BRCORP
TECHNOLOGY | INNOVATION | INDUSTRY

5th World Conference on
Applied Sciences, Engineering & Technology
02-04 June 2016, HCMUT, Vietnam

A Novel Image Security Measure with Conventional in-Browser Methods

DITIPRIYA SINHA[1], RITUPARNA CHAKI[2]
[1]*Department of Computer Science and Technology, National Institute of Technology Patna, Patna, India*
[2]*Department of AKCSIT, University of Calcutta, Kolkata, India*
Email: ditipriyasinha.cse@nitp.ac.in author, rchaki@ieee.org co-authors

Abstract: The extensive use of e commerce creates new ways for both image and brands to be attacked. Attackers use various gimmicks and techniques to find various loopholes in the system. While there has been good ways to identify threats, educating customers and identifying countermeasures. Attacks on customer sensitive information has the adverse effect of decreasing the consumer faith on online transactions, which happens in ecommerce. Prevention is better than cure, good knowledge and understanding of the online threats can be used as a tool to avoid online attacks. We describe an image rendering technique for protecting images in the ecommerce sites from being copied by fake vendors. The scheme does not allow any type of modification to be made to the image and generates an alarm whenever any user tries to do so. The advent of internet with ever increasing bandwidth resulting in more speed has led to a sudden upscale surge in the e-commerce sites around the internet. The users have the luxury of shopping for 24x7, and the sellers have the advantage of keeping the sites open always. The improved reach to customers, coupled with low operational costs has made this a popular choice among budding entrepreneurs. It has been seen that online fraud has been one of the biggest set-backs for China's e-commerce. Products with fake brand images have continued to appear in different sites, creating problems for the buyers as well as original sellers. The problem of pirated music and videos has long been known to cause troubles to dealers of originals. At present, another problem is threatening to ruin the e-commerce sites. When new goods are sold in certain e-commerce markets, buyers have trouble distinguishing between genuine goods and fake goods. Fake goods sellers use the same photos, so buyers are unaware of the duplicity. This is a particular problem for many of the e-commerce sites. The traditional methods of image protection include watermarking, tiling, etc. In this paper, a study of existing solutions for image protection has been presented, together with a research gap analysis. Finally, a secure logic has been pro-posed for image protection based on the technique of image rendering.

Keywords: *E-commerce, AJAX, HTML5 Canvas*

Introduction:

The advent of internet with ever increasing bandwidth resulting in more speed has led to a sudden upscale surge in the e-commerce sites around the internet. The users have the luxury of shopping for 24x7, and the sellers have the advantage of keeping the sites open always. The improved reach to customers, coupled with low operational costs has made this a popular choice among budding entrepreneurs. The users also have the options of choosing from a multitude of vendors offering different prices. However, so much facility does come with a price – the multitude of vendors offering different prices for the same product often turn out to be selling fake items. There is no guarantee towards the quality of product-as customers have to select them as per the images made available on the vendor's ecommerce site. Digital images can easily be tampered and it is difficult to tell whether the image under consideration is authentic or has been altered by some unauthentic user after capture by some readily available digital image processor.

It has been seen that online fraud has been one of the biggest set-backs for China's e-commerce Products with fake brand images have continued to appear in different sites, creating problems for the buyers as well as original sellers. The problem of pirated music and videos has long been known to cause troubles to dealers of originals. At present, another problem is threatening to ruin the e-commerce sites. When new goods are sold in certain e-commerce markets, buyers have trouble distinguishing between genuine goods and fake goods. Fake goods sellers use the same photos, so buyers are unaware of the duplicity. This is a particular problem for many of the e-commerce sites. Figure 1 shows two vendors offering Apple.

$10
Apple iPhone 4 4S Stylish Black Case Cover

$55

There is a variety of prices, but it is not easy to determine which vendors are selling genuine products and which are selling fake products. Pricing information and seller reputation do not provide any meaningful information to discriminate between the two. Thus we are interested to know not only about the integrity of the image, but also the originality of the seller.

The traditional methods of image protection include watermarking, tiling, etc. Watermarking involves the embedding of text/graphics in the image. The technique is the most popular of the lot due to ease of implementation. However, reverse watermarking has proved to be an easy tool for restoring the original image by fake vendors.

In this paper, a study of existing solutions for image protection has been presented, together with a research gap analysis. Finally, a secure logic has been proposed for image protection based on the technique of image rendering.

State of the Art Analysis:

In order to extract images from webpages, automated programs are used which can be classified into two major categories.

- The simplest solution is to create a program which parses the HTML page for image links and downloads them. This works only if the image links are present in the HTML code or in CSS/JavaScript files.
- Using packet sniffing/analyzing software to analyze network packets and determine the data as image and then store it.

This section examines and lists security protocols followed commonly used by E-commerce sites to protect images apart from using HTTPS connections.

- No Security - Major e-commerce websites and online shopping portals like Amazon and eBay have no elaborate image security attempts to protect the product images.
- Tiling images – Tiling the images into smaller fragments makes it tedious to save images as each portion needs to be saved and then stitched together in an image manipulation program to retrieve the original image. This is done by Flickr.
- Watermarking - The most common solution is watermarking the images but they can be still saved locally and watermarks can be done away with modern powerful image manipulation programs.
- Image as HTML Table – The image can be divided into a 2D collection of pixels and then each pixel is represented as a HTML table cell with appropriate background color matching that of the pixel.
- Limiting access to Images – User account creation is mandatory in many websites in order to browse detailed information and images about products. This implies that user behavior can be tracked at the server side and processed.

There have been different types of watermarks proposed in the literature, designed for different applications [1], [2]. For example, ownership assertion watermarks have been proposed for insertion in images that are to be made publicly available, so that unauthorized users who claim ownership or re-sell the images without the consent of the original owner can be held accountable. Such watermarks are typically robust [3]–[6] in the sense that the watermark is usually detectable even after the watermarked image has been processed by common image processing algorithms like scaling, cropping and compression. Another need of image authentication arises in, for example, electronic commerce where the seller transmits a digital image to the buyer over the network. In this case the buyer wants to be sure that the received image is indeed genuine. Here not only integrity of an image was verified but also it did not check original ownership. To address these issues, the idea of a trusted digital camera was proposed by Friedman in [7]. Freidman presented logic to address the issues of protecting image transmission by e-commerce sites to the buyer. The authors presented logic to address the issues of protecting image transmission by e-commerce sites to the buyer. A trusted digital camera was used in this scheme, whereby, a standard cryptographic signature is generated for each captured image, which is stored and transmitted along with the image. The digital signature helped to verify whether the image has been tampered, as well as the origin of the image. Yeung and Mintzer [8] proposed using an authentication watermark to protect the integrity of images using a pseudo random sequence and a modified error diffusion method to embed a binary watermark to an image, so that any change in pixel values to the image can be detected. Extensions of this technique to compressed images were proposed by Wu and Liu [19] – Lin and Chang [9] proposed a scheme to insert authentication data in JPEG coefficients so that the authentication watermark has some resilience against JPEG compression. Recently, this scheme has been extended to handle information in the form of video [10]. Here, the logic uses a watermark that allows a user with an appropriate key to verify the integrity and the ownership of an image. This authentication watermark can detect and localize any change to the image, including changes in pixel values or image size.

The security of the proposed techniques in [11] resides in the secrecy of the user key and a public key cryptographic system [12]. The secret key scheme uses a cryptographic hash function such as the MD5 [11]. Its security relies on the computational infeasibility to break the cryptographic hash function. One disadvantage of this scheme is that the secret key will need to be transmitted from the image owner to the user through a secure channel.

Holliman and Memon [15] discovered that the schemes in [13], [14] and many other block-based schemes proposed in the literature, are vulnerable to a "vector quantization" attack. [16] proposed a solution very similar to the public key scheme in [14], except that Coppersmith et al. used overlapping blocks to defeat the "vector quantization" at- tack. In this paper a visible watermark is inserted in the image. Visible watermarking refers to the type of technique where a visible stamp, e.g., a company logo, is inserted to the image [17]. There are generally two problem associated with visible water- marks. First a visible watermark must be difficult to remove. In this regard, Braudaway et al. suggest the insertion of random noise to the watermarked image to increase the difficultly in manually removing the watermark [17]. Second, a visible watermark must be able to withstand the impersonator problem.

1.1 Data Scraping Tool Detection

In order to thwart data scraping software, legitimate web browser detection is a must. Merely detecting browser user agent string is not enough as it can be spoofed using appropriate software. (This can be seen from exiting browsers as well i.e. Chrome can change its user agent string using extensions). Hence more sophisticated and granular methods are required to do this. Tools/Software that scrape data can be categorized into 3 major categories.

- Headless browsers-based Tools – These are full-fledged web browser without a GUI front-end and hence can fetch and render any HTML page with full CSS and JavaScript support. To detect such browsers, the window object's two properties namely **outerHeight** and **outerWidth** needs to be checked. The value of these two properties is 0 if the browser is running in headless mode otherwise not.
- Off-Screen Rendering Tools – PhantomJS in conjunction with Node.js can render any web page and even execute client side script and hence such tools can extract information by executing the JavaScript. To detect such tools, there are few properties of window object which should be checked such as **__phantomjs**, **callPhantom** and **_phantom**. Buffer property of the window can also be checked for Node.js existence.
- HTML-Only Tools – These tools only fetch the HTML page, parse and try to extract embedded URLs. These tools do not have the facility to execute client side scripts such as JavaScript and hence does not pose any threat to the proposed solution.
- Other Tools – There are some other common tools which can be used to scrape data such as Selenium web drivers, RhinoJS JavaScript interpreter and Chromium based automation drivers. These tools can similarly be checked with window object's properties from client side script.

Scope of the work

State of the art analysis shows that most ecommerce websites display product photographs on their websites to let prospective customers have a look at what they are buying. Websites don't have system-level access and run in a sand-boxed environment inside web browsers. This means that any agent viewing the page can copy the images and use it without consent for fraudulent purposes or personal gain.

The above study leads to the following observations:

- Modern powerful image manipulation programs with sophisticated algorithms can easily do away with watermarks and stitching together tiled images is trivial.

The HTML Table method can have two possible implementations with flaws which are:

- The raw image data is transferred to the website through AJAX calls and the client side JavaScript creates the necessary table from this data which incurs heavy CPU usage plus the pixel data from the table can be parsed with an appropriately written program to parse the whole HTML page.
- The whole table is created from the server side and then transferred to user side which will incur more bandwidth for the pixel information and table row/column tags and the HTML page can again be parsed to extract the information.

In the next section we describe the details of our image rendering algorithm for colored as well as gray scale images used by the ecommerce sites. The technique is browser dependent. The logic is built into server side and client side modules for image protection.

Proposed Work:

Based on the above observations, our proposed solution aims to thwart any attempts of copying the image at client side by making it too difficult to extract the image. The solution can not completely stop the user from accessing and copying the image which is examined at section. The aim of the proposed solution is to make it increasingly difficult for site scraping tools to extract the image. This is done via JavaScript code – as this have been observed to be a weakness in most site scraping tools which lack a dedicated JavaScript interpreter. However in case such a situation arises, the JavaScript uses browser local storage (HTML5 specific feature) in conjunction with some other code to verify if the request was indeed from a web browser.

The packet sniffing tools can be thwarted with TLS and HTTPS – which encrypts the network traffic between the server and the browser. Major browsers i.e. Google Chrome, Mozilla Firefox and Internet Explorer let users know if they are visiting websites whose Digital Certificate has expired or not and whether the HTTPS connection is safe or not. This means that packet sniffing tools also need to decrypt the data in order to access it. Thus the solution in its present does not try to solve the issue with packet sniffing tools.

Our solution uses HTML for base image rendering, as HTML has multiple elements on which images can be drawn. These are:

- HTML Image Element
- HTML5 Canvas
- Background property of any element i.e. div element.

It is normal for modern web browsers to include the "context menu" (accessed with a right click) which provides "Save Image" option. The proposed solution starts with disabling the context menu within the element. This implies that there is no save option from within the web browser. Here is a comparison of the elements with respect to context menu.

Element Type	HTML Tag	Has Context Menu?
HTML Image Element	<img>	Yes
HTML5 Canvas	<canvas>	Yes (in Chrome)
HTML DIV Element		No (for background image access)

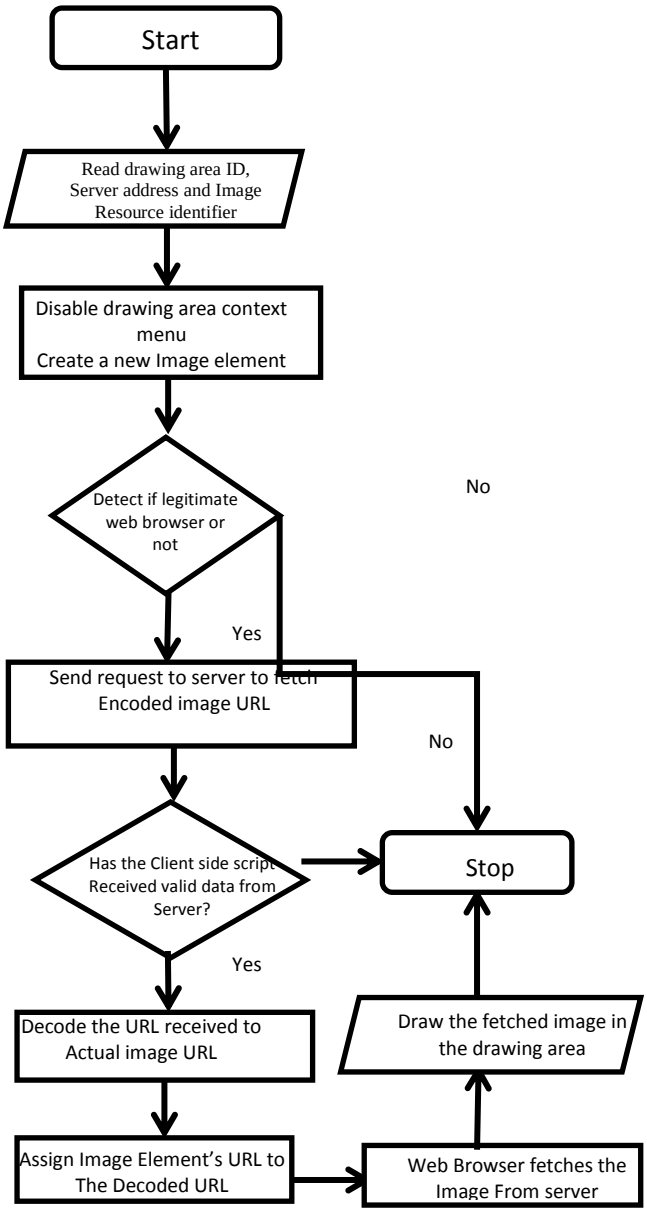
The basic algorithm is as follows:

- Create a default blank element which draws nothing.
- Disable the context menu for the element.
- Make AJAX calls to fetch image URL.
- Draw the image on the element.

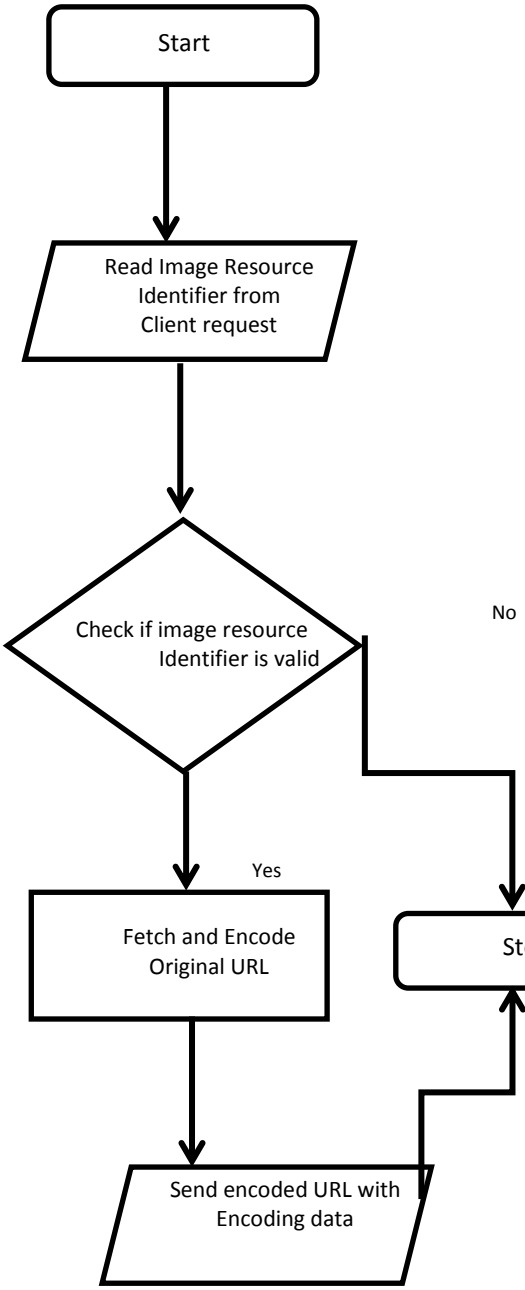
For elements which have associated context menu, this can be disabled via JavaScript. For background image property of elements, context menu does not provide a direct download option.

Flowchart:

Our proposed work is divided into two parts: Server side and Client side.



Client Side Algorithm



Server Side Algorithm

Algorithm

1.2 Client Side Algorithm

Input Parameters:

- ID : Drawing Area ID
- SA: Server address
- IRI: Image resource identifier
- DEC: Decoding routine

Output Parameters:

IURL: Actual image URL

Execution Environment

The following algorithm is assumed to run on a web browser with support for HTML5 and JavaScript with AJAX.

Steps:

1. Let IURL := null, EURL := null, DA := null

2. Select Drawing Area (DA) from ID given i.e. DA := Select(ID)

3. Disable context menu based on element type i.e.

```
switch(DA.type) :
   case ‘div element’ :
break
  case ‘canvas element’ :
DA.disable_context_menu()
  break
```

Create Image Element **IE**,

```
IE := ImageElement{ height := DA.height,
        width := DA.width,
        URL := null,
content := null }
```

5. Detect if the current context is a legitimate web browser. If success, continue with next step otherwise stop

6. Send request to Server at the addrecss SA for the specified Image resource **IRI** to fetch the encoded **URL** (**EURL**) i.e.

```
EURL := Fetch_From_Server(SA, IRI)
```

7. If no data is received from Server, stop. Otherwise continue with next steps

8. Decode URL from the data received from Server by calling decoding routine **DEC**

```
IURL := DEC(EURL)
```

9. Assign IE’s URL property to the decode URL’s value i.e.

```
IE.URL := IURL
```

10. The browser fetches the image from the decoded URL by sending GET request to Server internally

```
IE.content := GET(SA, IE.URL)
```

11. On completion of image loading, the drawing of image on **DA** is done

```
switch(DA.type) :
case ‘div element’ :
   DA.background_image := IE
break
case ‘canvas element’ :
        DA.draw(IE)            // Using canvas built-in methods
Break
```

1.3 Server Side Algorithm

Input Parameters:
Image resource Identifier

Output Parameters:

- Encoding Algorithm details
- Encoded URL

Steps:

1. Receive the request.
2. Check if the request image resource identifier is valid. On a valid identifier, follow the next steps. For an invalid identifier, stop.
3. Determine the actual image URL from the valid identifier.
4. Encode the actual image resource URL.
5. Pack the encoding algorithm information and the encoded URL and send it to the client.

For sending data to client, the data is packed as JSON data and then encoded in an appropriate scheme at the server and sent to client via HTTP.

1.4 Encoder and Decoder

Instead of exchanging URL directly in human readable format, the URL is encoded by a simple ASCII value shifting scheme. The solution does not work with any fixed encoding/decoding scheme but assumes that the user in place provides such a scheme. In order to provide such a scheme, the encoding scheme needs to be specified at the Server and the decoding scheme at the client side.

Screenshots

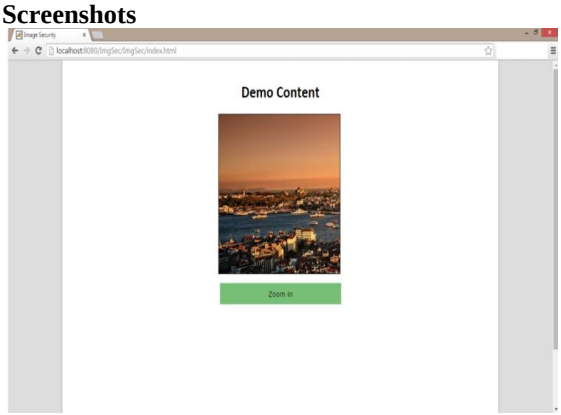


Figure 1: The rendered image in HTML5 Canvas with zoom in button

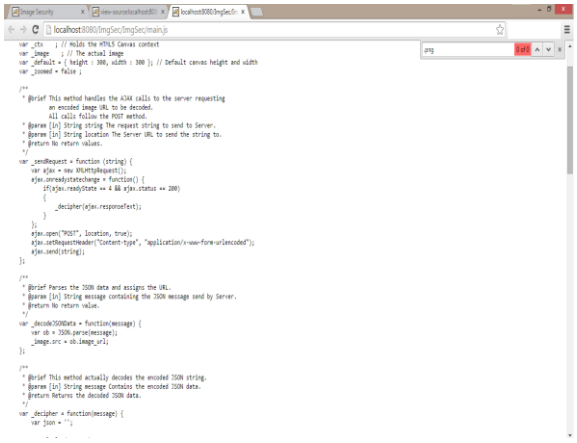


Figure 2: The javascript code does not contain any raw .png links to the image file

Figure 3: The HTML source has no direct link to the PNG image either

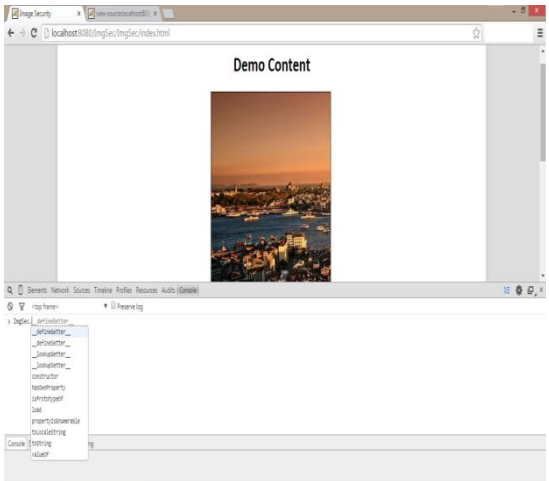


Figure 4: Demonstrates that using the javascript console to access the URL also does not work

Conclusions:

Images in the electronic commerce sites are vulnerable to different types of threats due to their ease of use. The state of the art study has led to the observation that the common tools for image protection, such as watermarking and tiling can be easily over-ridden. It has been seen that the detection of legitimate web browser is a must for fending off data scraping software. In this paper, we have proposed a security solu-

tion for image protection which is This can be achieved by using more sophisticated and granular methods are required to do this.

- The ability to extract the URL directly from JavaScript code or JavaScript console is also not available.

However, the solution in its current form suffers from certain drawback such as

- As the URL is exchanged between server and client side script, if an external agent gains the image URL, it can access the resource via it. This can be avoided with proper configuration at the server.
- URL can be avoided completely if the image data can be transferred to the client side code in proper encoded format. (Base64) This data can be further obfuscated and can be encrypted with HTTPS connection.

The following situation is cases whereby the inherent limitations of browser script execution environment limit the solution's effectiveness and the control for which is not in our hands.

- As the JavaScript on the client side is executed in a closed sand-boxed environment without access to any system level routines, it is impossible to stop the user from taking a screenshot by pressing the "prt sc" button on a standard 108 keys keyboard.
- Taking a photograph via camera with the screen on the image.

A possible remedy of the problem with the existing solution is to create an application instead of a website to let users access the contents. Windows API (Win32) for Microsoft Windows OS lets programs disable screenshots as well and the context menu can be disabled with equal ease. Similarly on Android platform, it can be done with FLAG_SECURE flag which disables the screenshot facility.

References

[1] F. Mintzer, G. Braudaway, and M. M. Yeung, "Effective and ineffective digital watermarks," in Proc. ICIP, Santa Barbara, CA, Oct. 1997.

[2] N. Memon and P. W. Wong, "Protecting digital media content: Water- marks for copyrighting and authentication," Commun. ACM, vol. 41, pp. 35–43, July 1998.

[3] M. D. Swanson, B. Zhu, and A. H. Tewfik, "Transparent robust image watermarking," in Proc. ICIP, Lausanne, Switzerland, Sept. 1996, pp. III 211–III 214.

[4] I. J. Cox, J. Kilian, T. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for multimedia," NEC Res. Inst., 95—10, 1995.

[5] N. Nikolaidis and I. Pitas, "Copyright protection of images using robust digital signatures," in Proc. ICASSP, May 1996.

[6] R. B. Wolfgang and E. J. Delp, "A watermarking technique for digital imagery: Further studies," in Proc. Int. Conf. Imaging Science, Systems, Technology, Las Vegas, NV, July 1997, pp. 279–287.

[7] G. L. Friedman, "The trustworthy digital camera: Restoring credibility to the photographic image," IEEE Trans. Consumer Electron., vol. 39, pp. 905–910, Nov. 1993.

[8] M. M. Yeung and F. Mintzer, "An invisible watermarking technique for image verification," in Proc. ICIP, Santa Barbara, CA, Oct. 1997.

[9] C.-Y. Lin and S.-F. Chang, "A robust image authentication method surviving JPEG lossy compression," Proc. SPIE, vol. 3312, pp. 296–307, 1998.

[10] "Issues and solutions for authenticating MPEG video," in Proc. SPIE/IS&T Symp. Electronic Imaging, Security, Watermarking Multi- media Contents, San Jose, CA, Jan. 1999.

[11] R. L. Rivest, "The MD5 message digest algorithm," Tech. Rep., 1992.

[12] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining dig- ital signatures and public-key cryptosystems," Commun. ACM, vol. 21, pp. 120–126, Feb. 1978.

[13] P. W. Wong, "A watermark for image integrity and ownership verification," in Proc. IS&T PIC Conf., Portland, OR, May 1998.

[14] "A public key watermark for image verification and authentication," in Proc. ICIP, Chicago, IL, Oct. 1998.

[15] M. Holliman and N. Memon, "Counterfeiting attacks on oblivious block-wise independent invisible watermarking schemes," IEEE Trans. Image Processing, vol. 6, pp. 432–441, Mar. 1997.

[16] D. Coppersmith, F. Mintzer, C. Tresser, C. W. Wu, and M. M. Yeung, "Fragile imperceptible digital watermark with privacy control," in Proc. SPIE/IS&T Electronic Imaging Symp., Security, Watermarking Multi- media Contents, San Jose, CA, Jan. 1999.

[17] G. W. Braudaway, K. A. Magerlein, and F. C. Mintzer, "Color correct digital watermarking of images," U.S. Patent 5530759, June 1996.

[18] Secret and public key authentication watermarking schemes that resist vector quantization attack.

[19] R. W. G. Hunt, The Reproduction of Color in Photography, Printing & Television, 4th ed. London, U.K. : Fountain, 1987.

[20] M. Wu and B. Liu, "Watermarking for image authentication," in Proc. ICIP, Chicago, IL, Oct. 1998.

[21] Lihua Ruan, Ding Tian, "A Research of Trust Based on E-Commerce", ISECS, 2008, Electronic Commerce and Security, International Symposium, Electronic Commerce and Security, International Symposium 2008, pp. 776-779, doi:10.1109/ISECS.2008.166