FoxBeacon: Web Bug Detector Implementing P3P Compact Policy for Mozilla Firefox

Chaiyong Ragkhitwetsagul
Carnegie Mellon University
Pittsburgh, PA
cragkhit@cs.cmu.edu

ABSTRACT
This paper describes the design, implementation, and evaluation of a web bug detector called "FoxBeacon." This detector is originally designed to be an extension of Mozilla Firefox. After being installed, FoxBeacon embeds itself into the Firefox browser and acts as a proxy. It reads every incoming web page and trying to find hidden web bugs. FoxBeacon treats each image as a web bug according to the pre-defined rules. In addition, FoxBeacon also includes the compact policy, one component of Platform of Privacy Policy (P3P), to incorporate and provide more information to its users. This paper also covers the evaluation of FoxBeacon functionalities and its results. The ultimate goal of this project is to optimize between least disruption to the browser's users and the most effective way in order to make the users aware of the hidden web bugs. In addition, reducing the number of false negatives and false positive is also another important goal.

Categories and Subject Descriptors
D.4.6 Security and protection, H.1.2 User / Machine systems, H.5.2 User interfaces

General Terms
Design, Human Factors, Security

Keywords
FoxBeacon, web bug, web beacon, pixel tab, clear gif, privacy, Mozilla Firefox, extension, P3P, compact policy, web browser, cookies, third-party cookies

1. INTRODUCTION
"Web beacon" or "web bug" (also known as pixel tag, and clear gif) is a privacy threat that is difficult to detect. Its name "web bug" comes from an English word means a tiny hidden microphone. It could be a tiny transparent picture embedded anywhere in a web page, and its major task is implicitly collect data about of a web users. See more definition of web bug in section 3.1. In this paper the term "web beacon" and "web bug" will be use, and they refer to exactly the same thing [13, 22, 23].

The detection of web bugs is based on the rule sets that characterize web-bugs. This characterization using rule sets is not conclusive and consider complete and formal. As a result, it can lead to many false-positives or false-negatives based on a more relaxed or more constrained rule set. Initial work done by Alsaid et.al [1] has resulted in a web bug detection tool called Bugnosis. Bugnosis is a web bug detector for Internet Explorer 6. It uses the pre-defined rule sets to categorize images. It also allows addition and deletion of new rule sets to detect web bugs. The most valuable part of Bugnosis is the rule updating feature via a centralized repository. Though Bugnosis is a very effective tool, its uses are limited to the fact that it can only support Internet Explorer. This binding to a particular browser make it difficult for people using other popular browsers to take advantage of Bugnosis. This project aims to create a web bug detector as an extension of famous Gecko-based web browsers (i.e. Mozilla Firefox). It is also enhanced by integrating new important features, such as using benefits of compact policy which is a component in The Platform for Privacy Preferences (P3P) helping in categorizing web bugs. This would greatly increase the efficiency of web bug detector. See more detail about compact policy in section 3.5.

2. BACKGROUND AND RELATED WORK
As one interesting quote regarding web bugs gathered from the Network Advertising Initiative [12] says:

> "Web beacons are a tool that can be used online to deliver a cookie in a third party context. This allows companies to perform many important tasks – including unique visitor counts, web usage patterns, assessments of the efficacy of ad campaigns, delivery of more relevant offers, and tailoring of web site content. The web beacon's cookie is typically delivered or read through a single pixel on the host site."

Web bug have been treated as privacy invasive. Regarding to the Privacy Foundation, there are many suspicious reasons of using web bugs:

1. Collecting how many times each web page has been viewed.
2. Sending some personal information data, such as gender, age, address, to marketing web sites. This could be useful for creating user profiles.
3. Being able to track one's surfing habit by tracking the user across many different web sites.
4. Collecting a person's search string to marketing websites.
5. Matching what a user buy from an advertisement link.
6. Counting the number of times of loading of a particular advertisement.
7. Collecting information of the browser that a user is using. This could be useful for deciding changing the content of a web page.

One advertising company can disseminate their web bug on other different company web sites. These web bugs generate cookies on

the user's machines. These web bugs from the advertising company can be used to track the web surfing behavior of a user.

For example, if one user (Mr. Know Nothing) visits a web site www.ihavewebbug.com which contains a web bug from an advertising company called I-Use-Web-Bug. The web bug on ihavewebbug.com will place a cookie on Mr. Know's machine. This means that when Mr. Know visits another website called www.ialsohavewebbug.com which also has an embedded web bug of the same company, and the browser tries to load the web bug (it is just an image), the cookie stored in the machine will be fetched to the web server of I-Use-Web-Bug Company. The Company can easily gather the data and create profile of each Internet user. These profiles are valuable for them to decide the plan of placing the future advertisements.

From the privacy perspective, using of web bugs (third party cookies) for tracking the Internet users' behaviors is obviously privacy invasive. This claim is valid because it implicitly collects the users' data without any consent. Moreover, there is no way for the users to opt-out. It is worth mentioning that this topic is unaware by most of the Internet users, and there are limited numbers of researches done regarding this particular problem [7].

2.1 Web bug research

The previous research on web bug includes study for defining the criteria of web bugs, and implementing a tool for detecting web bug.

2.1.1 Bugnosis: Detecting web bug with Bugnosis

Adil Alsaid and Devid Martin [1, 12] had dedicated their time in creating a tool call Bugnosis. It is an add-on for Internet Explorer. It plugs itself seamlessly into the browser and generates a warning whenever it discovers a web bug in a particular web page. The main policy of Bugnosis is to increase the awareness of the web bug to the internet users without blocking the web bug at all. On the other hand, it does provide the notifications for the user to increase their privacy awareness.

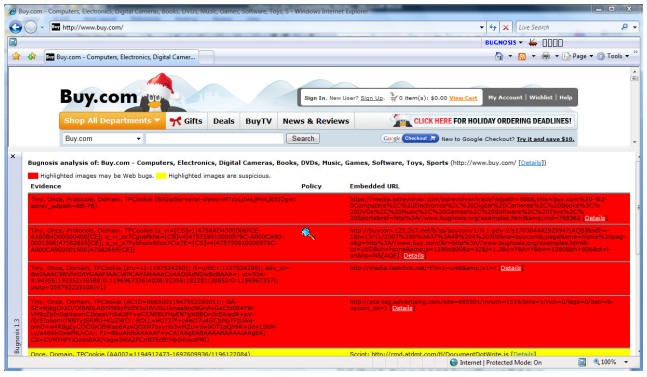

Figure 1 Bugnosis for Internet Explorer

From the implementation perspective, Bugnosis uses Document Object Model (DOM) for gathering all images from a web page. This is a more efficient way comparing to parsing a raw HTML document, and it also provides a way to insert additional items inside the loaded web page. Bugnosis notifies its user by generating an alert sound, "uh-oh!" when it finds a suspicious image. It also places a small picture of a walking bug inside the web page where the web bug is placed. This method increases clarification of web bug found and illustrates the location of the web bug. Bugnosis also provides a ways to contact web site's administrator via email to inform the web bug problem. It has been improved to include the database called 'expert database' collecting regular expressions of prohibited lists of regular expressions of suspicious web sites, also the white lists of the allowed web sites. This expert database can be modified by the users. Other technologies used are COM, ATL, and ActiveX. Up until now, Bugnosis has more than 100,000 users. This could be a good proof of its efficiency. Unfortunately, this tool is not fully compatible with the newest Internet Explorer 7.

2.1.2 Flexible Web Bug Detection

Like Bugnosis, Fabiano Fonseca, Robert Pinto, Wagner Meira Jr [TBA] from Federal University of Minas Gerais, Brazil did the similar tool called "Web Bug Detector", but for Mozilla browser. This major reason they chose Mozilla browser is because it is an open-source browser and it is easy to create a plug-in and user interfaces.

Their paper also includes the detailed implementation and provides the results of the experiments in the real work load environment by creating another stand-alone version of their tool with the same features and functionalities. They have tried more than 3,000 HTTP host names, and also other host types (HTTPS, FTP) for more than 3,000 host names. The analyzing of the results has some interesting information:

- There is 1 web bug detected per 18.6 HTTP request.
- More than 90% of web bugs associate to commercial web sites ending with *.com*.
- Top 25 web sites comprise more than 95% of all web bugs.
- Top 25 web sites response for more than 60% of web bugs in other unique web-bug-embeded websites.

The Flexible Web Bug Detector performed really well with 80% capture rate from total web bug occurrences found in another study [9].

2.1.3 Web Bug in Comtemporary Use

David M. Martin JR., Hailin Wu, and Adil Alsaid [12] tried to distinguish between intentionally data collection for surveillance purposes from the common web site's operation. They answered the problem with the solution that web sites which intentionally collect data from their users contain some elements which are dedicatedly design without any relation to the actual content of the web sites. As a result, they came up with the strong definition of the element treated as web bugs (see details about the definition of web bugs in section 3.1).

Furthermore, in using benefits of privacy policy, they also found that 29% of web-bug-enable web sites have privacy policies which say nothing about third-party contents in their web sites. The paper also mentions about P3P policy as an excellent possible solution in decreasing the human effort to read the actual privacy policy by delegate this workload to the automatic system. Anyway, there is one concern about the ignorance to comply with the privacy policy. See detailed definition of P3P policy from section 3.4.

2.2 Why Mozilla Firefox Extension?

For Mozilla Firefox (Gecko-based), the market shares in a browser market is 14.88 % and 77.86% for Microsoft Internet Explorer based [2, 10]. The market share for Mozilla Firefox is a small percentage as compared to Internet Explorer. However, given that a total number of internet users is 1,175 million [11]. Mozilla Firefox's modest share of 14.86% still covers a large number of users. Our project is to implement an extension for Mozilla Firefox; this could leverage the advantages of web bug detector.

Moreover, another major reason is Mozilla, opposites of Internet Explorer, is an open-source which has a lot of supporting knowledge, and tools. Mozilla Firefox uses XML User interface Language (XUL) for generating the user interfaces. This language is flexible and easy to implement.

3. DEFINITIONS

3.1 Web Bug Definitions

General definition: Web bug is any HTML element especially image which is created for two purposes: (1) implicitly embedded in the web page and (2) trying to collect data from users.

Specific definition: a web bug can also be defined by its following properties [1, 3, 14, 16, 17, 18]:

1. *Image's domain name is different from the URL's domain name*: the element has this property when the two right most dot-separated components in the URL (or two highest DNS levels) are different. For example, the website www.cmu.edu is different from www.mit.edu because the two right most component (cmu.edu and mit.edu are different). On the other hand, www.cs.cmu.edu doesn't have different host name from www.cmu.edu because their two right most are the same (cmu.edu).
2. *Image size is less than or equal to 7 pixels:* there is no useful purpose of using an image which its size is very small (less than 7 pixels). One possible reason is using it for justify the alignment of the web page. We can filter out this false positive by using other rules.
3. *Image has third-party cookie:* an images which sets a cookie on the user's machine when it is being loaded is very doubtful because it could be used for tracking user's surfing habit and creating the user profile.
4. *Appear only once:* it is assumed that an image which appears only once in a web page is more likely to be a tracking device especially when incorporated with other rules.
5. *Image's URL contains more than one protocol:* the protocol is from the set of 'http:', 'https:', 'ftp:', and 'file:' An example of an image's URL which consists of more than one protocol is:

   http://track.example.com/log/ftp://www.source.com

   This is because it has both http and ftp protocols. This property is useful because the additional protocol which is included in the URL may indicate some tracking information.
6. *Image's URL is lengthy:* the image which has lengthy URL seems to communicate something suspicious in its URL. The definition of 'lengthy' in this project is separated into two categories:

   (1) The image's URL contains only one element: the URL which has only one element (e.g. http, or ftp) is lengthy when it is longer than 100 characters.

   (2) The image's URL contains more than one element: the URL which has more than one element is lengthy when its length conforms to this threshold value:

   $$Threshold = \mu + 0.75\sigma$$

   μ is the mean of length of all images within the web page, and σ is the standard deviation of all images' string sizes.

In the perspective of FoxBeacon project, an image which includes main four properties (1-4) is considered as a web bug. Additional two properties (5-6) are used to increase the level of severity.

3.2 FoxBeacon Web Bug Definition

An HTML image which (1) has different domain name (2) has tiny size (3) has third-party cookie (4) appears only once.

3.3 Cookie Definition

Cookie is a parcel of text sent by a server to a browser and collected in the client machine. The main purpose of cookie is for personalization. Cookie is primarily used for remember a particular user by reading the user's cookie. Each time the user visits a web site which planted its cookie on the user's client machine, the cookie will be sent back to the server. Cookie can contain some sensitive personal data, such as, username, password, and settings. This facilitates the user because the user doesn't have to fill in the information every time he visits the same website [2].

3.4 The Platform for Privacy Preferences (P3P)

Cranor L. F (2002) [6] states the definition of P3P as the following:

> "The Platform for Privacy Preferences (P3P) is a standard for communicating the privacy policies of web sites to the clients that connect to them. With P3P, a web client can retrieve a machine-readable privacy policy from a web server and respond appropriately."

P3P is a new invention dedicating to transform the human-readable privacy policy into machine-readable format. The original human-readable privacy policy uses a lot of legal terms which makes it really difficult to understand. As a result, few people read privacy policy when they visit a web site. P3P is a solution to this problem as it delegates the responsibility of reading the privacy policy to the machine.

P3P policy is in XML format and has its own predefined elements which cover most of possible statements in the human-readable privacy policy. See the P3P (1.0) specification from http://www.w3.org/TR/P3P/ [19].

3.5 P3P Compact Policy Definition

A short summary of a full P3P policy is called 'compact policy.' The main use of compact policy is for optimization by allowing the processing of "cookies" before retrieving the full P3P policy. Compact policy should be used only when a web site enables

using of cookie. Usually, compact policy is included as a part of HTTP response header. Below are examples of P3P compact policy from HTTP response header of www.microsoft.com using P3P Validator [26]:

```
Cache-Control: private
Date: Thu, 29 Nov 2007 13:53:24 GMT
Location: /en/us/default.aspx
Server: Microsoft-IIS/7.0
Content-Length: 136
Content-Type: text/html; charset=utf-8
Client-Date: Thu, 29 Nov 2007 13:53:24 GMT
Client-Peer: 207.46.192.254:80
Client-Response-Num: 1
P3P: CP="ALL IND DSP COR ADM CONo CUR CUSo IVAo IVDo PSA
PSD TAI TELo OUR SAMo CNT COM INT NAV ONL PHY PRE PUR
UNI"
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
```

The highlighted part refers to the location of compact policy. Each abbreviation has its own meaning regarding to the elements in full P3P policy. See the specification of P3P compact policy from http://www.w3.org/TR/P3P/ [19].

4. DESIGN AND IMPLEMENTATION

4.1 XUL, DOM, and JavaScript

FoxBeacon, the new web bug detector is designed to work as an extension for Mozilla Firefox. Thus, it uses the technology of Document Object Model (DOM) and XML User Interface (XUL) language. XUL is used to create the Graphic User Interface (GUI), and DOM is a way to retrieve a web page (HTTP document) from the browser for analysis [4, 7, 8, 15, 21]. Using DOM to get access to the HTML elements has a lot of benefits:

- No need to parse raw HTML files- parsing a HTML file can produce some errors and complicate the implementation.
- Getting access to the document elements allows FoxBeacon to insert its own elements into the HTML document (web page) before showing it to the user.

In order to make FoxBeacon response to the action of web page loading, it has to include JavaScript files. The JavaScript acts like other programming languages. Because one way to work with DOM is using JavaScript and most of Mozilla Firefox extensions implementing JavaScript, I decided to include JavaScript as the major part of FoxBeacon.

4.2 P3P Compact Policy

For enhancing the efficiency in detecting web bugs, FoxBeacon implements using of P3P compact policy (see definition of P3P compact policy from section 3.5). There are several approaches to include compact policy in the web site. One possible way is including it in the HTTP header, so web browser can read it and decide their responses to a particular web page. Internet Explorer 6 heavily relies upon the compact policy. If it finds any web page which has third-party cookies (the source location differs from the domain), but there is no P3P compact policy regarding their existence or full P3P policy. These cookies will be blocked by default by Internet Explorer 6.

In this project, FoxBeacon uses P3P compact policies appearing in the HTTP response of each image treated as a web bug. The motivation is to examine the purpose why the web bugs's owner placed his web bug on another web site. FoxBeacon doesn't rely on P3P compact policy in judging an image as a web bug. It just provides all P3P compact policies gathered from the web bug originating web site to the user to convey more information to the user. Because nobody actually knows why one person places a hidden image in another person's web page, FoxBeacon passes on this decision to the user.

4.3 FoxBeacon Implementation

4.3.1 Graphic User Interface (GUI) and Actions

FoxBeacon's user interfaces are built from XML User Interface Language (XUL). The XUL user interface can be read and generated by Firefox browser. Each particular piece of UI is one XUL file. Mozilla Firefox's default GUIs are also created from XUL language. Thus, seamlessly attaching FoxBeacon to the browser is really expected.

Only XUL cannot create any dynamic action. It has to associate its elements to actions in JavaScript. As a result, all actions performed by FoxBeacon are included in different JavaScript files. Both XUL files and JavaScript files are placed together in the same place [14, 21, 24]. Extension of Mozilla Firefox has its own particular file and directory structure. Figure 1 shows the directory structure of an extension named 'extension.xpi.'

```
extension.xpi:
              /install.rdf
              /components/*
              /components/extensionOverlay.xul
              /components/extension.js
              /defaults/
              /defaults/preferences/*.js
              /plugins/*
              /chrome.manifest
              /chrome/icons/default/*
              /chrome/
              /chrome/content/
```

'extensionOverlay.xul' file contains everything about the main GUI and it associates actions to be performed with functions inside 'extension.js' file. These two files are the most important part of the software.

4.3.2 Algorithms

4.3.2.1 Retrieving all images elements

When a user use Mozilla Firefox enhanced by FoxBeacon to request a web page, FoxBeacon starts by getting the DOM of the web page and retrieves all image elements inside the DOM document. Then, it starts categorizing each image in the collected image lists by matching with the rule sets. When it finds a suspicious image conforming to the rule sets, it keeps that image in the result lists. After all images in the list have been processed, FoxBeacon starts retrieving the HTTP header of the web page for collecting the cookie setting and P3P compact policy of a particular image by creating a HTTP request to the image's source URL. It collects all responded compact policy and cookies in its compact policy list and cookie setting list respectively. Then, FoxBeacon compares each image in its image lists to the cookie setting lists. Which image sets a cookie on the machine will fall into the web bug lists. Finally, FoxBeacon shows the web bugs lists to its user by displaying the blinking image of a beacon in the

status bar of Mozilla Firefox (see Figure 2). The algorithm is shown below:

```
START
If (webpage.load) then
        while (DOM.end-of-file) {
                elm = getImageElm();
                addToImgList(elm);
        }
        Foreach (img in ImgList) {
                result = applyRules(img);
                if (result) then
                        addToResultList(img);
        }
        Foreach (img in ResultList) {

        getHTTPHeader();
        cpList = collectCompactPolicy();
        if (img set cookie) {
                addToWBList(ResultList);
        }
showWBList();
END
```

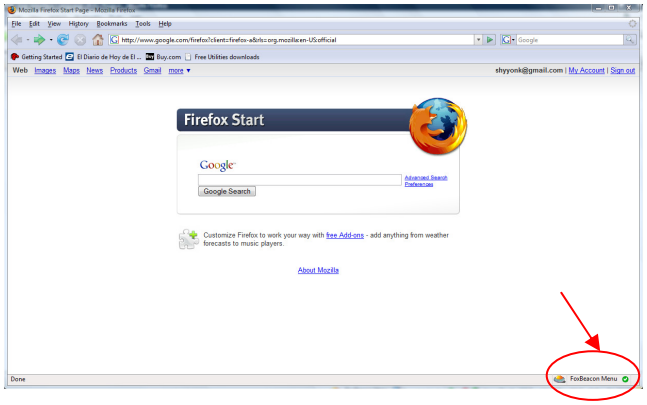

Figure 2 FoxBeacon icon is showing on the status bar.

Every time the user loads a new web page, FoxBeacon checks all the images in the content of that particular web page. If FoxBeacon found web bugs in a web page, its icon will turn to green and red and blinking to notify the user.

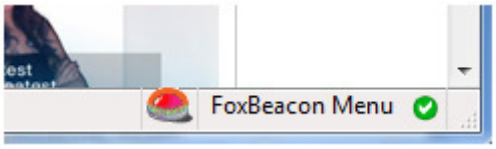

Figure 3 FoxBeacon is blinking when it found web bugs

When the user is notified, he can see the web bugs found by right-clicking on the menu and select “See Webbugs.”

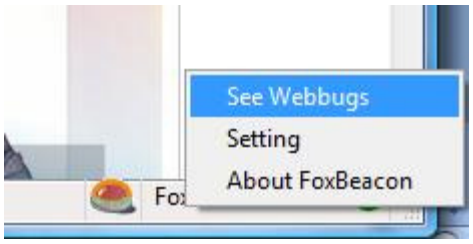

Figure 4 the results from selecting "See Webbugs" menu

The result pane will appear and contain all lists of web bugs found.

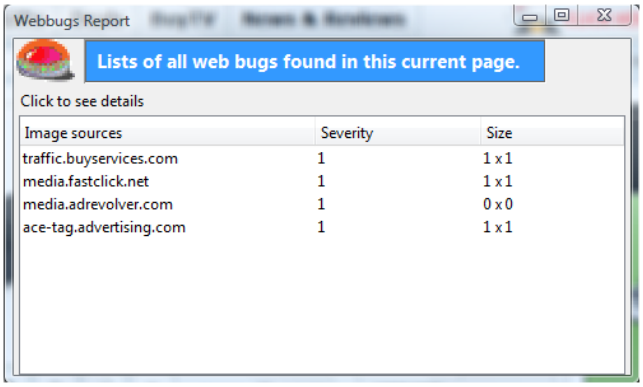

Figure 5 Webbugs Report Dialog

The user can click on each particular item in the list to see more details. The detail information dialog contains the following information:

(1) *Matched Rules:* All the rules which the web bug conforms to.

(2) *Why is it treated as a web bugs?* The explanation of the reason why this image is treated as a web bug. The information is applied from all the rule sets.

(3) *Why is it placed here?* The information extracted from the P3P compact policies. They are separated into different categories for ease of understanding.

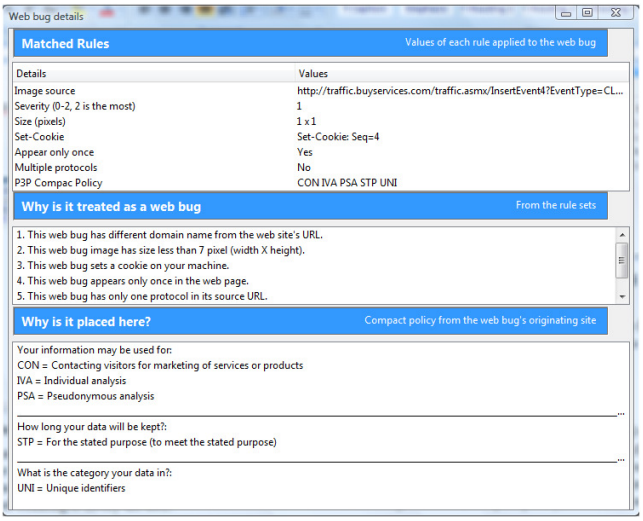

Figure 6 Detailed Information Dialog

From figure 6, an image from traffic.buyservices.com is treated as a web bug because it has different domain name from the web page where it resides and has a very small size (0 and 1 pixel), it also places a cookie on the user's machine, and it appears only once in the web page.

Likewise, from the P3P compact policy section, the reason of placing this web bug seems to be for (1) contacting the visitor for marketing of services or product (2) for individual analysis and (3) for pseudonymous analysis. It also shows the policy about retention of the data, and category of the data being collected from the user.

4.3.2.2 Process of Detecting Web Bugs

FoxBeacon is an extension of Mozilla Firefox browser, so it is able to access the document (web page) being loaded by the browser. It places itself at the same level as the web browser. While the web browser is processing a web page to be displayed the monitor screen for its user, FoxBeacon is running in background, gathering all images and report to the user when it found web bugs. The figure 7 illustrates the process.

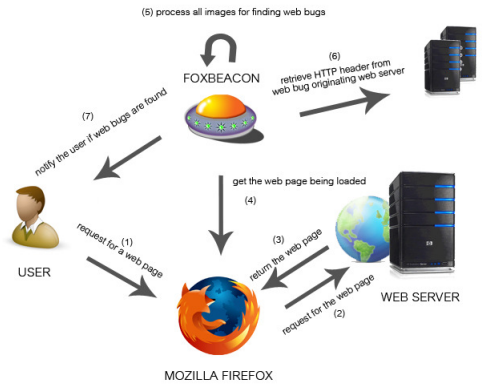

Figure 7 Process of FoxBeacon in finding web bugs

5. EVALUATION

The testing of FoxBeacon is complicated because the difficulty in finding a web page which contains web bugs inside. I used Bugnosis as a baseline because it is the only one web bug detector and it has improved a lot by releasing for public uses. I followed the examples from Bugnosis.org web site [5] and found that some web sites listed in Bugnosis.org have already removed web bugs out of their web pages already. However, there are still some web sites containing web bugs left. I also found other web sites from online documents [19, 25], and from my classmates. However, due to the limit of time, I did not complete an extensive testing by using a lot of various web sites, but this is in the Future Work section.

5.1 Testing With Websites

According to Bugnosis.org, there are three websites which still contain web bugs includes www.buy.com, http://freedownloadscenter.com/Utilities/, and http://www.mycomputer.com/agreements/privacy_policy.html.
For buy.com, using Bugnosis on Internet Explorer 7 found 4 web bugs. Using FoxBeacon on Firefox 2.0.0.11 also found 4 exactly the same web bugs. Another web site is freedownloadscenter.com, FoxBeacon found 1 web bug which is the same as Bugnosis. Other result can be seen from table 1.

Table 1 Results of FoxBeacon compare to Bugnosis

Web sites	Possible web bugs	Number of web bugs found	
		Bugnosis	FoxBeacon
www.us.buy.com	4	4	4
http://freedownloadscenter.com /Utilities	1	1	1
http://www.mycomputer.com/agreements/privacy_policy.html	1	1	1
http://www.elsalvador.com/	1	1	1 (with problem of frames)
http://www.mycomputer.com/agreements/privacy_policy.html	1	1	5

Although the result of the evaluation is small, it still can demonstrate the efficiency and problems of FoxBeacon. FoxBeacon performed very well for the first 3 web sites. However, for the fourth web sites, it missed one image because it resides within a frame. After open a new window with has only that frame, FoxBeacon could detect it perfectly. For the last web site, FoxBeacon missed few web bugs found by Bugnosis. This could be from the different of applying rule or from the bug in FoxBeacon. This issue will be investigated in future work.

6. FUTURE WORK

Many technical problems arose during implementing FoxBeacon. Some problems were solved but, due to limit of time, some still left. The future work of FoxBeacon will include these topics:

(1) *Fixing the problem of web page using frames:* following the result of section 5, FoxBeacon cannot detect web bugs reside in a web page comprising frames.

(2) *Testing with more pool of web-bug-enabled sites:* increasing the number of suspicious web sites which contain web bugs could help improving the efficiency of FoxBeacon and also reveals the problems or bugs within the program.

(3) *Generate more human-friendly result of P3P compact policy:* compact policies are lists of characters. Naïve users which do not have knowledge in computer or privacy technology are completely confused by the result. This is the big problem because they will ignore the result they cannot understand. Generating human-friendly result for FoxBeacon will alleviate this problem.

7. CONCLUSION

Web bug which is a tiny hidden image on a web site used for tracking purposes. It is privacy invasive and few people know their existing. FoxBeacon is a web bug detector intentionally designed as an extension of Mozilla Firefox. It implements many technologies, e.g. XUL, DOM, JavaScript, P3P. It plugs itself to Mozilla Firefox, and retrieves all images in the web page being loaded by the browser. When it finds web bugs, it notifies the user

and shows the detailed information regarding the web bugs. The result of this project is satisfying because FoxBeacon can detect myriad number of pre-proven web bugs. However, it still has a lot of flaws which has to be fixed. All of the problems found during evaluation and testing will be included in the future work. This would make FoxBeacon to be more reliable and accurate in the future. FoxBeacon latest version is available for downloading from www.cs.cmu.edu/~cragkhit/foxbeacon.

8. ACKNOWLEDGMENTS

Thanks to Professor Lorrie Faith Cranor, and Kami Vaniea (my TA) for contributing as my advisors in this project. Also thanks for Nichayada Ratana for proofing read this paper.

9. REFERENCES

[1] Alsaid A., Martin D., Detecting Web Bugs With Bugnosis: Privacy Advocacy Through Education. Available from http://www.cs.bu.edu/~dm/pubs/bugnosis-pet2002.pdf (2002); accessed 27 September 2007.

[2] Bennett C. J., Cookies, web bugs, webcams and cue cats: Patterns of surveillance on the world wide web. In Ethics and Information Technology (3) 197–210, 2001.

[3] Beware Of Web Bugs & Clear GIFs . Available from http://www.smartcomputing.com/editorial/article.asp?article=articles/archive/g0804/11g04/11g04.asp; accessed on 1 November 2007.

[4] Building an Extension. Available from http://developer.mozilla.org/en/docs/Building_an_Extension; accessed on 30 October 2007.

[5] Bugnosis. Available from http://www.bugnosis.org/examples.html; accessed on 1 December 2007.

[6] Cranor L. F., Chapter 7: Creating P3P Policy. In Web Privacy with P3P. O'Reilly, California, 2002, 110-132.

[7] Chung W., and Paynter J., Privacy Issues on the Internet. In Proceedings of the 35th Hawaii International Conference on System Sciences, 2002.

[8] Creating Applications with Mozilla. Available from http://books.mozdev.org/html/index.html; accessed on 30 October 2007.

[9] Development resources. Available from http://kb.mozillazine.org/Development_resources#XUL; accessed on 30 October 2007.

[10] Market Share by Net applications (2007) *Operating System Market Share for February 2007* Retrieved March 5, 2007 http://marketshare.hitslink.com/report.aspx?qprid=2.

[11] INTERNET USAGE STATISTICS - The Big Picture. Available from http://www.internetworldstats.com/stats.htm; accessed 24 October 2007.

[12] Martin D., Bugnosis Web Bug Detector. Available from http://www.bugnosis.org; accessed on 13 November 2007.

[13] Martin D., Wu H., and Alsaid A. Hidden surveillance by Web sites: Web bugs in contemporary use. Available from http://portal.acm.org/citation.cfm?id=953509&coll=GUIDE&dl=ACM&CFID=4140258&CFTOKEN=89024307; accessed on 20 October 2007.

[14] McCandlish S., EFF's Top 12 Ways to Protect Your Online Privacy. Available from http://w2.eff.org/Privacy/eff_privacy_top_12.html; accessed on 10 November 2007.

[15] Mozilla Devaloper Center. Available from http://developer.mozilla.org/; accessed on 10 November 2007.

[16] Olsen S., New tools hatch for sniffing out Web bugs. Available from http://www.news.com/2100-1023-253517.html; accessed on 13 November 2007.

[17] Olsen S., Privacy advocates shine light on "Web bugs". Available from http://www.news.com/Privacy-advocates-shine-light-on-Web-bugs/2100-1023_3-250230.html; accessed on 22 October 2007.

[18] Richard M. Smith, Why Are They Bugging You?. Available from http://www.privacyfoundation.org/resources/whyusewb.asp; accessed on 20 October 2007.

[19] Smith R. M., The Web Bug FAQ. Available from http://w2.eff.org/Privacy/Marketing/web_bug.html; accessed on 10 November 2007.

[20] The Platform for Privacy Preferences 1.0 (P3P1.0) Specification. Available from http://www.w3.org/TR/P3P/; accessed on 30 October 2007.

[21] Use of Internet "Cookies" and "Web Bugs" on Commerce Web Sites Raises Privacy and Security Concerns. Available from http://www.oig.doc.gov/oig/reports/2001/OS-OSE-14257-04-2001.pdf; accessed on 24 October 2007.

[22] Web bugs. Available from http://www.spywareinfo.com/articles/webbugs/; accessed on 8 November 2007.

[23] Web bugs spying on net users. Available from http://news.bbc.co.uk/2/hi/science/nature/1493152.stm; accessed 20 October 2007.

[24] Writing an Extension for Firefox. Available from http://www.orablogs.com/duffblog/archives/000536.html; accessed on 30 October 2007.

[25] The Web Bug FAQ. Available from http://w2.eff.org/Privacy/Marketing/web_bug.html; accessed on 30 November 2007.

[26] W3C P3P Validator, Available from http://www.w3.org/P3P/validator.html; accessed on 16 October 2007.