

 raccoons-team / ctf

 Watch 14  Star 22  Fork 14





 Code  Issues 0  Pull requests 0  Projects 0 Insights ▾


Branch: master ▾ ctf / 2016-03-07-boston-key-party-ctf / crypto\_5\_hmac\_crc /

Create new file Find file History

 Eterna1 links to files not as 'image'

Latest commit ff80d64 on Mar 8 2016

..		
 <a href="#">0c7433675c3c555afb77271d6a549bf5d941d2ab</a>	writeup for hmac_crc	a year ago
 <a href="#">crypto1.py</a>	writeup for hmac_crc	a year ago
 <a href="#">key.txt</a>	writeup for hmac_crc	a year ago
 <a href="#">readme.md</a>	links to files not as 'image'	a year ago

 [readme.md](#)

## Writeup for hmac\_crc CRYPTO (5)

We're trying a new mac here at BKP---HMAC-CRC. The hmac (with our key) of "zupe zecret" is '0xa57d43a032feb286'. What's the hmac of "BKPCTF"?

<https://s3.amazonaws.com/bostonkeyparty/2016/0c7433675c3c555afb77271d6a549bf5d941d2ab>

original script can be downloaded below

[0c7433675c3c555afb77271d6a549bf5d941d2ab](#)

We are given a hmac functions which takes hash function, message and key as input and gives sign of the message  
They give us the message and sign of this message using unknown key and we are supposed to sign another given message using the same key

code of signing function:

```
CRC_POLY = to_bits(65, (2**64) + 0xeff67c77d13835f7)
CONST = to_bits(64, 0xabaddeadbeef1dea)

def crc(msg):
    msg += CONST
    shift = 0
    while shift < len(msg) - 64:
        if msg[shift]:
            for i in range(65):
                msg[shift + i] ^= CRC_POLY[i]
            shift += 1
    return msg[-64:]

INNER = to_bits(8, 0x36) * 8
OUTER = to_bits(8, 0x5c) * 8

def hmac(h, key, msg):
    return h(xor(key, OUTER) + h(xor(key, INNER) + msg))
```

I was thinking about this algorithm and I came to the conclusion that when we flip one bit in key, all bits of the output depending on this bit also flip with no matter of other bits in key

another words when we have a message msg and two different keys: key1 and key2

$$\text{hmac}(h, \text{key1}, \text{msg}) \text{ xor } \text{hmac}(h, \text{key1 with flipped bit } x) = \text{hmac}(h, \text{key2}, \text{msg}) \text{ xor } \text{hmac}(h, \text{key2 with flipped bit } x)$$

this means that hmac is linear

We can gather all positions of changed bits in sign when bit at position x in key will change for the given message

I used gauss-jordan algorithm to compute which bits in key need to flip if I want flip one bit in sign at given position

I created my own pair key,sign for given message and I was looking which bits are different

when some bit was different I xored my key with bits I computed using gauss-jordan and I got needed key

my code is below

[crypto1.py](#)

put to file key.txt some 8B hex digit without '0x'

