Computing Is Pretty Strange

Steganography: Something Amazing To Do with Bits

Lawrence Snyder
University of Washington, Seattle

A Photo from *Spokesman Review*

Steganography

- The process of hiding information
- Two Greek roots meaning:
  "stego" == "roof" "stega" == "cover"

Why Hide Information?

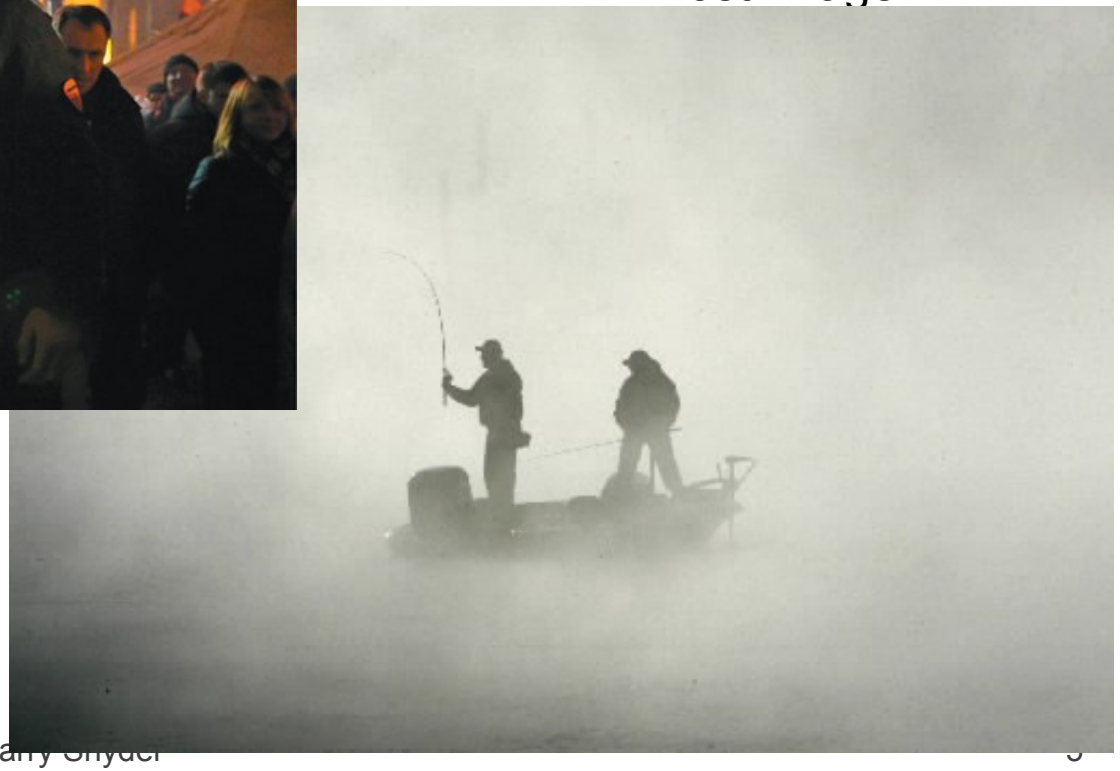
- Most common reason to hide information is to avoid being caught with it
  - Military and spy documents
  - Repressive governments restricting news/info
  - Avoid others "snooping" – privacy
- Hiding is different than encryption … uses the fact that the searcher doesn't know it's there

Illustrate A Way To Do It

- The Plan …
  - hide “subversive” protest photo in “calendar art”

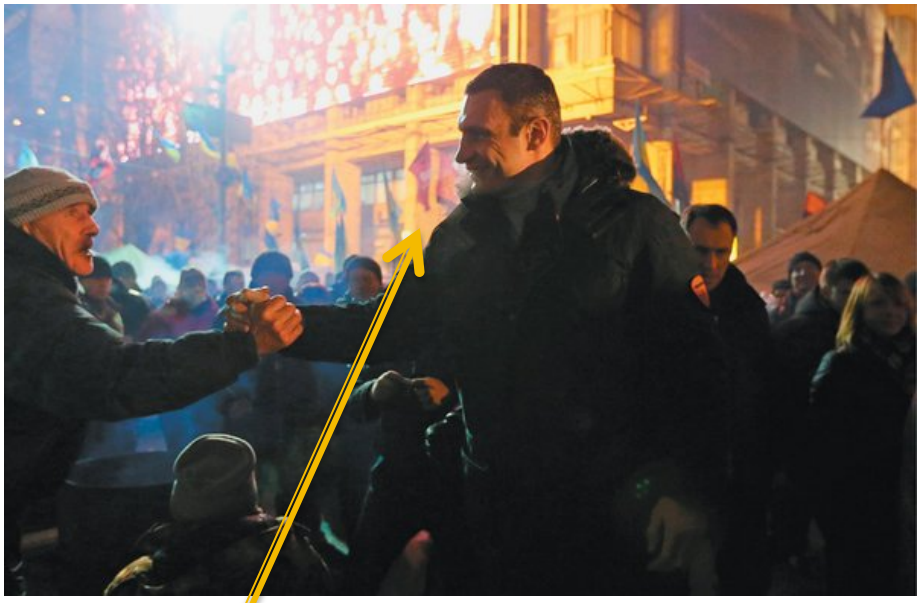
Guest Image

Host Image



Step 1: Reduce Bits of Guest

- We don't need all of the bits in RGB to get a decent picture



All bits

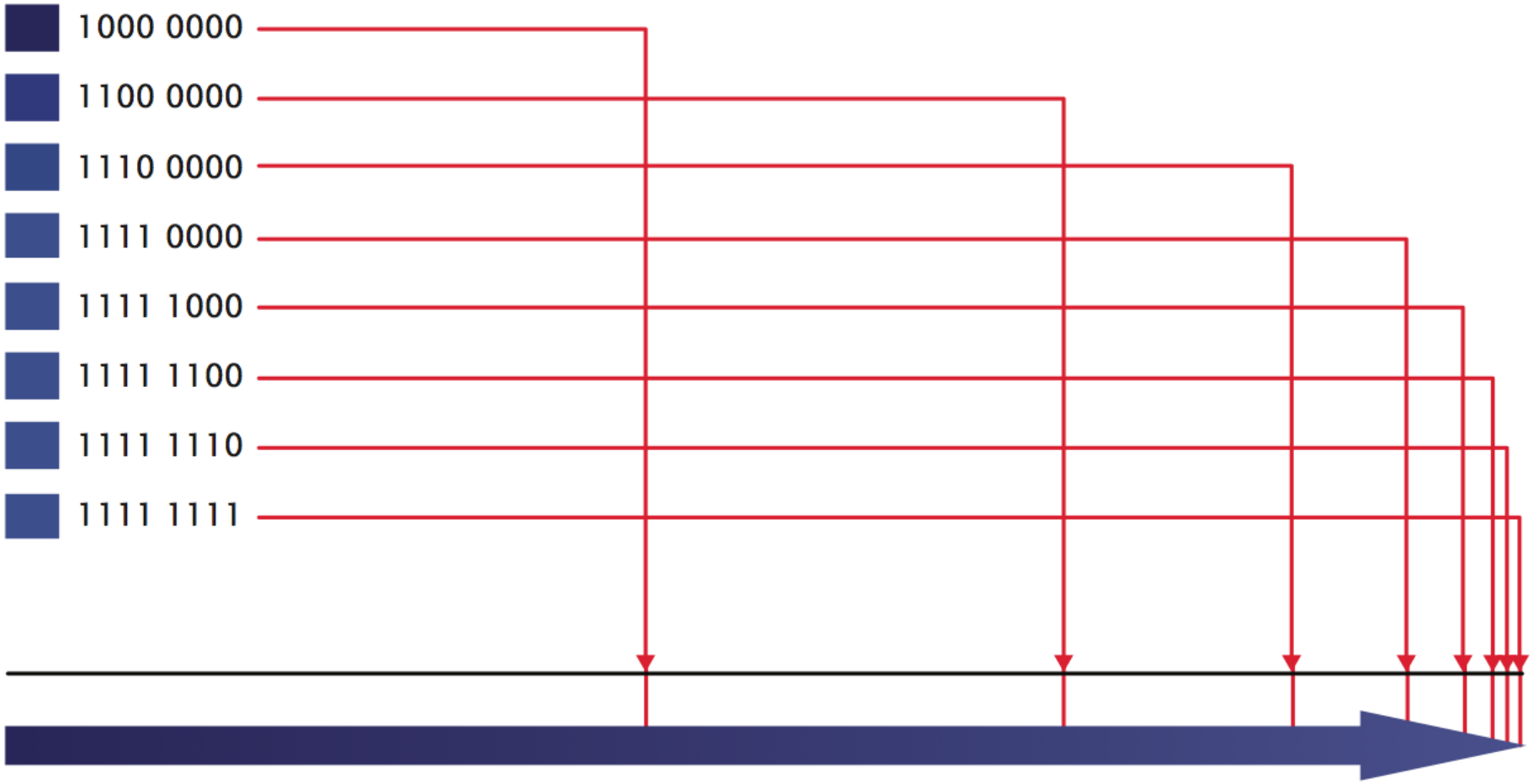
1011 0100 1101 0011 0001 1100

Left 2 bits of each color only

1011 0100 1101 0011 0001 1100

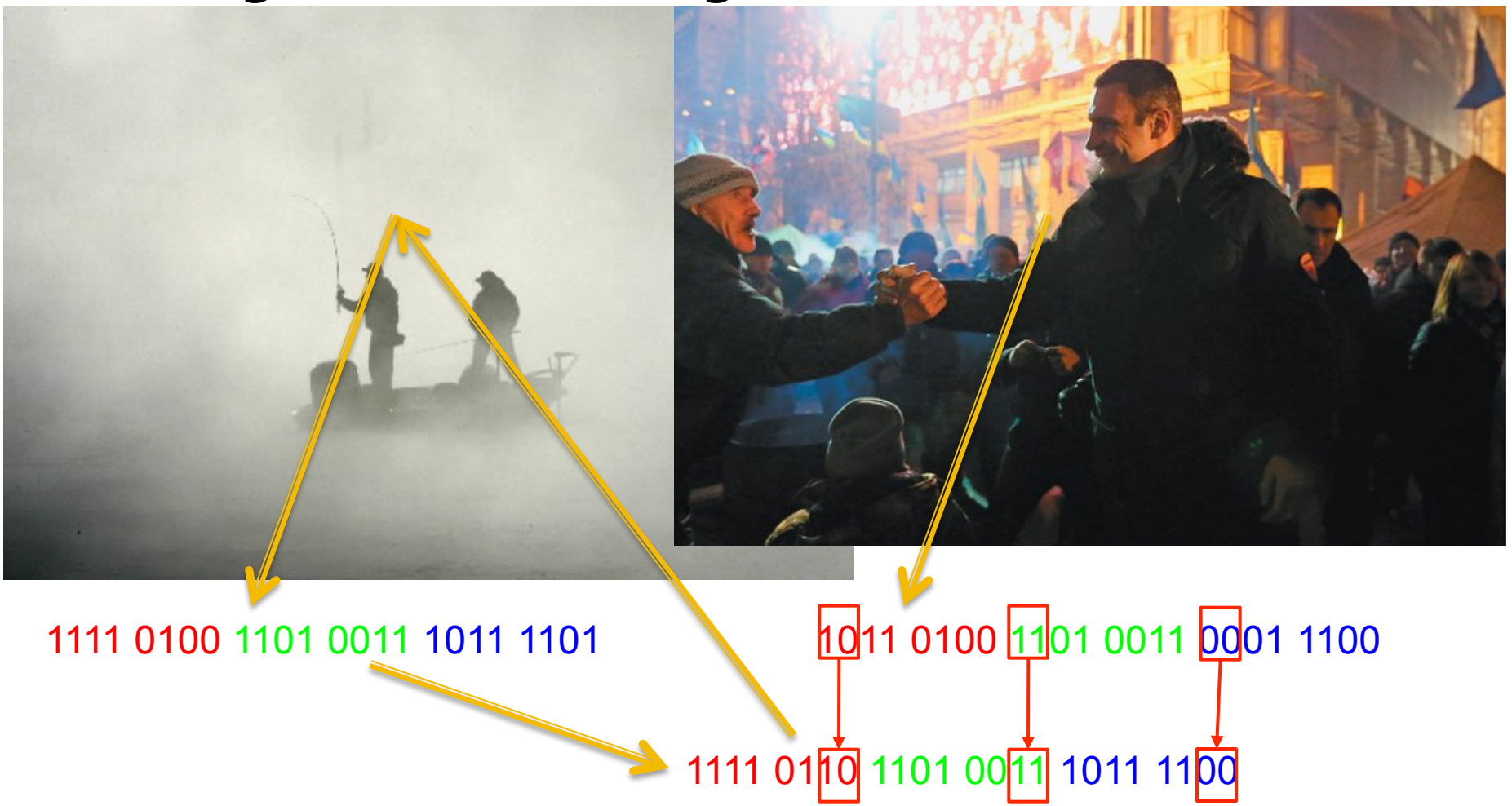
1000 0000 1100 0000 0000 0000

Each Bit Adds Another 1/2



Step 2: Replace Bits In Host

- Put guest bits into right 2 bits of host



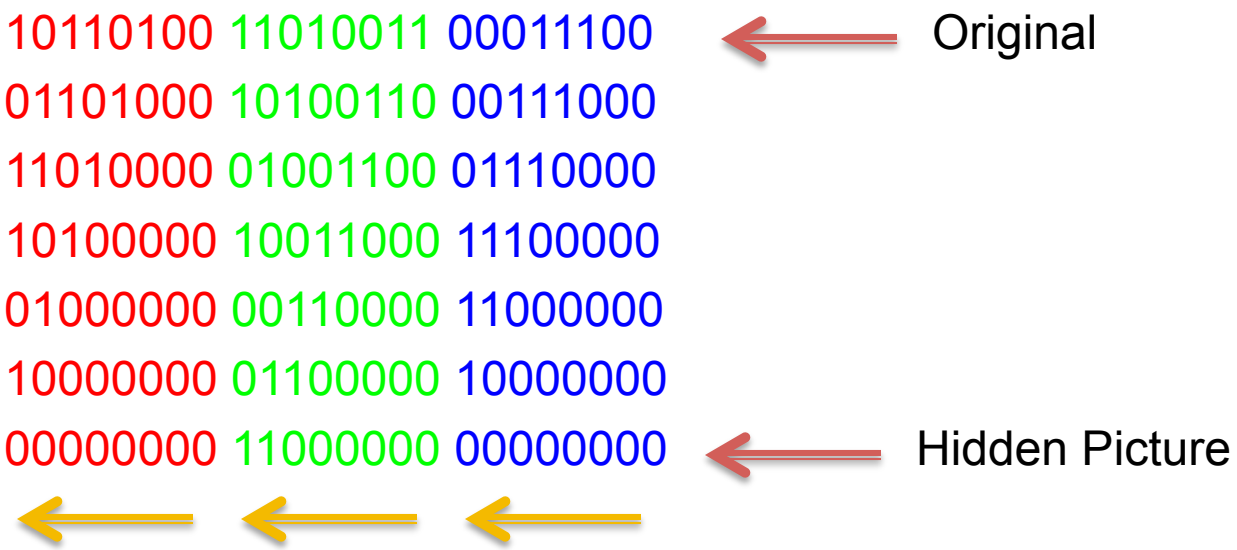
Compare foglg.jpg with stegFog.png

foglg.jpg

stegFog.png

Shift Bits To Reveal Hidden Pic

Each of the colors is shifted left 1 bit at a time



… and then we'll see the details

Processing Code For Guest→Host

```
PImage crowd, fog;
int i = 0;
int srcw=600;
int srch=405;
int wid=600;
int hi=389;
color c, cprime;

void setup( ) {
  size(srcw, srch);
  crowd = loadImage("ukraine.jpg");
  fog = loadImage("foglg.jpg");
  image(fog,0,0);
  for (int i=0; i<srcw; i++){
    for(int j=0; j<srch; j++) {
      c = get(i,j);
      if (i<wid && j<hi) {
        cprime=crowd.get(i,j);
        cprime=color(4*(int(red(c))/4) + (int(red(cprime))/64),
                     4*(int(green(c))/4) + (int(green(cprime))/64),
                     4*(int(blue(c))/4) + (int(blue(cprime))/64));
        set(i,j, cprime);
      } else {
        set(i,j,c);
      }
    }
  }
}
```

Encoding Code

```
void draw( ) {
  if (mousePressed) {
    saveFrame("stegFog.png");
  }
}
```

Code To Save Result on Click

Setup to Hide The Ukraine Pic

```
PImage crowd, fog;
int i = 0;
int srcw=600;
int srch=405;
int wid=600;
int hi=389;
color c, cprime;

void setup( ) {
  size(srcw, srch);
  crowd = loadImage("ukraine.jpg");
  fog = loadImage("foglg.jpg");
```

Embedding of Ukraine Pic

```
for (int i=0; i<srcw; i++){
  for(int j=0; j<srch; j++) {
    c = get(i,j);
    if (i<wid && j<hi) {
      cprime=crowd.get(i,j);
      cprime=color(4*(int(red(c))/4) + (int(red(cprime))/64),
                   4*(int(green(c))/4) + (int(green(cprime))/64),
                   4*(int(blue(c))/4) + (int(blue(cprime))/64));
      set(i,j, cprime);
    } else {
      set(i,j,c);
    }
  }
 }
}
```

How Does It Work

- After the pictures are loaded 10110100 11010011 00011100

```
cprime=color(4*(int(red(c))/4) + (int(red(cprime))/64),
             4*(int(green(c))/4) + (int(green(cprime))/64),
             4*(int(blue(c))/4) + (int(blue(cprime))/64));
```

Clear right 2 bits of host

101101xx 110100xx 000111xx

Extract left 2 bits of guest

New combined color

10110100 11010011 00011100

Code To Extract Image

```
void setup( ) {
  size(srcw, srch);
  fog = loadImage("stegFog.png");
  image(fog,0,0);
}

void draw( ) {
   if (step == 1) {
     for (int i=0; i<srcw; i++){
     for(int j=0; j<srch; j++) {
      c = get(i,j);
      if (i<wid && j<hi) {
          cprime=color(((int(red(c))*2)%256),
                       ((int(green(c))*2)%256),
                       ((int(blue(c))*2)%256));
          set(i,j, cprime);
      } else {
        set(i,j,c);
      }
     }
    }
    step = 0;
   }
}
```

How Does It Work

- Read in the file, and then on key press, shift the bits left one position

```
for (int i=0; i<srcw; i++){
for(int j=0; j<srch; j++) {
 c = get(i,j);
 if (i<wid && j<hi) {
    cprime=color(((int(red(c))*2)%256),
                 ((int(green(c))*2)%256),
                 ((int(blue(c))*2)%256));
    set(i,j, cprime);
 } else {
    set(i,j,c);
 }
}
}
```

Just Do It! Again

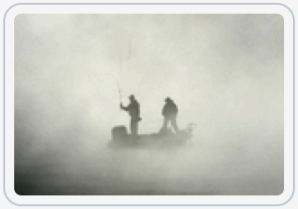
How Much Is Coded Like Original?

- Run A Test … www.tineye.com

Check The "Steganized" File

TinEye
Reverse Image Search

Steganized



PNG, 600x405, 130.7 KB

5 Results

Searched over 4.704 billion images in 0.365 seconds.

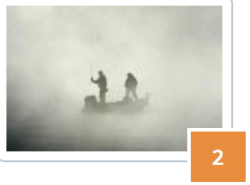
for file: stegFog.png

- These results expire in **72 hours**. Why?
- Share a success story!
- TinEye is free to use for non-commercial purposes.

Download the official TinEye extension for Firefox with right-click functionality!

Sort by:

- Best Match
- Most Changed
- Biggest Image
- Newest (new)
- Oldest



2

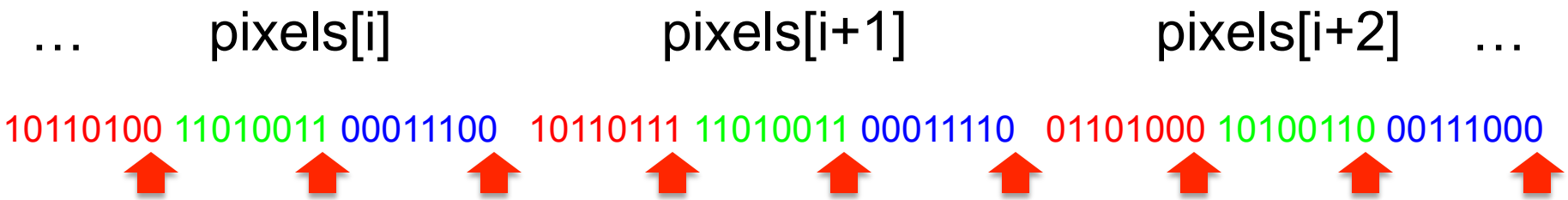
Compare | Link
JPEG Image
700x474, 14.8 KB

www.milliyet.com.tr
2.jpg
www.milliyet.com.tr/content/galeri/ye...
Crawled on 2008-04-18

forum.shiftdelete.net
2.jpg
forum.shiftdelete.net/sdn-magazin/gun...
Crawled on 2008-02-28

Alternative Way of Using Pictures

- Our approach used a pixel-for-pixel encoding, which hid a decent, but not perfect, image
- Another approach is just use the last bit in each pixel to encode any string of bits …
  - Recall the pixels[ ] array in Processing



  - Use last bit of each color in each pixel
    - Now they encode: 010110000 but it could be anything
    - Or just use the last bit in the blue byte in each pixel

Fini!

- Steganography can be used extensively – there are many places to hide information
- Tomorrow, you'll hide a picture, too.