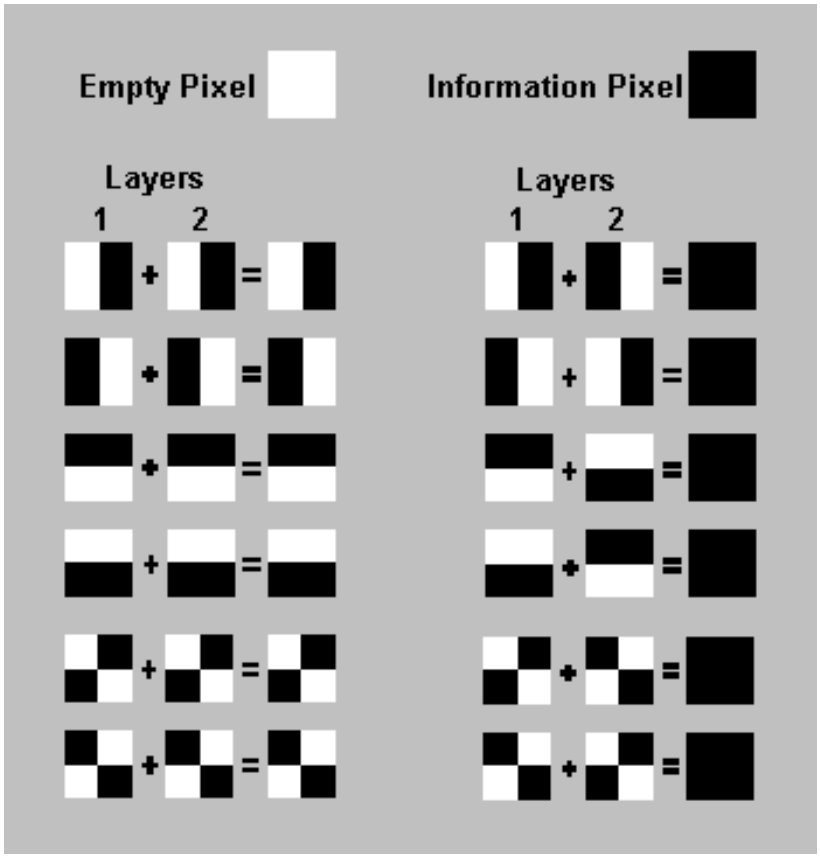
Miguel Huerta
CSC 451 Computer Networks
Professor Kowalski
November 17, 2012

Visual Cryptography

When you hear or think of encryption, the first thing that might pop up into your mind is some form of hiding information. Well in reality that is exactly what encryption is, encryption is defined as the process of encoding information in such a way that hackers or eavesdroppers cannot read it only the persons meant to read the message can. Within this paper we are going to introduce a different kind of encryption known as visual cryptography. If Visual Cryptography is new to you, no worries by the time you finish reading this paper you will have a better understanding of what Visual Cryptography or VC for short is, and also the different types of VC that have been created. This paper is going to introduce the first type of Visual Cryptography and explain how it works. As you continue reading the paper we will introduce a different type of Visual Cryptography and explain the advantages that it may have compared to the traditional VC.

Visual Cryptography or VC for short is defined as an encryption technique to hide information in images in such a way that it can be decrypted by the human vision if the correct key is used. The whole main goal of Visual Cryptography is to be able to decrypt the secret message without using any computation at all. Instead we decrypt the message by using our own human vision. Naor and Shamir first proposed visual Cryptography in the 1994 at Eurocrypt. Naor and Shamir are computer scientist, who study various fields in computer science but they mainly study the foundations of cryptography. Their proposed technique is quite interesting. Firstly the secret, which is being hidden, is divided into two separate shares or layers. One layer contains random pixels and the other layer contains the secret information. By looking at the two shares it is impossible to retrieve the hidden information from them. The reason for no information being able to be retrieved by just containing one of the shares is because the shares are just made up of random noise or dots everywhere. Depending on how each pixel is divided, they can

either be divided into 2 or 4 or any way your want. So if the pixel is divided into 4 parts for each pixel. That means that there is always going to be the same amount of white and black pixels. So dividing the pixel into 4 smaller blocks there is going to be 2 white and 2 black blocks.



This image shows that if the pixel is divided into 4 blocks there are 6 possible states that the pixel can have. Continuing on with the Visual Cryptography description, once the pixels are divided depending on the technique chosen then the layers are created with the information and random data. Once these layers are made they are printed into transparent sheets so that the secret can be retrieved. Next, we stack the transparent layers on top of each other. By stacking the layers on top of each other it is an OR operation, so once the shares are stacked the random dots are set up in such a way that the secret is retrieved. For example if a pixel on layer one is identical to a pixel being stacked from layer 2 then this is going to retrieve a pixel that is half white half black or also known as grey or empty meaning no information is stored. On the other hand if the pixels are opposite on the layers then the retrieved pixel will be black or also known as an information pixel. This is how the secret

is retrieved. This form of encryption is known as a one-time pad system, which is how Visual Cryptography is so secure.

Now that we described the first proposed Visual Cryptography scheme we can now further extend our discussion and move to other methods. Next we will examine the same type of VC scheme but instead of this being a (2,2) Visual Cryptography, we will now talk about a (k, n) VC. This type of Visual Cryptography is exactly identical to the original VC the only difference is that in this scheme the users set up the amount of shares that are generated and also the number of shares needed in order to retrieve the secret. As long as the user have k shares present they can retrieve the secret otherwise no information will be shown in regards to the secret. Lets say that we have a secret and we want to use a (4,5) Visual Cryptography. For us in order to retrieve the secret we need four people present with their shares, or else no information is leaked. No mater how much computational power is available no information can be recovered from a single share. By going to http://www.cl.cam.ac.uk/~fms27/vck/ you can download software that will allow you to experiment with VC and make your own secrets and retrieve them.

So far we have talked about traditional Visual Cryptography, which uses schemes that employ pixel expansion. Pixel expansion is what we did to each pixel, when we either divided each pixel into either 2, or 4 smaller blocks. A disadvantage to this is that depending on the pixel expansion the generated shares are going to be bigger than the original image. In this part of the essay we will describe a form of VC that sets out to minimize this pixel expansion so that not only the shares generated do not get bigger but also try to keep good quality in the recovered secret. Ryo Ito, Hidenoir Kuwakado and Hatsukazu Tanaka proposed this scheme. Their scheme removes the need for pixel expansion. The scheme still uses the traditional (k, n) scheme but the difference is that they include another variable m, which is equal to one. This variable m is the number of subpixels in the shared pixel. Another difference is that a Boolean vector is used for the structure of the scheme. In this vector vi represents the color of the pixel in the i-th shared image. Since so far we are only not dealing with any color images, vi = 1 for a pixel that is black, and vi = 0 for a pixel that is white. If we generate our n x m matrices C0, C1, we then created Boolean matrices S0 and S1 that are chosen at random from C0, C1. The matrices S0 and S1 are going to be of size n X m. We then need to pick a column in S0 in

order to share a white pixel and we pick another column from S1 to share a black pixel. For example if we generate a vector V = [0,1,0], from one of the columns in S1. Then v1 is white in the first shared image, v2 is black in the second shared image, and v3 is white for the last shared image. To retrieve that secret that is hidden within the shares we use the same OR operation applied to the pixels that are in vector V. Because this proposed scheme does not use pixel expansion the variable m is always equal to one and n is based type of (k, n) scheme that is to be used. For example is we decide to use a (3,4) scheme, we would declare n = 4. As with any Visual Cryptography scheme the most important part is always the contrast of the recovered secret image. If the contrast is low the secret is harder to recover. For this scheme the contrast is obtained using the following equations |po – p1 |. All this states is the difference in po and p1, where po and p1 are the probabilities of when a black pixel is generated from a black and white pixel in the secret image. Next we will talk about the security for this scheme. If was discovered that the security for this scheme is as secure as the traditional VC scheme.

Continuing out description of different VC schemes, we now switch directions and decide to talk about the robustness in Visual Cryptography. So far we have only covered images, which are black and white. So it is fair to state that up to this point all of our Visual Cryptography Schemes only use black and white pixels. As Jonathon Weir states, “these black pixels and white pixels are very resilient due to the fact that white pixels will always be white and black pixels will always be black” (18). What this means is that if for any reason the image is altered in any way there is not going to be any change to the pixels. Now we extend this discussion into describing how the binary images we have been using for our schemes can withstand attacks that are commonly used on images. These attacks can come in the form of resizing an image, scaling the image, etc. After the attacker finishes their attacks the pixels in the image still remain the same as they were before the attacks. Since the pictures we have been using so far are in black and white, the values of the pixels don’t change even after the attack. That is why using binary images is a good choice when protecting data is one of the goals, as it is with Visual Cryptography. An example that Weis gives in his book is he takes the traditional VC scheme and he down sampled a number of times. Once the downsizing has been finish no information in any of the shares have been leaked out even after the

downsizing. So overall traditional cryptography using binary images if truly secure because they will not leak out any information when an attacker tries different image attacks on the shares.

Next we will touch a bit more on the security within VC, and finish it off talking about the complexity within VC. In order for the security of a cryptographic scheme to be seriously considered secure it must be completely random. Randomness is a big part when the shares are created. The very first scheme we describe proposed by Naor and Shamir, was completely based on randomness when the set of pixel patterns where chosen. This form of randomness when it comes to choosing pixel patterns all the way up to creating the shares that are to be distributed is the core security feature within Visual Cryptography. What does this mean you might be asking? Well, this means that if you try to use any analysis on the given share no information will be given in regards to the secret that is hidden within the shares. Now we will touch real quickly on the complexity that is found within VC. All of the VC schemes that we have mentioned thus far except size invariant, all the resulting share sizes grow larger than the actual secret image. Weir states, "Typically as the contrast improves, the share size also increases quite dramatically" (21). By the share sizes getting bigger this increases the time that is required to process the image, which as a result increases the complexity of the scheme. If the share gets overwhelming huge they become unmanageable. Up to this point we have only hidden a single secret at a time, but what if we try to hide multiple secrets within the same shares? Well the same think occurs, the shares become too large to manage and difficult.

We will now further our description of Visual Cryptography and present a new type, which is called Extended Visual Cryptography. Extended VC is described taking the idea of VC further by making shares that can be meaningful to anyone that views them. What we mean by meaningful is that instead of just having the random noise on the shares they can include another image on the shares that are present at first, but once the shares are superimposed the secret is revealed. The way that EVC works is that the first n shares that we are going to generate need to be images. These images can include anything such as boats, a car, or anything that can be meaningful. Usually with this specific scheme the secret message is the last think to be dealt. In order for us to do this

we need a technique that has to take into account the pixel colors that are in the secret image so that when we stack out shares the individual images that are present on the shares disappear that the secret is shown. Weir states, “It is assumed that no information is known on the pixel values of the original images that is being hidden. The only thing that is known is that the pixel can be black or white” (32). In order for this Extended Visual Cryptography scheme to work correctly three conditions must be met. First. The images that are present on the shares must reveal the secret image once they are superimposed. Second, if the shares are inspected they should not reveal any information regarding the secret image that is hidden. Finally, the image that is within the shares should not be altered in any way.

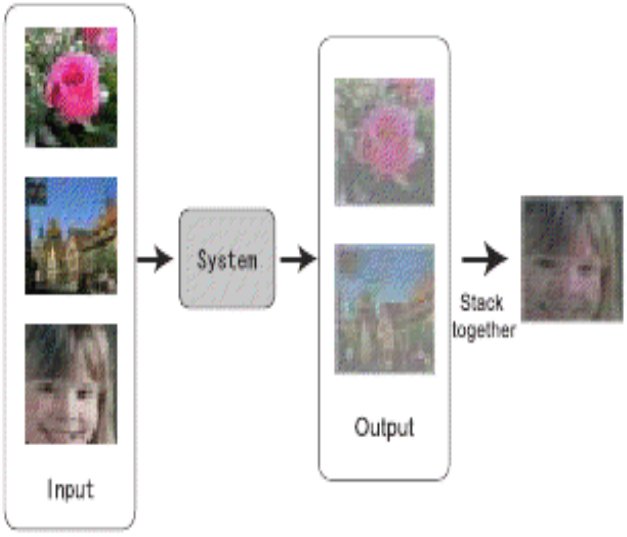


Figure 1: The basic idea of the proposed system.

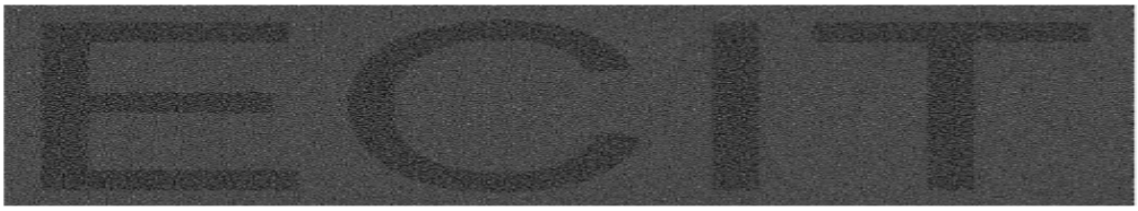
Next we will describe another scheme that is also part of the Extended Visual Cryptography family. The scheme we will examine next is called Halftone VC. This scheme expands on the traditional VC, but also takes extended visual cryptography a step further. This half toning technique can be applied to both color and grayscale images. Weir defines half toning as, “Half toning simulates a continuous tone through the use of dots, carrying either in size or in spacing” (36). Zhou et al. set out to improve these techniques by proposing a type of scheme in which the images carry important visual

information. We have noted that the traditional VS schemes all produce shares in which random patterns are printed on the shares and no true meaning of these patterns are revealed until they are superimposed. With half toning it attempts to take away the suspicion that any form of data encryption has taken place. The shares can carry a piece of information such as an image that is removed when the shares are superimposed.

Our next scheme we will describe is called Dot-Size Variant VC; this specific scheme is a bit interesting. Dot-size variant is defined as "instead of having single black and white dots which make up a VC share, we use a cluster of smaller dots to represent these black and white pixels (Weir 39). The process for this scheme is, we generate two extended VC shares, these shares look normal when viewed, just as any random visual cryptography share would. But in reality these shares are generated with dot size variant in mind. What is so impressive about this specific scheme you might be wondering. Well once the shares are printed out on to their transparencies if anyone tries to take a picture, photocopy them or scan them the smaller dots that were inserted into the shares are altered and the share becomes useless. This is an added form of security is good for when anyone tries to copy the share. This is also good because accurate copies cannot be created at all, and if the shares are copied the information is altered and the shares are completely different from the original. We now move into the direction of explaining how the shares or actually how the scheme works. We will first start the description by stating that in this specific scheme each of the layers that are going to be used to retrieve the hidden secret have dot size variant patterns applied to each. This is done to protect against copying like it was stated earlier. The shares look identical to shares that would be generated using traditional VC but when they are copied the dots are altered and changes the look of the shares. This VC scheme is broken down into two parts. The first part is were we generate the random traditional VC shares, also in this stage we choose the message or image that we want to show when the shares are either copied of scanned. Next we take the shares that we just generated and we use an Extended form of Visual Cryptography on them. We also apply the extended form to the text or image that was chosen if any of the share are too be copied. Next we take the resultant shares and apply the dot size variant scheme to them. Weir describes two types of dot size variant schemes, "the first scheme is a densely populated pixel scheme, and the other is a

sparsely populated scheme" (43). Both of these schemes generate the same output; they work the same way depending which one you choose to use. When we are applying the dot size variant we must first expand each pixel into a 5 x 5 block. This is called a "zoom factor." Another thing that occurs within this zooming is that different block styles are chosen that have specific patterns to represent a black or white pixel. "The patterns are known as Filter1 and Filter2" (Weir 44). If the shares are large the black pixels are the ones that are replaces, the white pixels are scaled to the new size and made up from white pixels. So from the sentence before we can define Densely populated pixel set as having only solid black pixels, and sparsely populated pixel set would be the opposite and would only contain solid white pixels with a few black pixels. Lets continue the description of the proposed scheme, so once the filter has been chosen we must modify the extended shares with the filter. After this step the shares have been completely finished and include dot size variant pixel sets to help show if any copies have been made.

Figure 2.10: The resultant share using densely positioned pixels. © Weir & Yan 2009



The image that is right on top is a completed share that has dot size variant scheme within it. This specific share has densely populated pixel set. Once a picture was taken of the share the bottom image is the outcome. As you can see the pixels have altered and have shown the text that was chosen in case the share was to be copied.

Dot Size Variant Visual Cryptography scheme is mainly designed for printed images. We continually kept emphasizing the problem of having any of the shares copied. For that reason this scheme should be primarily used within applications that are meant for printing purposes, so they could detect copied shares. A good printer is needed to achieve superior results. By using a good enough printer preferably a printer that can

print at 1200 DPI (dots-per-inch) to prevent copies from being made.

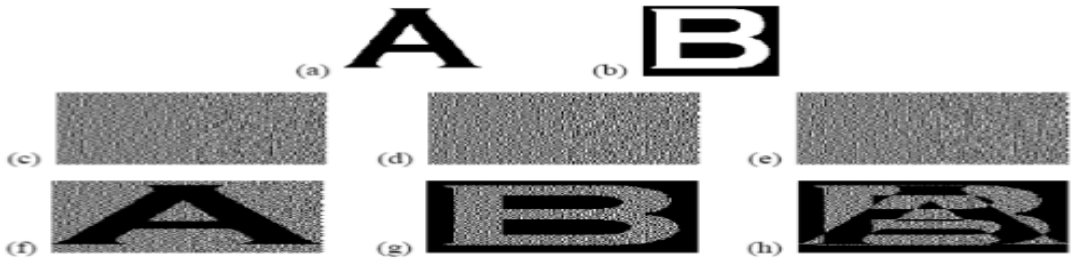
We now will further our description of Visual Cryptography, and think about the question, what if we want to share more than one secret. Up until now we have only dealt with sharing one secret at a time. But just because we have only described these types of schemes doesn't mean sharing multiple secrets is not possible. Dynamic Visual Cryptography is exactly the solution we are searching for if we would like to share more than one secret at a time. Weir states, "The core idea behind dynamic visual cryptography is increasing the overall capacity of a visual cryptography scheme. This means that using a set of two or more shares, we can potentially hide two or more secrets." So as we posed the question at the beginning of the paragraph, this new type of VC would be perfect for the case of hiding multiple secrets within the shares we generate.

Wu and Chen originally studied a scheme that would hide multiple secrets within two shares. Their examination concealed two secrets within two sets of shares. We will give the two shares the name F1 and F2. The way that they retrieve the first secret was that they superimposed the two shares. This is how we have been retrieving secrets thus far. But now u might be wondering how are we going to get the other secret that is hidden within the shares. The answer is, the second secret becomes available when the first share F1 is "rotated anti-clockwise 90" and then superimposed on the second share. The drawback to this first attempt of sharing two secrets within 2 shares is that because of the angles that are required to reveal the second secret we cannot share more than two secrets at a single time. But more extended forms of sharing many secrets began to be studied. We will not describe all of these in detail only state some characteristics about them. There was a scheme that was developed that used the same stacking of the transparencies to retrieve the secret but the only difference was that in this scheme the improvements made are the numbers of sub-pixels are reduced. Another scheme was also developed that also extended the multiple secret sharing scheme, in this scheme circular shares are generated so that the angles that are allowed do not have any limitations on them. The multiple secrets are retrieved by stacking the two shares on top of each other and rotating the second share clockwise (the angle can be between 0 and 360 degrees). The final scheme we will describe quickly is a scheme that hides 2 or more secrets into circular shares. This scheme can also be extended to work with grayscale images using half

toning techniques. However, there is a drawback to this specific scheme. As you might have noticed all the problems that we have defined all are about the problem of pixel expansion, and this scheme is no exception to this problem. The expansion for this scheme is twice the number of secrets that are to be hidden, so for our circle shares they will increase largely when many secrets are hidden. Another problem that arises when we use circle shares is that because we many secrets hidden in the shares, knowing how many secrets are actually hidden are a bit hard. Also the angles that might be required to show a specific secret might be small and easy to miss.

We will not describe a form of multiple sharing, named disjoint visual cryptography. The main idea behind disjoint visual cryptography is to share separate images while only using one master key. This uses the same stacking technique as all the other schemes to reveal the secret that is hidden within the share. The shares can be arranged in three different ways; they can either be arranged horizontally, vertical, or diagonally. So, for example is the shares happen to be arranged vertically, then in order to reveal the secret that is hidden within the share you must shirt the master key to the vertical direction. By now you might have determined how this visual cryptography scheme differs from the traditional visual cryptography schemes. If you haven't figured it out we will let you know. The way that this scheme differs is that we usually generate two separate shares and superimpose those two shares to retrieve a secret. However, with this scheme we only need to one share along with a master key that works for all the shares that you might generate. Although this seems like it is easier to extract secrets from the shares some problems do arise. The main risk is the leaking of information. An image that was shown in a book shows two secrets that are to be shares. The images are an image of an A and a B. Once the shares are generated we have three separate shares. We have the share for each of the images, and the master key share. We show superimposing the master key on either the share to reveal the A or the share to reveal the B the secret. However, if we stack the shares with both the images we leak information. Once the two shares are stacked together partial images revealing both secrets are shown.

The image below demonstrates what we have just described.



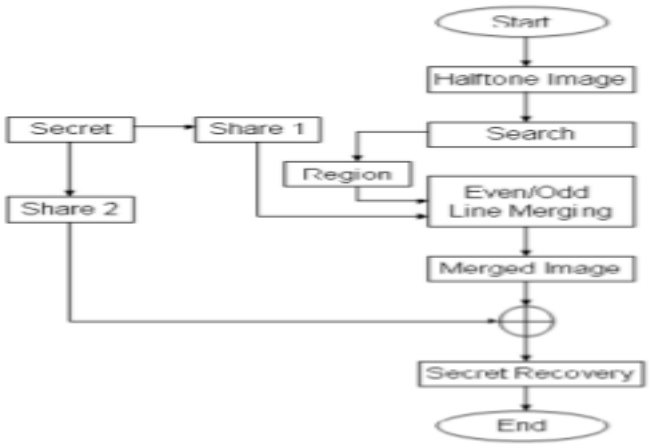
Next we examine another form of multiple sharing, named Joint visual cryptography. You might be wondering why this sounds somewhat similar to the scheme we just described. This scheme uses the same procedures as the disjoint scheme; shares are generated for the secrets and are retrieved using a master key. However, there are three different techniques to accomplish this. The first technique is known as contrast based joint combination of shares. This is built on the idea that multiple shares can be created along with one key, but overlapping the shares will give one final share and by staking the key on that final share the first secret is shown. Then either by shifting the key horizontal or vertical the other secrets are shown. This technique has a disadvantage; it cannot share a secret image that has been made up of only black pixels. The reason for this is that if we choose to retrieve a secret that is only made up of black pixels then the other secrets that are suppose to be retrieve by shifting the key will not show because there is not going to be any room on the share to insert the black pixels that are required to display that secret.

The second technique is named even-odd joint combination of shares. The way that shit technique works is that we are given two secrets, which are the same size. We first generate two separate shares one for each secret, and also we generate a random master key. Next we merge the two shares together, but there is a certain way that we fill the new share. We take the first share and fill the new share's even rows; we then take the second share and fill the combined share's odd rows with the second share. After the combining of the shares is complete the new share is going to be twice the size of the secrets. So we must next adjust the master key and generate a new key with this new size. The difference between disjoint and even-odd shares is that the key that we generate is the same size as the share that is hiding the two secrets. What this does is increases the

security of the technique because it doesn't give away no information regarding the amount of secrets that are hidden.

The final part of the joint techniques is multiple joint combinations of shares. This final technique shares multiple secrets using only one share and a master key. Moving the master key around shows the secrets, until all the secrets are all revealed. The way this technique works is that one pixel from each secret is chosen and expanded into a 2 x 2 array. Once all the arrays have been generated, meaning once an array is created for each of the shares then those arrays are moved to a larger image. For example if we decide to hide three secrets, then the final share which contains all the secrets in it is going to be three times the size of the images. The way that the final share is created is that for each pixel that we expand that array is placed into its group of four pixels in the final share. This is done for all the pixels in secret one, and then for all other secrets to be hidden. For the key creation the same process is used as with the pixels in each secret. However, for the key the ordering is reversed. If we did not reverse the ordering of the process then all of the secrets would be revealed at the same time. By reversing the process then all we have to do is shift the key by four pixels (either two pixels up or down, or side to side) to show each hidden secret. One problem with this technique is that the contrast drops when a lot of secrets are hidden. Contrast is very important when we are dealing with visual cryptography. If the contrast is not that good then when we recover the secret we will not be able to make out what the secret is. On the other hand if the contrast is good, when the secret is retrieve is it legible and easy to read which makes it easier for the people reading it.

We next will describe another form of sharing multiple secrets but in this case we embed a share of visual cryptography in a halftone image. With this scheme the secret shares are going to be embedded into the cover images. By doing this we remove any suspicion that anybody might have about any form of encryption taken place. When we are done generating our shares they will look very appealing to the people looking at them, but the most important part is that they are not going to know that in reality there is a secret hidden in the share. There is an image pasted below from Weir book titled Visual Cryptography and its Application. The image is the flowchart for the scheme that we are describing for embedding a secret into a cover image.



We will not describe the process. We start by either selecting or searching for an image so that we can hide the shares within the image. Once we selected the image to use we must now convert the image into a halftone version. For this specific scheme the way the halftone image is created is by using a method known as dispersed-dot dithering. "Dispersed dots were chosen because they usually have a square shape, this corresponds to the square nature of the VC shares allowing a share to be inserted into a halftone image with minimal changes to the overall image" (Weir 64). Once the image has been converted then we move on to the next part in the scheme. We not must generate our shares that we are going to embed into the image. Next we must search for a part within the image that we are using where we can insert the share. When a part in the image is found we then embed the share into the image using the technique of even and odd lines. We can either insert the share using the even scan lines and the images odd scan lines or the other way. Whichever way we choose we will get the same outcome in the end. Once the merging is done the image includes the secret that we must retrieve by overlapping the other share over the region where the secret is hidden. We can also hide multiple secrets if we choose; we are not limited to only hiding a secret at a time. The way that we hide many secrets is the same way we hide a single secret, we search for regions in the image that the secret can be hidden at. Once the regions are found then we can hide the secrets. We must use the key in order to retrieve the secrets, by going to the region where the secret is hidden and retrieving it.

Keeping the discussion going we will not move to yet another type of visual cryptography. This time we will talk about Color Visual Cryptography. Weir states, "

One of the most potentially useful types of visual cryptography scheme is color visual cryptography" (69). He goes on and states the reason for his statement. Cutting down the statement to just a precise and straightforward explanation is that color is what people are used to seeing more now. With this type of visual cryptography however, we do run into some problems. Up until now all the decryption that we have been doing have not involved a computer at all. They way that we have been getting our secrets is by stacking the shares on top of each other. For color VC there are some schemes that require a computer to retrieve the secret, but there are also some schemes that keep the traditional way to decrypting. We now go on to describing one of the many color schemes that are available. For this scheme each pixel of the color image that is going to be hidden is expanded into a 2 x 2 block of sub pixels. Once all of these blocks have been completed we now must fill all of the blocks with red, green, blue, and white colors respectively. There are a total of 24 different combinations for two pixels. In order for us to encrypt a pixel into the color image we must round the pixel's color value. Next we must select a random order for the pixel blocks for the first share that we are going to generate. We must also select the ordering that we wish to have on the second share, but we must pick a correct ordering because the combination that we are going to get back must match the color we need. One advantage of using this scheme is that we can represent up to 24 colors and also the resolution is reduced. But we also have a disadvantage and it is that once the 24 colors are fixed so are the sub-pixel colors. There is more color schemes that are describes within the section for color visual cryptography. We will not describe any for the reason that they state certain things that they do not fully explain. Instead we will move to sharing images using random masks.

Right now the most robust way of hiding a secret within an image is to use some form of visual cryptography. The most basic form of VC can achieve this and no computer is needed in order to retrieve the secret. Watermarking is a way of hiding a watermark in a document and this type of hiding does not need any key to retrieve it. This is the exact idea that was taken for hiding secrets. For the next technique description this is the idea that was taken. A watermark was taken and a mask was applied to it to hide it more securely. When speaking about mask for this particular case the mask is a mechanism that is found within Internet Explorer. More specific a function called select

all. A quick explanation on this is, the software shows an image to viewers that have been hidden in the image they are viewing. By the user selecting the function the hidden image is shown, if for any reason the viewer cancels the operation the viewer will be shown the original image not the hidden image. We will expand on this explanation more in the next paragraph.

The main purpose of this part of the paper is to demonstrate how to hide an image using Internet Explorer. Within Internet Explorer the select all function has a fixed mask, this mask that is found within IE is what gives the transparent effect. If you would like to see this transparent mask simply open a white image and then use select all function. The mask will look blue and white. Since we cover the essential description of how the IE select all function works we will not touch on another scheme that embeds a share in a color image. Shares can indeed be embedded into a color image. The merging technique to accomplish this is different than the scheme for halftone. The share that is generated is random, and the image itself looks as though it has some random noise on it. This scheme works just like all the other schemes. We take the share that is not embedded into the image and impose that share in order to get the secret out. With this scheme the image remains almost exactly the same and not that many changes were made to it after the embedding. The changes that were made are hard to detect and in many cases the changes cannot be found by anyone who searches for them.

Most of the schemes that we have been describing so far just expand the idea of visual cryptography. We either describe a technique to do something new, or we use the traditional visual cryptography scheme and expand on it to accomplish another task. We will not move in a semi different direction and worry now about the overall quality of the recovered secret. Most of the recovered secrets have been ok when is comes to overall quality. We cannot say that they were terrible in quality or else we would not have taken the time to describe the scheme if the secret could not be seen at all. This form of visual cryptography that deals with the quality of the recovered secret is called progressive VC. One of the main differences between these sections is that most of the schemes that we are going to describe require computers in order to perfectly reconstruct the secret.

Throughout many different parts of this paper we have touched on the subject of contrast when we were talking about the recovered secret. Most of the time the loss in

contrast was not huge enough for us to have many complains about. However, what if perfect recovery is something that you might be in search of. Rarely is quality not something that is not asked for. Most of the time we wish to have perfect or somewhat perfect recovery of our secret. Halftone based and color visual cryptography is something that we have discussed within this paper. In this part of the paper we will talk about them some more and see how the overall quality of the secret is compared to the original. Half toning is used most often within the printing industry for printing applications; it has been proved very effective. For our case however, half toning is just used to convert a grayscale image into a binary image. Once we have our binary image we can apply the visual cryptography scheme so that we can hide our secret. But in the end using this type of technique the loss of quality to the recovered secret is unavoidable. We now move into color images, for color images we have two different ways of applying half toning. One of the methods is to split the color image into three channels, cyan, magenta and yellow. We then take each channel and treat is as its own grayscale image and we apply the half toning and visual cryptography to each. Once the channel shares have been generated in order for us to create the color shares we combine the channel shares. With the other method we do not need to break the image into channels, we just apply the color half toning directly to the image. After the halftone technique has been applied we then split the image into channels in order for us to use a visual cryptography to the channels. If you did not notice, these two methods basically do the same thing in the end it is just a minor step in between the process that is switched. So in the end no matter which method you choose to use you will get the same results. However, these two methods in the end when the recovery is generated still are not perfectly recovered.

As we have noted no matter which way we use for our visual cryptography technique it seems the in the end we still get a loss of overall quality. Within this section we are going to state a new method of converting grayscale images and color images into binary images without loosing any information. We would also talk about a new way of doing our recovery in which the secret is fully and perfectly recovered. The schemes that we will talk about still hold the advantages of traditional visual cryptograph, such as simplicity; the decoding process and they remain secure. The one difference is that we will not present a new method of recovering the secret. Up until now we have been

recovering our secrets by using the OR operation (stacking the share on top of each other). But now we will present a new way of recovering the secret XOR, by using this way of recovery we are guaranteed full quality of secret in the end. Before when we would use the OR operation the contrast had to be lowered because of the operation we used. Instead using XOR this changes the whole overall quality in the end, and we retrieve our secret with no loss of contrast. There is a disadvantage to using XOR; the disadvantage is that a computer is needed. The stacking procedure that we were doing before can only demonstrate the OR operation, if we wish to demonstrate the XOR we must use a computer.

We went off course for a few paragraphs but now we will get back on track and describe some more visual cryptography schemes. We just told you that using the XOR operation a secret can be perfectly recovered, but a computer is needed to do this operation which defeats the whole simplicity for decryption that VC constantly states. In this part of the paper we will talk about image hatching for visual cryptography. The hatching of images or engraving styles are still used today. When dealing with currency, engraving and hatching is widely used because it helps to find counterfeit currency. Weir defines image hatching as, "in general is a series of similar strikes which use various lengths, angles, mutual space, and other properties of lines to represent parts of an image" (99). By applying these different strokes it gives the image more shape along with depth. We will discuss a scheme that uses image hatching along with visual cryptography; this scheme differs from image hatching techniques in the way in which the hatched image is generated. They take the existing image hatching techniques and extend them into the domain of visual cryptography; the scheme we will describe can be used on any grayscale image.

The scheme for hatching an image has eight different textures, which are used for different purposes such as shading effects. Eight different textures where chosen because that is how many thresholds were taken of the original image. We apply a different texture to each threshold that we have. The steps are as follows

"Step 1 load original image and obtain dimension, width w and height h

Step 2 create 8 new blank images, w x h: n1, …n8

Step 3 for s1, draw vertical and horizontal lines on n1 with a space of 2 pixels between each line

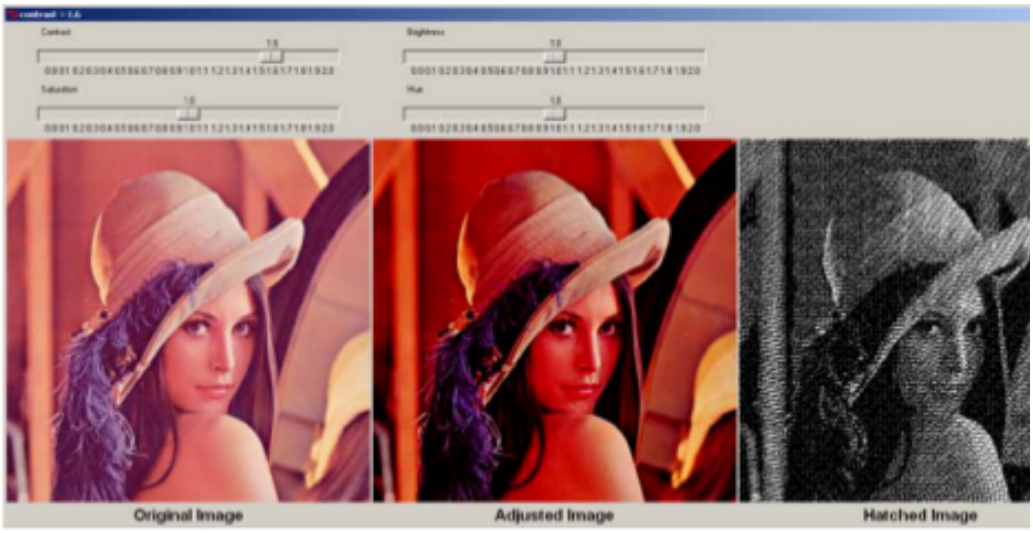
Step 4 for s2, draw vertical black lines on n2 with a space of 3 pixels between each line

Step 5 for s3, draw horizontal black lines on n3 with a space of 5 pixels between each line

Step 6 for s4, draw positive diagonal black lines on n4 with a space of 5 pixels between each line

Step 7 for s4, draw negative diagonal black lines on n5 with a space of 5 pixels between each line

Step 8 for s6 → s8, draw elliptical black lines on n6 → n8 with a space of 6 pixels between each line" (Weir 102). This is part of the algorithm that was used for the creation of the texture code. Using these textures that we just mentioned are applied to the eight different thresholds of the image. The way that the textures are applied is by inverting them and then superimposing them. We do this for all of the thresholds and textures. The order must match so s1 has to go with threshold 1 and so on. All of these steps that we have talked about is part of the software that is being discussed, which takes an image, and give you as an output a hatched image. The image below is the screenshot for the software.



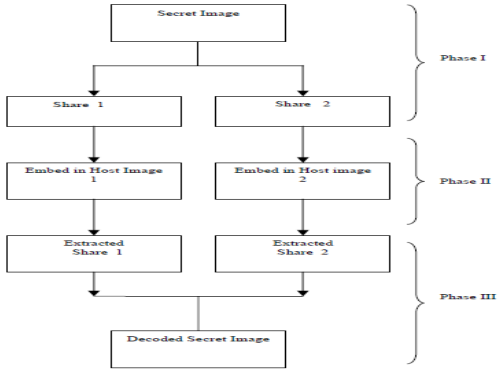
You might be wondering how this has to do with visual cryptography, this is the first step that we must take in order to retrieve our hatched image so that then we can

apply a VC scheme. So once we generate our hatched image, we must also now choose a secret that we which to hide. For this scheme we will use the size invariant scheme that we discussed earlier, for the reason that the shares that are generated are not larger than the images. In order for use to properly embed the first share into the hatched image we must first alter it. We alter the image by reducing the pixel density, however this has a tradeoff between security and being able to hide the share. In the end the overall contrast of the recovered secret is going to be lower. With the second share we must decide whether or not we would like a hatched mask. If we choose to the mask is created, if not the share is left as is. We retrieve the secret as we have been doing by superimposing.

We have discussed many different visual cryptography schemes but I believe we have now reached the end of this paper. The final topic we will discuss is how all of these VC schemes we have been talking about can be applied to practical use in different application. We will mostly be talking about watermarking and how we can use visual cryptography along with watermarking to hide secrets. In the context of this discussion we will define watermarking as the process of hiding information in such a way that it is difficult to remove. The scheme we will discuss takes a cover image along with generated shares and uses watermarking techniques to hide the information within the image. We start off by first generating our shares using the traditional visual cryptography (2,2) scheme. The way we will divide each pixel is dividing it into 4 sub-pixels. The share can either be vertical, horizontal, or diagonal. We retrieve the secret by stacking the shares on top of each other. The next phase is hiding the shares using digital watermarking. We need to pick two images in which the share can be embedded into. We will use discrete cosine transformation in order to convert the image into frequency domain. The reason that DCT is used for watermarking is that it divides the image into distinct frequencies, and this allows us to embed the watermark into the image easier. The algorithm that Dr. Khemchandani used will be stated below, it consists of 19 different steps. The software that was use was MATLAB 7.0. The steps are, "**Step 1:** Set minimum coefficient difference. **Step 2:** Set the size of the block in cover image to be used for each bit in watermark. **Step 3:** Read in cover object.

Step 4: Determine size of cover image. **Step 5:** Determine maximum message size based on cover object and block size. **Step 6:** Read in the message image. **Step 7:** Reshape the

message to a vector. **Step 8:** Check that the message is not too large for cover. **Step 9:** Pad the message out to the maximum message size with ones. **Step 10:** Process the image in blocks. **Step 11:** Transform block using DCT. **Step 12:** If message bit is black then value of frequency coefficient (5, 2) > (4, 3). **Step 13:** End if **Step 14:** If message bit is white then value of frequency coefficient **(**5, 2) < (4, 3). **Step 15:** End if **Step 16:** Adjust the two values such that their difference >=k. **Step 17:** Transform block back into spatial domain. **Step 18:** Move on to next block, at the end of row move to next row. **Step 19:** Exit"(Khemchandani 4). Once you have accomplished all of these steps the share has now successfully been embedded into an image using watermarking. The next phase is to decrypt the watermarked shares to get back the original shares that we generated. Once we have the shares retrieved we cannot decode the message. The steps for the decryption process are as follows, "**Step 1**: Set the size of the block in cover to be used for each bit in watermark. **Step 2:** Read in the watermarked object. **Step 3:** Determine size of watermarked image. **Step 4:** Determine max message size based on cover object and block size. **Step 5:** Process the image in blocks. **Step 6:** Transform block using DCT. **Step 7:** If dct_block (5, 2) > dct_block (4, 3) then message bit is 0, otherwise message bit is 1. **Step 8:** End if **Step 9:** Move on to next block, at the end of row move to next row. **Step 10:** Reshape the embedded message. **Step 11:** Exit." This concludes out explanation of watermarking using visual cryptography. This recent example that we just talked about requires MATLAB 7.0 to simulate the VC scheme.



This, conclude our journey through the world of visual cryptography. We started off by first giving a little information about who were the first two people that proposed this form of encryption. We quickly gave a bit of information about where and when this form of encryption was proposed. We then moved along to the main goal of the paper,

which was to define what visual cryptography is and how it is used. We included a picture of different ways that a pixel can be expanded in order for the traditional form of visual cryptography to be accomplished. Once we covered the traditional (2,2) visual cryptography scheme we moved along and extended upon our discussion. Throughout the whole paper we described different forms of visual cryptography that have been proposed and also stated how they might differ from one another. We also described how most of the schemes that we covered, the secret is recovered the same way by stacking or superimposing the shares on top of each other. As we got closer to the end of the paper that is where some of the schemes got a bit more complex and a computer was needed to retrieve the secret. Instead of stacking the shares on top of each other to get the secret back we have to use the XOR operation, which we could not simulate by stacking the shares. However, the XOR proved to be a good idea because the secret that we recovered at the end was a perfect recovery and no contrast was lost like the OR operation. This is what was discussed in the paper, we didn't get into the mathematics of some more in depth talks about the schemes because the whole point of the paper was to describe the different schemes and give a quick description on them. Hoped you enjoyed learning about something that for me was particularly interesting to research and learn for the first time myself.

Works Cited

Weir, Jonathan. Visual Cryptography and its Application. Ventus Publishing ApS, 2012.

Hsu, Ching-Sheng. Digital Watermarking Scheme with Visual Cryptography. Proceedings of the International MultiConference of Engineers and Computer Scientists 2008 Vol I. IMECS 2008, 19-21 March, 2008, Hong Kong.

Khemchandani, Vineeta. A Visual Cryptographic Technique to Secure Image Shares. International Journal of Engineering Research and Applications. Vol. 2, Issue 1, Jan-Feb 2012.

Ito, Ryo. Image Size Invariant Visual Cryptography. Special Section on Information Theory and its Applications. October 1999.