

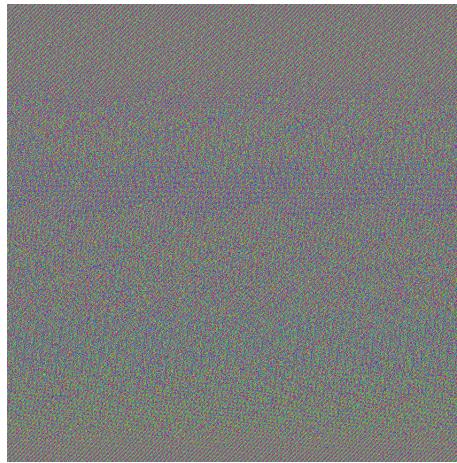
Ange Albertini's extensions to the ECB Penguin.

### 3 ECB as an Electronic Coloring Book

by Philippe Teuwen

Hey boys and girls, remember Natalie and Ben's warnings in PoC||GTFO 4:13 about ECB? Forbidden things are attractive, I know, I was young too. Let's explore that area together so that you'll have fun and you'll always remember not to use ECB later in your grown-up life.

But first of all let me clarify one thing: the ubiquitous ECB penguin is a kind of a fraud, brandished like a scarecrow! The reality when you get an encrypted image in ECB mode is that you've no clue of its characteristics, its size, its pixel representation. Let's take another example than the penguin (as the source image of this fraud seems to be lost forever). A wrong guess, such as assuming a square format, will render just a meaningless bunch of static.



So to get the penguin back, the penguin's author cheated and encrypted only the pixel values, but not the description of the image, such as its size. Moreover he probably tried different keys until he got the tuxedo as black as possible as he has no control on the encrypted result.

Does it mean ECB is not that bad? Don't get me wrong, ECB is a very bad way to encrypt and we'll blow it apart. But what's ECB? No need to understand the underlying crypto, just that the image is being sliced in small pieces—sixteen bytes wide in case of AES-ECB—and each piece is replaced by random garbage. Identical pieces are replaced by the same random data and if two pieces are different their respective encrypted versions are too. That's why we can distinguish the penguin.

But we can do much better; instead of displaying directly the mangled pixels we can paint them! We know that identical blocks of random data represent the encrypted version of the same initial block of color, so let's pick a color ourselves and paint over those similar pieces. That's what this little program does. You'll find it as `ElectronicColoringBook.py` by unzipping this PDF.<sup>3</sup> It also tries to guess the right ratio by checking which one will give columns of pixels as coherent as possible.

```
$ ElectronicColoringBook.py test.bin
```



Already better! The lines are properly aligned but the image is too flat. That's because we painted each byte as one pixel but the original image was probably created with three bytes per pixel, so let's fix that.

---

<sup>3</sup><https://github.com/doegox/ElectronicColoringBook>

```
$ ElectronicColoringBook.py test.bin -pixelwidth=3
```



As we don't know the original colors, the tool is choosing some randomly at each execution. Now that the ratio and pixel width are correct we can observe vertical stripes. That's what happens when you can't have an exact number of pixels in each block and that's exactly the case here. We guessed that each pixel requires three bytes and the blocks are 16-byte wide so if some pixels of the same color—let's say #AABBCC—are side by side we get three types of encrypted blocks.

```
1 AABCCAABBCAABBCCAABBCCAA --> 81E49040C91E64A8F2EB52EB313EADF4
2 BBCCAABBCAABBCCAABBCCAABBCCAA --> 769B3981E49040C9164A83B6CBFB12BF
3 CCAABBCAABBCAABBCCAABBCCAA --> 12B4502017A19C0EB313EADF47638FB2
4 AABCCAABBCAABBCCAABBCCAA --> 81E49040C91E64A8F2EB52EB313EADF4
5 BBCAABBCAABBCCAABBCCAABBCCAA --> 769B3981E49040C9164A83B6CBFB12BF
etc
```

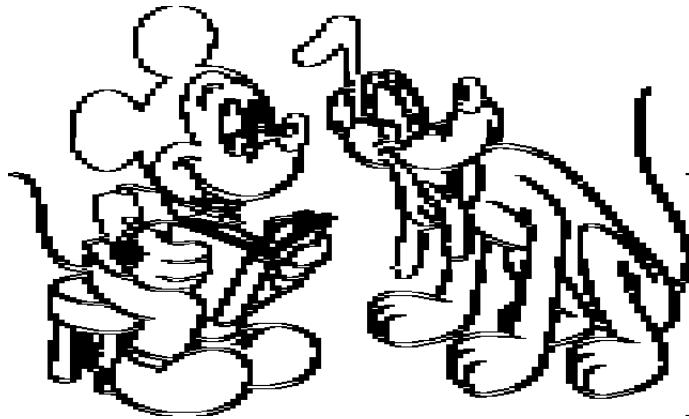
So we've got three types of encrypted data for the same color, repeating over and over. Still one last complication: Pluto's tail is visible on the left of the image, because before the encrypted pixels there is the encrypted file header. So we'll apply a small offset to skip it, and as before we'll group blocks by three.

```
$ ElectronicColoringBook.py test.bin -p 3 -groups=3 -offset=1
```



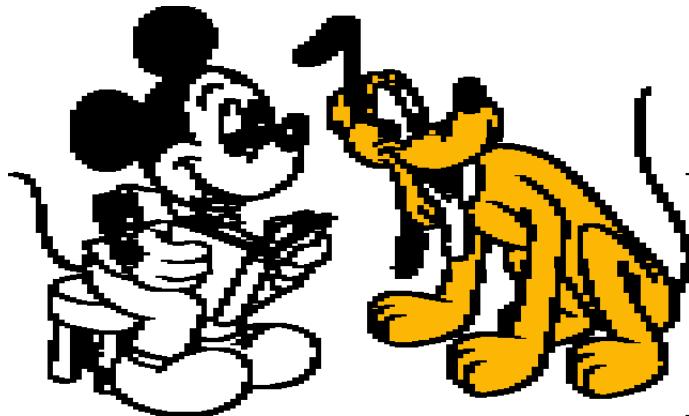
And now let's make it a real coloring book by choosing those colors ourselves! We'll draw the ten most frequent colors in white (#ffffff) and the remaining blocks, which typically contain all kinds of transitions from one color area to another one, in black (#000000).

```
$ ElectronicColoringBook.py test.bin -p 3 -g 3 -o 1 -palette=\n'#####\######\######\######\######\######\######\######\#000000',
```



Kids, those colors are encoded with their RGB values. If this is confusing, ask the geekiest of your parents; she can help you. Colors are sorted by largest areas, so let's keep the white color for the background. Let's paint Pluto in orange (#fcb604) and Mickey's head in black.

```
$ ElectronicColoringBook.py test.bin -p 3 -g 3 -o 1 -P \n'#####\#fcb604#000000#####\######\######\######\######\#000000',
```



If you don't know which area corresponds to which color in the palette, just try it out with a flashy color. Eventually, we wind up with something like this.

```
$ ElectronicColoringBook.py test.bin -p 3 -g 3 -o 1 -P \n'#####\#fcb604#000000#f9fa00#fccdcc#fc1b23#a61604#a61604#fc8591#97fe37#000000',
```

Note to copyright owners:

We were careful to disclose only images encrypted with AES-256 and a random key that was immediately destroyed. This should be safe enough, right?



Much better than the ECB penguin, don't you think? So remember that ECB should really stand for "Electronic Coloring Book." They should therefore should be only used by kids to have fun, never by grown-ups for a serious job!

Maybe Dad is wondering why we didn't use a picture of Lenna as in any decent scientific paper about image processing? Tell him simply that it's for a coloring book, not Playboy! There are more complex examples and explanations in the project directory. It's even possible to colorize other things, such as binaries or XORed images!

When no one has your floppy disks in stock...  
here's a new four letter word to use:

The word is KYBE. Because KYBE can ship any model floppy disk, data cassette or mag card in only two days. You'll get the same high performance products we've built for OEM's for years. Consistent quality media that meets the most demanding specifications. The full line is competitively priced, backed by an unconditional 90 day warranty and inventoried for fast delivery.

Call toll free (800) 225-8715.  
Dealer inquiries invited

KYBE  
Dennison KYBE Corporation  
132 Calvary Street, Waltham, Mass. 02154  
Tel. (617) 899-0012; Telex 94-0179  
Outside Mass. call toll free (800) 225-8715  
Offices & representatives worldwide