Doom cheat code encryption

From Just Solve the File Format Problem

Doom cheat code encryption was used in the original 1993 version of Doom to make the cheat codes a little harder for hackers to find, so they didn't appear in the raw binary code as plain ASCII characters. (Of course, the hackers found them anyway.) Twitter user @Foone (https://web.archive.org/web/20221114212355/https://twitter.com/Foone) described it in a 2019 thread.

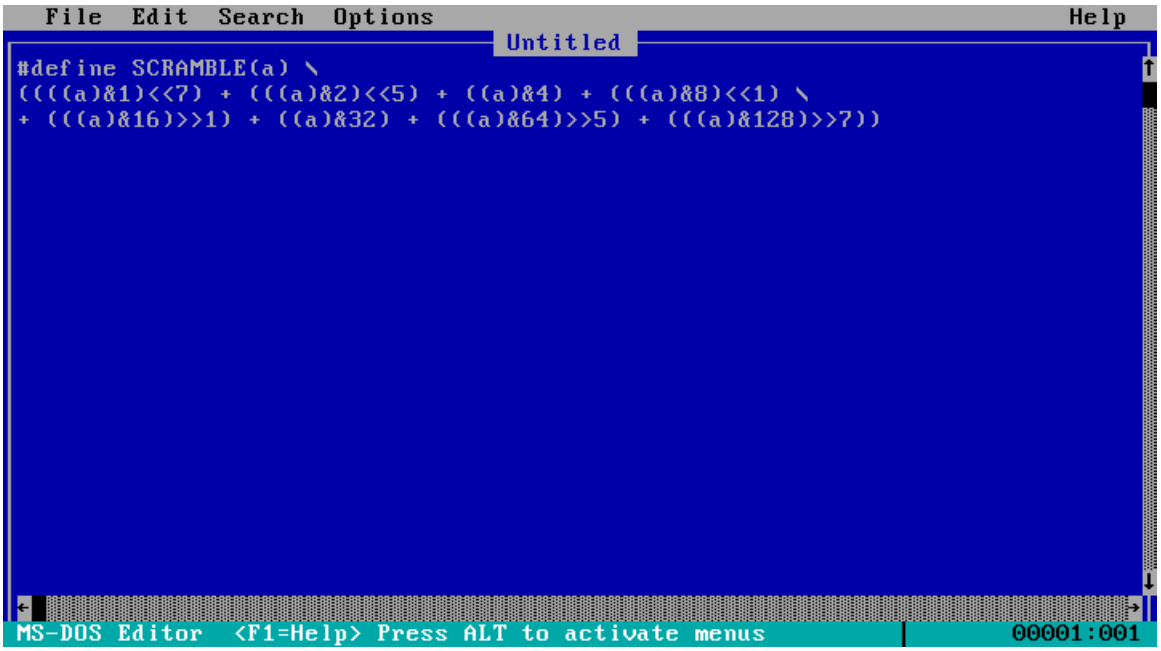
This low-grade encryption is done by shifting the bits of 8-bit numbers (which can represent single ASCII characters) which reverses the order of the bits except for those representing 4 and 32. The shifted values are stored in a lookup table in the Doom program.

File Format	
Name	Doom cheat code encryption
Ontology	▪ Electronic File Formats ▪ Encryption ▪ **Doom cheat code encryption**
Released	1993

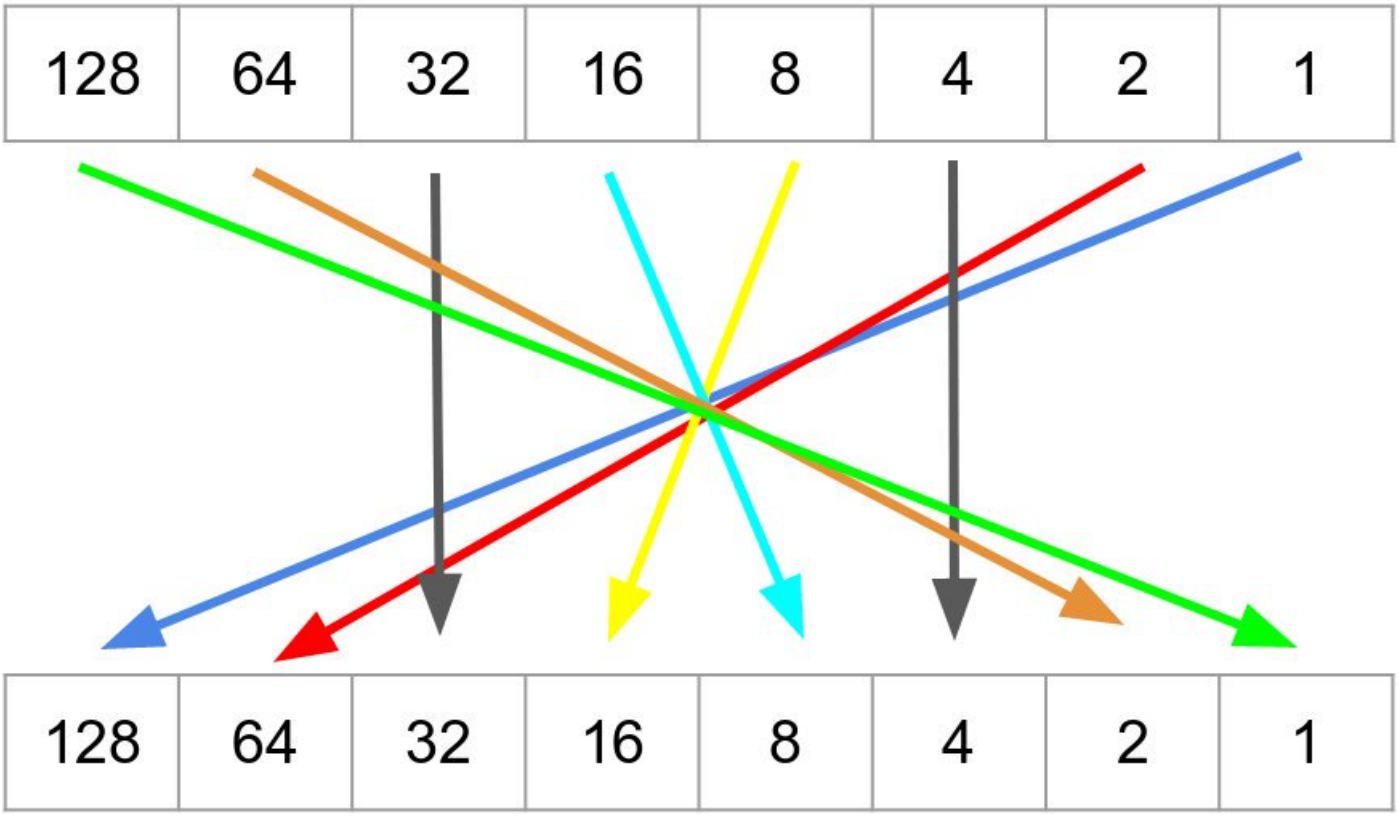
Details

As described by @Foone, who has allowed these descriptions to be released as CC0 (https://web.archive.org/web/20221114212355/https://twitter.com/Foone/status/1190656026342637569) so they can be used here:

> So Doom (1993) has a neat bit of encryption in it. It's not very strong encryption, but it's still encryption.
> And it's not used in any sort of way you'd normally expect: not copy protection, or multiplayer anti-cheat, or anti-tampering on saves... It's to slow down FAQs.
>
> So here's the code I'm talking about, the macro SCRAMBLE
> It looks annoyingly complicated but it's not, really.
> It's taking an 8-bit number and shifting around some of the bits.



> If you diagram out what's happening, it makes slightly (BUT ONLY SLIGHTLY) more sense.
> It kinda looks like they started with a a "reverse the order of these bits" function but then switched it so the 4 and 32 don't get switched, they just go straight through.



So, how is this code used?
Well, in m_cheat.c, it's used to build a lookup table which has all the values pre-encrypted, so it can quickly look them up later. Then, when you press a key, it translates it through this table:

```
int
cht_CheckCheat
( cheatseq_t*   cht,
  char          key )
{
    int i;
    int rc = 0;

    if (firsttime)
    {
        firsttime = 0;
        for (i=0;i<256;i++) cheat_xlate_table[i] = SCRAMBLE(i);
    }

    if (!cht->p)
        cht->p = cht->sequence; // initialize if first time

    if (*cht->p == 0)
        *(cht->p++) = key;
    else if
        (cheat_xlate_table[(unsigned char)key] == *cht->p) cht->p++;
    else
        cht->p = cht->sequence;

    if (*cht->p == 1)
        cht->p++;
    else if (*cht->p == 0xff) // end of sequence character
    {
        cht->p = cht->sequence;
        rc = 1;
    }

    return rc;
}
```

The thread goes on with more discussion of how these codes were used and discovered. It's worthwhile reading for people into this sort of trivia.

Links

- Twitter thread unrolled (https://web.archive.org/web/20221114212355/https://threadreaderapp.com/thread/1189249817492557826.html)

Categories: File Formats | Electronic File Formats | Encryption | Game data files | Id Software

---