CTF之Crypto新手入门

原创

RayLee23333 于 2020-11-23 11:11:20 发布 7305 收藏 48

分类专栏： ctf crypto

本文链接：https://blog.csdn.net/qq_33458986/article/details/100171871


ctf 同时被 2 个专栏收录

6 篇文章 0 订阅

订阅专栏


crypto

3 篇文章 0 订阅

订阅专栏

Crypto

看了《密码编码学与网络安全》的大部分内容，首先做了攻防世界网站里面的新手题目，对它考查的内容有了初步的认识，有加密和编码的内容，加密包含了几种密码：栅栏密码，摩斯密码，RSA密码，轮转机，培根密码等

编码包括：BASE64（6个bit组成一个字符，共有64种字符）Unicode ，ASCII

学习到的Crypto所需工具

最好在kali环境下完成，RSA密码使用openssll以及代码版的RSATools

编码的话收集到了许多转换工具，这里也不一一赘述，比较好用的工具叫 **convertor**

openssl的RSA解密过程

这种题目一般会给你一个.enc文件（被加密过的flag文件）还有一个.pem文件（公钥文件）

在Kali系统的terminal中使用OpenSSL执行 `openssl rsa -pubin -in pubkey.pem -text -modulus` ，即可看到该pem文件内的公钥e和模数n，接着在质数分解网站分解这个模数得到p和q两个质数，之后可以使用exe版的RSAtool得出私钥d。

如果是要求我们解密enc文件的话，我们需要使用python版的rsatools获得私钥的pem文件，执行以下命令：

```
python3 rsatool.py -f PEM -o private.pem -p Value_p -q Value_q -e Value_e
```

我们就获得了一个private.pem的私钥文件，之后再执行以下命令：

```
openssl rsautl -decrypt -in flag.enc -inkey private.pem -out flag.dec
```

我们就能得到一个flag.dec文件，里面就是解密得到的字符了。

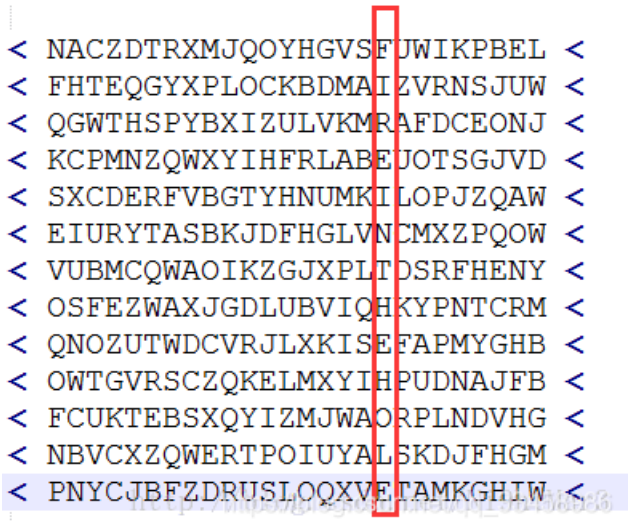
RSA中的攻击方式

1. 共模攻击
2. 低指数/高指数攻击
3. 中国余数定理/广播攻击
4. 低指数（相比2而言，指数已经比较高了），密文间有线性关系，也可以进行攻击

XCTF的轮转机解密

Wheel Cipher
加密表：
1: < ZWAXJGDLUBVIQHKYPNTCRMOSFE <
2: < KPBELNACZDTRXMJQOYHGVSFUWI <
3: < BDMAIZVRNSJUWFHTEQGYXPLOCK <
4: < RPLNDVHGFCUKTEBSXQYIZMJWAO <
5: < IHFRLABEUOTSGJVDKCPMNZQWXY <
6: < AMKGHIWPNYCJBFZDRUSLOQXVET <
7: < GWTHSPYBXIZULVKMRAFDCEONJQ <
8: < NOZUTWDCVRJLXKISEFAPMYGHBQ <
9: < XPLTDSRFHENYVUBMCQWAOIKZGJ <
10: < UDNAJFBOWTGVRSCZQKELMXYIHP <
11： < MNBVCXZQWERTPOIUYALSKDJFHG <
12： < LVNCMXZPQOWEIURYTASBKJDFHG <
13： < JZQAWSXCDERFVBGTYHNUMKILOP <

密钥为：2，3，7，5，13,12,9，1，8，10，4，11，6
密文为：NFQKSEVOQOFNP



既然是车轮，看来需要轮换，一开始以为是将密文对应密钥位置进行替换，发现不对，查了查发现Jefferson wheel cipher(杰弗逊转轮加密器),差不多重新排一下序，并把密文转到第一个位置，发现flag：FIREINTHEHOLE

一些奇奇怪怪的加密或编码方法

BrainFuck和OOK！编码：一个好心人的解码链接
猪圈密码，培根密码，标准银河字母的编码，佛箴言密码，云影密码
一个很全的集合

python相关知识

熟悉python的base64库内的各种decode和encode，chr和ord是两对互补的函数
以及python中调用Crypto中的AES加密等操作

11.21更新：

关于sage：sage是一个相当全面的数学工具，第三届红帽杯和其他大大小小比赛都有相关的sage写的题目。
这里介绍红帽杯中的Related一题，题目大致是，三个不同的明文通过RSA加密后的密文已知，公钥也已知，同时三个明文的和也已知，其实上述这些条件已经可以解出答案了，但是题目还给出了两个无关的信息，这是我觉得题目可能要混淆人的地方，总结下来，就是解答下面这个方程组：

$$x + x + x$$ 17

```
#!/usr/bin/sage -python
from Crypto.Util import number
from sage.all import *


N = 160849237602641690994843533179529793483618558609352561574020279833494570217676143321731540442069670152521051
091152899206856573945178791771034143484874773780252595897609962709093253717314338762898978743037334241151177760 4
259235904148205973770872139611825475677815243582169215482423688118215600080695840300550673289182355532480052893 4
757672719379501318525189471726279397236710401497352477683714139039769105043411654493442696289499967521222951945 8
232333718451108074699446023452930683465746302735398701161588175565235651990938745870972303141663652202907309373 8
098322859941413734149820596787018164037098140262736081225164 9
Cs = [106072354000985866999943925848418065920006608161913150089479177736054763658845720565446214668076362374158 93
192966935651590312237598366247520986667580174438232591692369894702423377081613821241343307094343575042030793564 1
183024884018881975176253339237101727389137714846285573101649743844628560470654869130466471333862469764579612651 1
534910303994680238689731517663327429541037198642203910674521623040112354286371430111475323988882044211253828519 4
875243192862692290859625788686421276234445677411280606266052059579743874849594812733193363406594409214632722438 5
923765183101712972340815550287275389519347617268784433110719901, 266534807595283666545532335089184278193847137294 3
896177948046901127648217780657532963063228780230203325378931053293617434754585479452556620021360669764370971665 6
619743473463613391689402725053682169256850873752706252379747752552015341379702582040497607180172854652311649467 8
787144256986761422125883800803611005266144235337671967492747413802588429049681475080330918199790425603367035641 2
827952738096938533084575999865754077733911351903655245482932366624226960722515684608470595713112772035186848337 5
138773025602253783595007177712673092409157674720974653789039702431795168654387038080256838321255342848782705785 5
24911705L, 488122571389541415183068525928874098142466240024889708636516664385340994781865450969229925096093851140 0
178276416929668757746679501254041354795468626916196040017280791985239849062273782179873724736552198083211250561 9
192059448730545500442981534768431023858984817288359193663144417753847196868565476919041282010484259630583394963 5
804243587437543349568335983514245152298831480814924718742325554563620890239769297665303713208766519408552972494 7
443856480134916058427933039012464716197806221216765180154233949297999618011342678854874769762792918534509941727 7
514336871895320190003343422118382995123154789034186420560976797l7L, 125344259734580612805730133780548362488883351 9
198966169076118474130362704619767247747943108676623695140384169222126709673116428645230760767457471129655666350 2
506683228995680732465415088464386342872490680369016655478936552807671968568443756281773813513113878888432223074 4
822799071467801057930486754765848958175210322557397925701113923697213082573030671328710797477330607663002433808 1
124142200612113688850435053038506912906079973403207309246156198371852177700671999937121772761984895354214794816 4
821095854093211573035128059236764164673155736737017384505692476799121977302450135397244937801849525848138917398 3
7153776754362L]
s = 28051355011019774582989056743626549699 0

e = 17
l = len(Cs)
PR = PolynomialRing( Zmod(N), 'x', l )
x = PR.gens()
f1 = (65537*x[0] - 66666*x[1] + 12345*x[2] - x[3])
f2 = x[0] + x[1] + x[2] - s
Fs = [f1, f2]
Fs.extend( [ (x[i]**e - Cs[i]) for i in range(l) ] )
I = Ideal(Fs)
B = I.groebner_basis()
m = ''
for b in B[:-1][::-1]:
    assert b.degree() == 1
    mi = ZZ( -b(0,0,0,0) )
    m += number.long_to_bytes(mi)
print m
```

其实，这个代码是可以简化的，如下：

```
#!/usr/bin/sage -python
from Crypto.Util import number
from sage.all import *


N = 160849237602641690994843533179529793483618558609352561574020279833494570217676143321731540442069670152521051
091152899206856573945178791771034143484874773780252595897609962709093253717314338762898978743037334241151177760 4
259235904148205973770872139611825475677815243582169215482423688118215600080695840300550673289182355532480052893 4
757672719379501318525189471726279397236710401497352477683714139039769105043411654493442696289499967521222951945 8
232333718451108074699446023452930683465746302735398701161588175565235651990938745870972303141663652202907309373 8
098322859941413734149820596787018164037098140262736081225164 9
Cs = [106072354000985866999943925848418065920006608161913150089479177736054763658845720565446214668076362374158 93
192966935651590312237598366247520986667580174438232591692369894702423377081613821241343307094343575042030793564 1
183024884018881975176253339237101727389137714846285573101649743844628560470654869130466471333862469764579612651 1
534910303994680238689731517663327429541037198642203910674521623040112354286371430111475323988882044211253828519 4
875243192862692290859625788686421276234445677411280606266052059579743874849594812733193363406594409214632722438 5
923765183101712972340815550287275389519347617268784433110719901, 266534807595283666545532335089184278193847137294 3
896177948046901127648217780657532963063228780230203325378931053293617434754585479452556620021360669764370971665 6
619743473463613391689402725053682169256850873752706252379747752552015341379702582040497607180172854652311649467 8
787144256986761422125883800803611005266144235337671967492747413802588429049681475080330918199790425603367035641 2
827952738096938533084575999865754077733911351903655245482932366624226960722515684608470595713112772035186848337 5
138773025602253783595007177712673092409157674720974653789039702431795168654387038080256838321255342848782705785 5
24911705L, 488122571389541415183068525928874098142466240024889708636516664385340994781865450969229925096093851140 0
178276416929668757746679501254041354795468626916196040017280791985239849062273782179873724736552198083211250561 9
192059448730545500442981534768431023858984817288359193663144417753847196868565476919041282010484259630583394963 5
804243587437543349568335983514245152298831480814924718742325554563620890239769297665303713208766519408552972494 7
443856480134916058427933039039012464716197806221216765180154233949297999618011342678854874769762792918534509941727 7
514336871895320190003343422118382995123154789034186420560976797 17L, 125344259734580612805730133780548362488883351 9
898661690761184741303627046197672477479431086766236951403841692221267096731164286452307607674574711296556663502 5
066832289956807324654150884643863428724906803690166554789365528076719685684437562817738135131138788884322230744 8
227990714678010579304867547658489581752103225573979257011139236972130825730306713287107974773306076630024338081 1
241422006121136888504350530385069129060799734032073092461561983718521777006719999371217727619848953542147948164 8
210958540932115730351280592367641646731557367370173845056924767991219773024501353972449378018495258481389173983 7
153776754362L]
s = 28051355011019774582989056743626549699 0

e = 17
l = 3
PR = PolynomialRing( Zmod(N), 'x', l )
x = PR.gens()
#f1 = (65537*x[0] - 66666*x[1] + 12345*x[2] - x[3])#其实就是这里的x[3]没必要解出来，感觉出题人就是在混淆视听
f2 = x[0] + x[1] + x[2] - s
Fs = [f2]
Fs.extend( [ (x[i]**e - Cs[i]) for i in range(l) ] )
I = Ideal(Fs)
B = I.groebner_basis()
m = ''
for b in B[::-1]:
    assert b.degree() == 1
    mi = ZZ( -b(0,0,0) )
    m += number.long_to_bytes(mi)
print m
```

sage中，term（项）和monomial（单项式）是不同的的，类Polynomial中有两个函数可以分别得到leading term和leading monomial（leading的意思可以在书中找到），这里主要说明一下term是带coefficient的monomial，如term是$2x$ 而相应的monomial是x

$$d$$

小总结

Crypto的题目千变万化，还有其他许多类型，接下来应该具体参考CTFWiki之类的网站还有实际做题了