A Mathematical Description of the CSS Cipher

by Charles M. Hannum <root@ihack.net>

Given:

- a 40-bit scrambling key expressed as a vector of five 8-bit bytes in k[0:4],
- a DVD sector expressed as a vector of 2^11 (2048) 8-bit bytes in s[0:2047],
- a function r() which reverses the bits in an 8-bit byte,
- two linear feedback shift registers l[] and m[], defined as follows,
- the sum of l[] and m[], called x[],
- an 8-bit PAL p[], defined as follows, and
- the temporary variables k'[], l'[] and m'[],

Then:

k'[n] = r(k[n] ^ s[84+n])

l[127] = k'[1] | (1 << 8) | (k'[0] << 9)

m[127] = k'[4] | (k'[3] << 8) | ((k'[2] & 00011111) << 16) | (1 << 21) | ((k'[2] & 11100000) << 17)

x[127] = 0

l'[n] = l[n] ^ (l[n] >> 14)

l[n=128:2047] = (l[n-1] >> 8) ^ (l'[n-1] << 9) ^ (l'[n-1] << 12) ^ (l'[n-1] << 15)

m'[n] = m[n] ^ (m[n] >> 3) ^ (m[n] >> 4) ^ (m[n] >> 12)

m[n=128:2047] = (m[n-1] >> 8) ^ (m'[n-1] << 17)

x[n=128:2047] = !(l[n] >> 9) + (m[n] >> 17) + (x[n-1] >> 8)

The PAL p[] is defined with the following equations, where a is the lowest input bit and A is the lowest output bit:

```
A = ! (a & b) ^ d
B = ! (e & f) ^ g
C = ! (A & B) ^ f
D = ! (A & B) ^ b
E = ! (a & b) ^ c
F = ! (e | f) ^ h
G = ! (E & F) ^ a
H = ! (E | F) ^ e
```

And the descrambled sector corresponding to k[] and s[] is:

```
s'[n=0:127] = s[n]
s'[n=128:2047] = p[s[n]] ^ x[n]
```

A translation of the previous into C code:

```
#include #include
int
main()
{
        unsigned char k[5], s[2048], kp[5], sp[2048];
        unsigned int l[2048], m[2048], x[2048], lp[2048], mp[2048];
        unsigned char r[256], p[256];
        int n;

        for (n = 0; n <= 255; n++) {
                int a, b, c, d, e, f, g, h, A, B, C, D, E, F, G, H;

                a = (n >> 0) & 1;
                b = (n >> 1) & 1;
                c = (n >> 2) & 1;
                d = (n >> 3) & 1;
                e = (n >> 4) & 1;
                f = (n >> 5) & 1;
                g = (n >> 6) & 1;
                h = (n >> 7) & 1;

                r[n] = (a << 7) | (b << 6) | (c << 5) | (d << 4) | (e << 3) |
                       (f << 2) | (g << 1) | (h << 0);

                A = (a & b) ^ d ^ 1;
                B = (e & f) ^ g ^ 1;
                C = (A & B) ^ f ^ 1;
                D = (A & B) ^ b ^ 1;
                E = (a & b) ^ c ^ 1;
                F = (e | f) ^ h ^ 1;
                G = (E & F) ^ a ^ 1;
                H = (E | F) ^ e ^ 1;

                p[n] = (H << 7) | (G << 6) | (F << 5) | (E << 4) | (D << 3) |
                       (C << 2) | (B << 1) | (A << 0);
        }

        for (read(0, k, 5); read(0, s, 2048); write(1, sp, 2048)) {
                for (n = 0; n <= 4; n++)
                        kp[n] = r[k[n] ^ s[84+n]];

                l[127] = kp[1] | (1 << 8) | (kp[0] << 9);
                m[127] = kp[4] | (kp[3] << 8) | ((kp[2] & 0x1f) << 16) |
                         (1 << 21) | ((kp[2] & 0xe0) << 17);
                x[127] = 0;

                for (n = 128; n <= 2047; n++) {
                        lp[n-1] = l[n-1] ^ (l[n-1] >> 14);
                        l[n] = (l[n-1] >> 8) ^ (lp[n-1] << 9) ^
                               (lp[n-1] << 12) ^ (lp[n-1] << 15);
                        l[n] &= 0x1ffff;

                        mp[n-1] = m[n-1] ^ (m[n-1] >> 3) ^ (m[n-1] >> 4) ^
                                  (m[n-1] >> 12);
                        m[n] = (m[n-1] >> 8) ^ (mp[n-1] << 17);
                        m[n] &= 0x1ffffff;

                        x[n] = (0xff & ~(l[n] >> 9)) + (m[n] >> 17) +
                               (x[n-1] >> 8);
                }

                for (n = 0; n <= 127; n++)
                        sp[n] = s[n];
                for (n = 128; n <= 2047; n++)
                        sp[n] = p[s[n]] ^ x[n];
        }
```

```
}
```

Dave Touretzky
Last modified: Tue Mar 13 03:04:41 EST 2001