
251ABB42B014573E439F2FA4573557B25699FFC11E631CE8EE975A86E7E272BCF5F76A93450348FE3F
    n =
0x367198D6B5614E95813ADD8F22A4717BC72BE1EABD933D1B86944FDB75B8ED230BE62D7D1B69D222095C128C86F82012ECB116191F
D9D018A6D02F84DB27BC51A21307DC86F4BF771C691C143E5ABE549B5BD2D6EB1A21FD6270E7E1B48FE0611FBB2E1B0B3524E6F4DE8B
4E4A345DA44A13DE825B72608DB6C7C4A40B78266E6C87BBFDEF6B48381D49C4507A58BCD47B76D64B45908B158BD7EBC4DACB0B1CFD
6C2C19574F40EB2EFD0E9E10DC7005CAD39BCAF52B9EAC3873368D69031C5E724684A44F068EFD1D3DC096D9B5D6411E58BDEE43E46B
99A0D0494B9DB28195AF901AFF130D4A6E203DAD08DA57FA7E40262A5BADB2A323EDA28B44696AB305D
    d = hack_RSA(e,n)
    print d
#d为：
422190901650907812920180123687944676069788522092850669615064693823744099274668340988114145183193919060974344
767652532554396336235392398907619947051575839 9
```

解密的脚本，参考自：http://rickgray.me/2015/03/23/bctf2015-writeup.html 写得很简单，但很清晰

```
import binascii

n =
0x367198D6B5614E95813ADD8F22A4717BC72BE1EABD933D1B86944FDB75B8ED230BE62D7D1B69D222095C128C86F82012ECB116191F
D9D018A6D02F84DB27BC51A21307DC86F4BF771C691C143E5ABE549B5BD2D6EB1A21FD6270E7E1B48FE0611FBB2E1B0B3524E6F4DE8B
4E4A345DA44A13DE825B72608DB6C7C4A40B78266E6C87BBFDEF6B48381D49C4507A58BCD47B76D64B45908B158BD7EBC4DACB0B1CFD
6C2C19574F40EB2EFD0E9E10DC7005CAD39BCAF52B9EAC3873368D69031C5E724684A44F068EFD1D3DC096D9B5D6411E58BDEE43E46B
99A0D0494B9DB28195AF901AFF130D4A6E203DAD08DA57FA7E40262A5BADB2A323EDA28B44696AB305D
d =
42219090165090781292018012368794467606978852209285066961506469382374409927466834098811414518319391906097434476765253255439633623539239890761994705157583991
c =
0x1e04304936215de8e21965cfca9c245b1a8f38339875d36779c0f123c475bc24d5eef50e7d9ff5830e80c62e8083ec55f27456c80b0ab26546b9aeb8af30e82b650690a2ed7ea407dcd094ab9c9d3d25a93b2140dcebae1814610302896e67f3ae37d108cd029fae6362ea7ac1168974c1a747ec9173799e1107e7a56d783660418ebdf6898d7037cea25867093216c2c702ef3eef71f694a6063f5f0f1179c8a2afe9898ae8dec5bb393cdffa3a52a297cd96d1ea602309ecf47cd009829b44ed3100cf6194510c53c25ca7435f60ce5f4f614cdd2c63756093b848a70aade002d6bc8f316c9e5503f32d39a56193d1d92b697b48f5aa43417631846824b5e86

m = hex(pow(c,d,n)).rstrip("L")
print m
print binascii.unhexlify(m[2:])
#0x424354467b39657452656134792l7d
#BCTF{9etRea4y!}
```

参考文献：

http://www.ruanyifeng.com/blog/2013/06/rsa_algorithm_part_one.html

http://stackoverflow.com/questions/3116907/rsa-get-exponent-and-modulus-given-a-public-key

http://www.openssl.org/docs/apps/openssl.html

https://github.com/pablocelayes/rsa-wiener-attack

http://rickgray.me/2015/03/23/bctf2015-writeup.html

转载于:https://www.cnblogs.com/wangaohui/p/4377234.html