devcraft.io

CTF write ups by vakzz

Project maintained by vakzz
Hosted on GitHub Pages — Theme by mattgraham

Integrity - 0CTF 2017 Quals

Mar 20, 2017

> *Just a simple scheme.*
>
> *nc 202.120.7.217 8221*

75 Pts, 135 solved, Crypto. integrity_f2ed28d6534491b42c922e7d21f59495.zip

This was a fun little challenge where you had two options:

- `r` register a username and get a secret token
- `l` login as a user using a secret

The goal was to login as `admin` and be given the flag, but they didn't let you register with that username as that would have been too easy.

The format of the secret was `IV || encrypted` and what was encrypted was `md5(pad(username)) || pad(username)`.

CBC works by encrypting the first block with a random IV, then uses the resulting ciphertext as the IV for encrypting the next block.

So when logging in, if instead of sending the initial IV we send the first encrypted block, the remaining blocks will be successfully decrypted.

In this case, the fist block is the md5 and remaining is the padded username. So if we register a username `md5(pad("admin")).digest() || admin` then our secret will be `IV || real md5 || admin md5 || admin`.

So all we have to do is remove the `IV`, then the `real md5` ciphertext will be used as the IV and everything will be decrypted correctly, passing the checksum and logging us in as admin.

```
#!/usr/bin/env python

from pwn import *
from hashlib import md5

BS = 16
```

```
pad = lambda s: s + (BS - len(s) % BS) * chr(BS - len(s) % BS)
unpad = lambda s : s[0:-ord(s[-1])]


name = md5(pad("admin")).digest() + "admin"

r = remote("202.120.7.217", 8221)

r.sendlineafter("or [l]ogin\n", "r")
r.sendline(name)
r.recvuntil("secret:\n")
secret = r.recvline().strip()

r.sendlineafter("or [l]ogin\n", "l")
r.sendline(secret[32:])
r.interactive()

"""
Welcome admin!
flag{Easy_br0ken_scheme_cann0t_keep_y0ur_integrity}
"""
```