

BABY_REBELLION



CHALLENGE INFO

Baby Rebellion

The earth has been taken over by cyborgs for a long time. We are a group of humans, called 'The Rebellion', fighting for our freedom. Lately, cyborgs have set up a lab where they insert microchips inside humans to track them down. Our team of IT experts has hacked one of the cyborgs' mail servers. There is a suspicious encrypted mail which possibly contains information related to the location of the lab. Can you decrypt the message and find the coordinates of the lab?



This challenge has a downloadable part.

MATERIAL:
andromeda.crt
corius.crt
mechi.crt
challenge

FLAG:
HTB{37_c0m3_h0m3_d1nn3r_15_r34dy}

SOLVER:
M1gnus

FOOTHOLD

The challenge provides three [x509](#) certificates (**andromeda.crt**, **corius.crt**, **mechi.crt**) and a p7m file (a file format based on [pkcs7](#)) [base64 encoded](#). By using Openssl is possible to dump informations about certificate's moduli and p7m file's content:

```
openssl x509 -in andromeda.crt -text          to dump complete informations about the cert
openssl x509 -in andromeda.crt -modulus      to dump certificate's modulus
openssl pkcs7 -in mailcert.p7m -inform DER -print  to dump complete informations about the p7m file
```

from the p7m file is possible to learn that a symmetric key was encrypted using the public keys of Andromeda, Corius and Mechi and is also possible to extract the encrypted flag and the iv.

from the certificates is possible to learn that the public exponent used for the [RSA](#) encryption is the same for all the x509 certificates and it's really small (3). With this informations an attacker know what he needs to perform the right attack.

THE ATTACK

From the informations gained is possible to perform the [Hastad's broadcast attack](#) (implemented also in [RSArmageddon](#)) to recover the symmetric key we need to decrypt the flag.

THE IMPLEMENTATION

```
import binascii
import re
from Crypto.Cipher import AES
```

```
c1 =
0xb58e82bea0e7a56624a98d12abe2b6dc36c677a82d29ae3ba41a3ea60d71e3012ebb3b77c934e7ddef1b773ead7f6bb3151df788ca456d896bcca38b650f94a5ac1753a36c388a1
0df5e8e3827d695b9f84ae512a5cee43c78af16353f4e0d90b9e2fc4abd200d590f6ce531d1d7cfb2f774caebe442ca2e28a42c54c3e9383b6cdb5c17af2534aec3921331aec6ee929e
63bb08cb1056039a8f5f6e30a751b0ec3dc6667a953508bafadabea06c41dd50b5056faaf49aa5cddb83a7e3ada096c68d20ad76f1c29998745697371e3f715cd33bfee04efaed93614
8f9c70b600d2f80df459eb67858509a6cc42ef22469d30c9c629efa827cc3cdf0beaeacf6a

c2 =
0x1a10597041de0a661f19be335e38a9e7fa4433fa5fd1eca32f05cf87d7ccac0eb071fef38078680e37af7008261e785252210476f40d47787bfd6e487e5d9f12bc7ac099f8e7f1bfff59
cbd8aa89be9df66989b3a84031b0aa3ca0feb51d209dc37e9ec14b54759d0754870dbf06cace73218f3efd53d5837091ec43d5f082eca770d26ba1cea02e84539fec0f44ce68d22bdda
5fde8275e582ed5cc045e53eddf640c52cc5715cd603ef32b5e4440038ed9fff9a666fc4fb7c4cf44970a1d71d8a755bd16b43aa6d784b351edd0ba1abac440941f7d6ad3cadf2ded02f
c3b62a017a69f115720818ae4bb8fc958f0655e125c76694d9d8b1878a97a6c4c3c158

c3 =
0x5f8b4c626d3d8812df42775016fb5d8fa2f8549a0371cf79ec3f57363a232645f25eb9ea300e93063f21835e1265f49938c0bd2388c8ef49a4f56b25535f38165fc19d40b7e592b1e
311629273af1a0e931aa150017813d8d12ff38dd563ebf78482837c74e3dce46767c165e14967d9e3a95e00286976dcdc95896e878bf08bb8a8a76742eee03184cb5b71a5cf97043e
7bfb601afd300927027c9264644a6aeb295db892f5d2e4db3873c3a088c3e75ad195aa458ce926494b411f3c366265270698bb8c90375d8114b80b36f97f35272bb573827e83b614
8a49a920aa772130e0baee29044e0bd7d891a3111a6acb9446f882659132f655af1def09496ea9
```

```
n1 =
0xb5FC37C1BE11A17DDFF95873938DDEC917F340792C00AA18E78D90D861718D449216CA710C2BF54513789F11250FEB9D1912DCC3A79F85F0B2E30DA44713
C1C728B76890D236E504D33D97DD2B7DD0A962F2E3475293CA511943A6D0953FD5FD5FDFCA7DF5C8F68217AA7B2172AE695186E8FF3AE7DFA9B1D00227B5D797
40EA79AA5ADF9BB7A06992BCDE4728C6B553F2A0C8593418535952B3889304AD1588B08EEB839A84E70547BF14B6C3CC3F9B43AE0B13562BA525DFEB0567F
7EF932C68035AB724A5D448E446B04C270748746BFA741F5D9E8B2F901C9171E267D564C318FD758A4F9C43BB797FC087F81C616B2065562EBEAD45A544DA04
B0F6303DA37
```

```
n2 =
0xCFFC46EC62D10A6342BEC1A120FE445682053786E32E0E687FC30C7BFFA450D19CDD81E16B53DF25B638CA165189A73A12AC6056596F0AC793EF6C624E8F
EFE6D01329E81F0EC735EE95C99AE92F2F4FF91F702BF3D3933C7D7D5247D96EDA3FE9694CD2B1C0E030AAB4FDD4881CC042CD0FC9EC6E5891EB9BFAAAD
33549DDC300181A645506EA546E5BB9A1DB9DB48360BC30AE14989525C27C02390808EAB58331756041635314F42BC302FB6B84ED846E69D71008474B9D876CF
355A6CB30BCB897D28523300357205BBD3B2B6ED9070C0F7C3241C226580ECE542817461913A775FAF687AD1ED7A24141357BBF4920B146F8806BECDF91C77
BE05B0671F
```

```
n3 =
0xAE7A7165280F3BAFB7BD0CCF968F4AEC3CBBE3266A16782825B515E5F641FFBAFCA39696917EC869BD5E023901955CDC22B7F8F7017013DD417D2418951F
3796476FBD9678AF696F25EC3DEB6FB45BD7699F60640FF9B9D6D474A649021917C53D112B8D0C70395511FC730723460504E856ECF1C9BA3714FFFF8FE965C
754AD72DA0B0710433590CDF097D2C1986D1FAE4212DC13127F190489D2B1526DC60C97888EE01C91905A449B7FF22DF6BB80CC6357122A3903BD827098400F
F75D620B29EEAF38B2368D82A7EE962E7A9CCF03D352B2AE3C79B3D5649ED4D1FC06F3AA132FA458FBD4C0D14065497519BF9BDF305D17BB653DAB575FA
CE48830115
```

e = 3

```
enc_flag =
binascii.unhexlify('497fd388ca7450b7a5c02aae250a3be60b367b4132819f139cf8bfc6c589542b80aa2c57d0c8c0d5da464be941928558a978b240f2f1b2f6b097f8b5a4a4fd512a
a8ace52b845a2c532cfae807cc81818aff6a920d218f199ff025b98fe7be37a60b2227bdcede92ab354a5dd49ac155dd49f50c69b95af17d40a7867f6826b7dfc1753327f458cbf641ca
080aa656861689933106335e52ba3c4b67fca725224795aa7e9da8f05f3b592903e070fa99')
iv = binascii.unhexlify('5f78552ba3d2f1568309953f73694305')
```

c4 = crt([c1, c2, c3], [n1, n2, n3])

```
aes_key = int(c4.nth_root(e))
aes_key = aes_key.to_bytes(-(aes_key.bit_length()//8), "big")
aes_key = aes_key[-32:]
```

decryptor = AES.new(aes_key, AES.MODE_CBC, iv = iv)

```
flag = decryptor.decrypt(enc_flag).decode()
flag = re.findall("(?:\n|\r|)(HTB{.*})(?:$| |\n)", flag)[0]
print(flag)
```

Recover the flag

By executing the script the flag will be printed to stdout.

CHEESE!

```
sage: load("baby_rebellion.sage")  
HTB{37.220464, -115.835938}
```