Decoding RevC2 strings

By Jason Reaves

Published: 2024-12-19 · Archived: 2026-04-05 22:40:35 UTC

3 min read

Dec 19, 2024

By: Jason Reaves

Recently Zscaler reported on a component being leveraged in campaigns and related to Venom Spider[1]. Some of the recent samples we looked at appeared to have their strings obfuscated.

```
f598477a2cac439195ccf740bb38f50c2032a80be1cfeb5d34e1577f750c72bb
```

The sample builds out a table using hardcoded data:

```
@AB,0xffffffff,CDEFGHIJ456789:;<=KLMNOPQ,0x00,0x01,0x02,0x3,0x4,0x5,0x6, 0x7,0x8,0x9,0xa,0xb,0xc-0x1
```

The table aligns with decoding in base91+, the alphabet string in the binary looks more like base95 but for decoding we can just rip the decoding table out and use that to decode the strings. It also lets us quickly check if the alphabet remains static by trying it over lots of samples quickly.

Get Jason Reaves's stories in your inbox

Join Medium for free to get updates from this writer.

Remember me for faster sign in

After the base95 decoding the result is XOR decoded using a hardcoded string, using some base91 decode code in python we can create a decoder:

```
decode_table = [255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,

def decode(encoded_str):
    ''' Decode Base91 string to a bytearray '''
    v = -1
    b = 0
    n = 0
    out = bytearray()
```

```
    for strletter in encoded_str:
        c = decode_table[strletter]
        if c == 255:
            continue
        if(v < 0):
            v = c
        else:
            v += c*91
            b |= v << n
            n += 13 if (v & 8191)>88 else 14
            while True:
                out += struct.pack('B', b&255)
                b >>= 8
                n -= 8
                if not n>7:
                    break
            v = -1
    if v+1:
        out += struct.pack('B', (b | v << n) & 255 )
    return out

def decr(a):
    t = decode(a)
    key = bytearray(b'8dPtXeHtprHxQELs')
    for i in range(len(t)):
        t[i] ^= key[i%len(key)]
    return t
```

Decoding strings:

```
bytearray(b'cmd /c ')
bytearray(b'dir "%LocalAppData%\\Login Data" /s /b & dir "%appdata%\\Login Data" /s /b')
bytearray(b'dir "%LocalAppData%\\Cookies" /s /b & dir "%appdata%\\Cookies" /s /b')
bytearray(b'C:\\ProgramData\\Temp\\Cookies')
bytearray(b'C:\\ProgramData\\Temp')
bytearray(b'SELECT host_key, name, encrypted_value, path, is_secure, is_httponly, samesite, expires_
bytearray(b'new.ocx')
bytearray(b'ws://nopsec.]org:8082')
```

Another sample(c81d49c1907f27ea24a938ebbeb5f21bd30b4b186d99ec9c9458ce34f6bef72e):

```
bytearray(b'cmd /c ')
bytearray(b'dir "%LocalAppData%\\Cookies" /s /b & dir "%appdata%\\Cookies" /s /b')
bytearray(b'C:\\ProgramData\\Temp\\Cookies')
bytearray(b'C:\\ProgramData\\Temp')
```

```
bytearray(b'SELECT host_key, name, encrypted_value, path, is_secure, is_httponly, samesite, expires_
bytearray(b'module.ocx')
bytearray(b'ws://finatick.]com:8082')
```

Continuing to trace the samples back we found a few different versions, such as this one that writes the data to disk:

```
a10266c38c5f24201aa68cb3b0f7f24f44f4b5df635c5e2aebddb041b00d8a8f
```

IOCs:

```
jetmains.]com:8082
zoho-cloudfront.]com:8082
finatick.]com
nopsec.]org
```

Potential distro related:

```
cloudyvault.]org
cloudmort.]com
seopager.]xyz
gdrive.]rest
shadon.]net
sharesmydrive.]com
```

OCX filename checks:

```
xpr.ocx
new.ocx
brain.ocx
dWin.ocx
fer.ocx
iDriver.ocx
bajo.ocx
mojo.ocx
module.ocx
pp.ocx
```

References

1: https://www.zscaler.com/blogs/security-research/unveiling-revc2-and-venom-loader

2: https://thedfirreport.com/2024/12/02/the-curious-case-of-an-egg-cellent-resume/
