Find EMM keys 84 times faster

This document describes how to find PowerVu EMM keys 84 times faster with the same hardware.

24.12.2014 (Version 1.0) up-to-date version http://colibri-dvb.info => PowerVu
Colibri <colibri.dvb@gmail.com>

An other user and I have independently found a way to do a very high speed improvement, compared to the way that was described in the PowerVu_management_keys_hacked.pdf.

This was needed because I have found only EMM keys with many subscribers with the old method, but I was unable to find a key with only 263 subscribers, So an improvement was needed to find the keys in a shorter time.

Here an example of a decryption of test data with a fictive key. You can see the internal shift register state (= pattern key) before each single bit decryption.

BYTE Encrypted_Bytes[] = {0xE5, 0x59, 0x58, 0xE4, 0xE3, 0x22, 0x95, 0xE1, 0x0E};

BYTE Key[] = {0x43, 0x4F, 0x4C, 0x49, 0x42, 0x52, 0x49};//same as {'C', 'O', 'L', 'I', 'B', 'R', 'I'};

It will produce 9 * 0x00 bytes.

```
[00] key: 43 4F 4C 49 42 52 49 = 01000011010011110100110001001001010000100101001001001001 / enc 1
[01] key: A1 A7 A6 2C A1 29 24 = 10100001101001111010011000101100101000010010100100100100 / enc 1
[02] key: 50 D3 D3 16 50 94 92 = 01010000110100111101001100010110010100001001010010010010 / enc 1
[03] key: 28 69 E9 8B 28 4A 49 = 00101000011010011110100110001011001010000100101001001001 / enc 0
[04] key: 94 34 F4 CD 94 25 24 = 10010100001101001111010011001101100101000010010100100100 / enc 0
[05] key: 4A 1A 7A 66 CA 12 92 = 01001010000110100111101001100110110010100001001010010010 / enc 1
[06] key: 25 0D 3D 33 65 09 49 = 00100101000011010011110100110011011001010000100101001001 / enc 0
[07] key: 92 86 9E 91 B2 84 A4 = 10010010100001101001111010010001101100101000010010100100 / enc 1
[08] key: 49 43 4F 48 D9 42 52 = 01001001010000110100111101001000110110010100001001010010 / enc 0
[09] key: 24 A1 A7 A4 6C A1 29 = 00100100101000011010011110100100011011001010000100101001 / enc 1
[0A] key: 92 50 D3 DA 36 50 94 = 10010010010100001101001111011010001101100101000010010100 / enc 0
[0B] key: 49 28 69 ED 1B 28 4A = 01001001001010000110100111101101000110110010100001001010 / enc 1
[0C] key: 24 94 34 F6 8D 94 25 = 00100100100101000011010011110110100011011001010000100101 / enc 1
[0D] key: 92 4A 1A 73 46 CA 12 = 10010010010010100001101001110011010001101100101000010010 / enc 0
[0E] key: 49 25 0D 39 A3 65 09 = 01001001001001010000110100111001101000110110010100001001 / enc 0
[0F] key: A4 92 86 94 D1 B2 84 = 10100100100100101000011010010100110100011011001010000100 / enc 1
[10] key: 52 49 43 4A 68 D9 42 = 01010010010010010100001101001010011010001101100101000010 / enc 0
[11] key: 29 24 A1 A5 34 6C A1 = 00101001001001001010000110100101001101000110110010100001 / enc 1
[12] key: 94 92 50 DA 9A 36 50 = 10010100100100100101000011011010100110100011011001010000 / enc 0
[13] key: 4A 49 28 6D 4D 1B 28 = 01001010010010010010100001101101010011010001101100101000 / enc 1
[14] key: 25 24 94 36 A6 8D 94 = 00100101001001001001010000110110101001101000110110010100 / enc 1
[15] key: 12 92 4A 1B 53 46 CA = 00010010100100100100101000011011010100110100011011001010 / enc 0
[16] key: 09 49 25 0D A9 A3 65 = 00001001010010010010010100001101101010011010001101100101 / enc 0
[17] key: 84 A4 92 8E D4 D1 B2 = 10000100101001001001001010001110110101001101000110110010 / enc 0
[18] key: 42 52 49 47 6A 68 D9 = 01000010010100100100100101000111011010100110100011011001 / enc 1
[19] key: A1 29 24 AB B5 34 6C = 10100001001010010010010010101011101101010011010001101100 / enc 1
[1A] key: 50 94 92 55 DA 9A 36 = 01010000100101001001001001010101110110101001101000110110 / enc 1
[1B] key: 28 4A 49 2A ED 4D 1B = 00101000010010100100100100101010111011010100110100011011 / enc 0
[1C] key: 94 25 24 9D 76 A6 8D = 10010100001001010010010010011101011101101010011010001101 / enc 0
[1D] key: CA 12 92 46 BB 53 46 = 11001010000100101001001001000110101110110101001101000110 / enc 1
[1E] key: 65 09 49 23 5D A9 A3 = 01100101000010010100100100100011010111011010100110100011 / enc 0
[1F] key: B2 84 A4 99 AE D4 D1 = 10110010100001001010010010011001101011101101010011010001 / enc 0
[20] key: D9 42 52 44 D7 6A 68 = 11011001010000100101001001000100110101110110101001101000 / enc 1
[21] key: 6C A1 29 22 6B B5 34 = 01101100101000010010100100100010011010111011010100110100 / enc 1
[22] key: 36 50 94 91 35 DA 9A = 00110110010100001001010010010001001101011101101010011010 / enc 1
[23] key: 1B 28 4A 48 9A ED 4D = 00011011001010000100101001001000100110101110110101001101 / enc 0
[24] key: 8D 94 25 2C 4D 76 A6 = 10001101100101000010010100101100010011010111011010100110 / enc 0
[25] key: 46 CA 12 96 26 BB 53 = 01000110110010100001001010010110001001101011101101010011 / enc 0
[26] key: A3 65 09 43 13 5D A9 = 10100011011001010000100101000011000100110101110110101001 / enc 1
[27] key: D1 B2 84 A9 89 AE D4 = 11010001101100101000010010101001100010011010111011010100 / enc 1
[28] key: 68 D9 42 54 C4 D7 6A = 01101000110110010100001001010100110001001101011101101010 / enc 0
[29] key: 34 6C A1 2A 62 6B B5 = 00110100011011001010000100101010011000100110101110110101 / enc 0
[2A] key: 9A 36 50 9D 31 35 DA = 10011010001101100101000010011101001100010011010111011010 / enc 1
[2B] key: 4D 1B 28 4E 98 9A ED = 01001101000110110010100001001110100110001001101011101101 / enc 0
[2C] key: A6 8D 94 2F 4C 4D 76 = 10100110100011011001010000101111010011000100110101110110 / enc 0
[2D] key: 53 46 CA 17 A6 26 BB = 01010011010001101100101000010111101001100010011010111011 / enc 0
[2E] key: A9 A3 65 03 D3 13 5D = 10101001101000110110010100000011110100110001001101011101 / enc 1
[2F] key: D4 D1 B2 89 E9 89 AE = 11010100110100011011001010001001111010011000100110101110 / enc 0
[30] key: 6A 68 D9 44 F4 C4 D7 = 01101010011010001101100101000100111101001100010011010111 / enc 1
[31] key: B5 34 6C AA 7A 62 6B = 10110101001101000110110010101010011110100110001001101011 / enc 0
[32] key: DA 9A 36 5D 3D 31 35 = 11011010100110100011011001011101001111010011000100110101 / enc 0
[33] key: ED 4D 1B 26 9E 98 9A = 11101101010011010001101100100110100111101001100010011010 / enc 1
[34] key: 76 A6 8D 93 4F 4C 4D = 01110110101001101000110110010011010011110100110001001101 / enc 0
[35] key: BB 53 46 C1 A7 A6 26 = 10111011010100110100011011000001101001111010011000100110 / enc 1
[36] key: 5D A9 A3 60 D3 D3 13 = 01011101101010011010001101100000110100111101001100010011 / enc 0
[37] key: AE D4 D1 B8 69 E9 89 = 10101110110101001101000110111000011010011110100110001001 / enc 1
```

```
[38] key: D7 6A 68 D4 34 F4 C4 = 11010111011010100110100011010100001101001111010011000100 / enc 1
[39] key: 6B B5 34 6A 1A 7A 62 = 01101011101101010011010001101010000110100111101001100010 / enc 1
[3A] key: 35 DA 9A 35 0D 3D 31 = 00110101110110101001101000110101000011010011110100110001 / enc 1
[3B] key: 9A ED 4D 12 86 9E 98 = 10011010111011010100110100010010100001101001111010011000 / enc 0
[3C] key: 4D 76 A6 89 43 4F 4C = 01001101011101101010011010001001010000110100111101001100 / enc 0
[3D] key: 26 BB 53 44 A1 A7 A6 = 00100110101110110101001101000100101000011010011110100110 / enc 0
[3E] key: 13 5D A9 A2 50 D3 D3 = 00010011010111011010100110100010010100001101001111010011 / enc 0
[3F] key: 89 AE D4 D9 28 69 E9 = 10001001101011101101010011011001001010000110100111101001 / enc 1
[40] key: C4 D7 6A 64 94 34 F4 = 11000100110101110110101001100100100101000011010011110100 / enc 0
[41] key: 62 6B B5 32 4A 1A 7A = 01100010011010111011010100110010010010100001101001111010 / enc 0
[42] key: 31 35 DA 99 25 0D 3D = 00110001001101011101101010011001001001010000110100111101 / enc 0
[43] key: 98 9A ED 44 92 86 9E = 10011000100110101110110101000100100100101000011010011110 / enc 0
[44] key: 4C 4D 76 A2 49 43 4F = 01001100010011010111011010100010010010010100001101001111 / enc 1
[45] key: A6 26 BB 59 24 A1 A7 = 10100110001001101011101101011001001001001010000110100111 / enc 1
[46] key: D3 13 5D A4 92 50 D3 = 11010011000100110101110110100100100100100101000011010011 / enc 1
[47] key: E9 89 AE DA 49 28 69 = 11101001100010011010111011011010010010010010100001101001 / enc 0
```

The old way was to store the encrypted pattern that produce zero bits (E5, 59, 58, …).
If you find the corresponding pattern key `[00] key: 43 4F 4C 49 42 52 49` then you can search the original EMM block with this encrypted pattern and can try to move backward the shift register by 16 bits to get the EMM key.

The original encrypted patten is E55958E4 … in binary form it's
1110 0101 0101 1001 0101 1000 1110 0100 …

If we skip the first encrypted bit and take the remaining data as pattern it will look like:
(E55958E4 …) << **1** =
110 0101 0101 1001 0101 1000 1110 0100 ...
This pattern will also decrypt to zero bits and the corresponding key is `[01] key: A1 A7 A6 2C A1 29 24`
The calculate the EMM key from this pattern key we need to move backward the shift register by 16+**1** bits.

If we skip two encrypted bits and take the remaining data as pattern it will look like:
(E55958E4 …) << **2** =
10 0101 0101 1001 0101 1000 1110 0100 ...
This pattern will also decrypt to zero bits and the corresponding key is `[02] key: 50 D3 D3 16 50 94 92`
The calculate the EMM key from this pattern key we need to move backward the shift register by 16+**2** bits.

If we skip tree encrypted bits and take the remaining data as pattern it will look like:
(E55958E4 …) << **3** =
0 0101 0101 1001 0101 1000 1110 0100 ...
This pattern will also decrypt to zero bits and the corresponding key is `[03] key: 28 69 E9 8B 28 4A 49`
The calculate the EMM key from this pattern key we need to move backward the shift register by 16+**3** bits.

So we can extract 84 patterns that decrypts to zero instead only one pattern per EMM block.
I have also used a 7 byte long pattern instead of a 9 byte pattern.
Because I have more pattern to store in the look-up table I have increased the size.
Instead of storing 256*256*256 patterns 9 byte each (144 MB table size) it can store 256*256*256*2 patterns 7 byte each (224 MB table size).

Because you only need to encrypt 7 instead of 9 bytes the brute force is a little bit faster.
By using 84 instead of 1 pattern per EMM block, you will find a key 84 times faster.