Website Malicious Codes Detection

M.Sc. Hisham Raad Jafer

Abstract

As we all know Internet is a very widely used network, containing different types of multimedia, documents, programs and websites, for that reason it is difficult to maintain it as a malicious-free environment, these malicious codes are embedded into an innocent looking website, these websites in turn are very wide spread sites. An important feature a malicious code depends on reaching to as many users as it can get.

The proposed system is implemented generally to find and detect a malicious code in Internet websites, because of the wide spread of vandal users and malicious programs, the need for that tool has become a flat fact. A targeted website is analyzed for existing code and if that code is does not exist then its links are followed individually to check whether or not there is an indirect malicious code, this approach is useful in rapidly growing websites.

The system is run on the server and when the client makes its request to load a website this request is sent first to the server then the system follows that on line to internet by (URL) and checks the contents of its website and follows all its sub-links in that website then if the website is clean (malicious free) then the client is granted access to that site, otherwise if it detects a malicious code and its type it prepares a report to the user about the infections and type of malicious code as a warning message check result area. This system has been implemented by using Visual Basic (VB) language version 6.0.

1. Introduction

The system ***Website Malicious Code Detection (WMCD)*** is detecting malicious code. This program aims to give details of structure analyzing website tags (links) the HTML file (source code), content link analysis, and detect malicious code (**viruses, Trojan horse, worm)** in existing links in the web page.
Because of the complexity, computers and computer networks have become a target of computer crime more and more often. Large theoretical and practical efforts are concentrated today on this problem. Nevertheless, a perfectly secure system is still a myth. Many modern computer system still lack properly implemented security services, contain a variety of vulnerabilities exploited by threats, analyzing website techniques are used to strengthen the system security and increase its resistance to internal and external attacks {1},{4}.

After the user queries the internet, the internet displays many result, user selects one of these website to download it, before that the system begins to download the webpage ability to open HTML file (source code), steps are that guide to structure analyze webpage tags (links) and call database content many kinds of malicious code. This database flexible update by add new malicious code or delete; matches between them if contain malicious code, analyze the content links if it is real when this link is connect to other page it is real and important to detect malicious code in this webpage {1}.

2- Malicious Code

Malicious code is any software program designed to move from computer to computer and network to network, in order to intentionally modify computer systems without the consent of the owner or the operator includes viruses, Trojan horses, worms, script attacks, and rogue internet code. The intentional part of the definition is important design flaws in the Microsoft windows operating system which is responsible for more data loss than all the malicious code put together, but windows wasn't intentionally designed

to destroy your data and crash your system. Today, add all harmful programs created with scripting languages and empowered by internet technologies: macro viruses, HTML, java applets, ActiveX, VBScript, JavaScript, and Instant message {10} .

2-1 Major Type of Malicious Mobile Code {11}

Most malicious code program can be categorized as a virus, Trojan, worm, or mixture. A rogue program may be written in assembly language, c++, java, or in visual basic for applications (VBS), still it but is classified as one of these major types. The malicious program functions as two or more of this type.

- **Virus**

  A virus is a malicious program that modifies other host files or boot areas to replicate (a few exceptions). In most cases the host object is modified to include a complete copy of the malicious code program. The subsequent running of the infected host files or boot area then infects other objects.

- **Trojan**

  A Trojan or Trojan horse is a non replicating program masquerading as one type of program with its real intent hidden from the user. For example, a user downloads and runs a new, free version of his favorite multiplayer game from a website. The game promises thrills and excitement. But its true intent is to install a Trojan routine that allows malicious hackers to take control of the user's machine. A Trojan does not modify and infect other files.

- **worm**

  A worm is a sophisticated piece of replicating code that uses its own program coding to spread, with minimal user intervention. Worms typically use widely available applications (email, chat channels) to spread; a worm might attach itself to a piece of outgoing email or use a file transfer command between trusted systems. Worms take advantage of holes in software and exploit systems. Unlike viruses, worms rarely host themselves within a legitimate file or boot area.

2-2 Malicious Code Spread

There are many ways to spread malicious code, but here is the most popular scenario. The author writes the rogue program and posts it to a (virus exchange). A spreader downloads the program and sends it to a legitimate, unsuspecting site. It can be sent as a Trojan file or emailed as an attachment to an email list group. The unsuspecting users execute the file, which can then infect other files or take control of their system. The user emails the malicious code to another friend {11}.

2. The proposed system

This system is implemented in three levels; levels are show in the Figure (1)

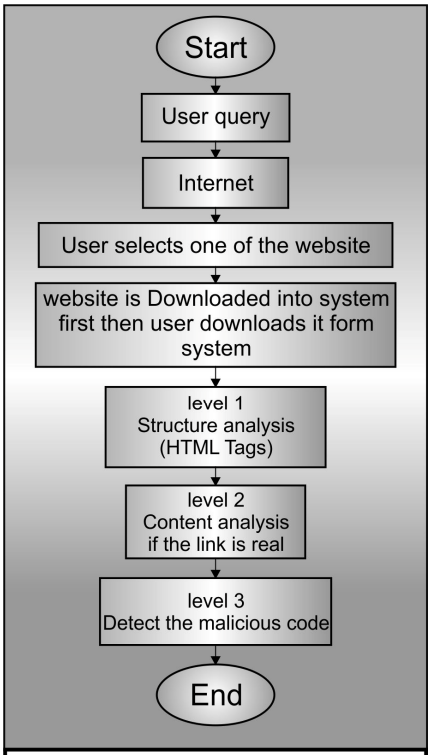


Figure -1 System Overview

Each level has some tasks to do, and when it completes its tasks, the next level will be continuing the work until the total levels of the

system is complete. Each of tasks acts as part of analysis and test cycle to provide the goal of this research.

These levels are: -

1- Structure analysis.
2- Content analysis.
3- Detect malicious code.

First level is open HTML files (source code) and calls the database, searches in these HTML files for all tags (links) involved in webpage, this analysis aim to checks hyperlinks in the webpage and matching between this tags and malicious code in library if it is exist in it.{5}

Second level is content analysis by check. If links are real or not, they are not real when they are not connect to other web page that it's not important but they are real if they are connect to other web page important to check content for any suspicions code {7}.

Third level detects malicious code by using database content, which contains many kinds of malicious codes, matches between tags (links) and kinds of malicious code. It displays the file name and type of malicious code in the area of check result {10},{11}.

3. The proposed system Description

The objective of using model is to document what functions the system should offer to the user. System representation for hyperlink security analysis for website and detection of the Figure (2) illustrate the malicious code.

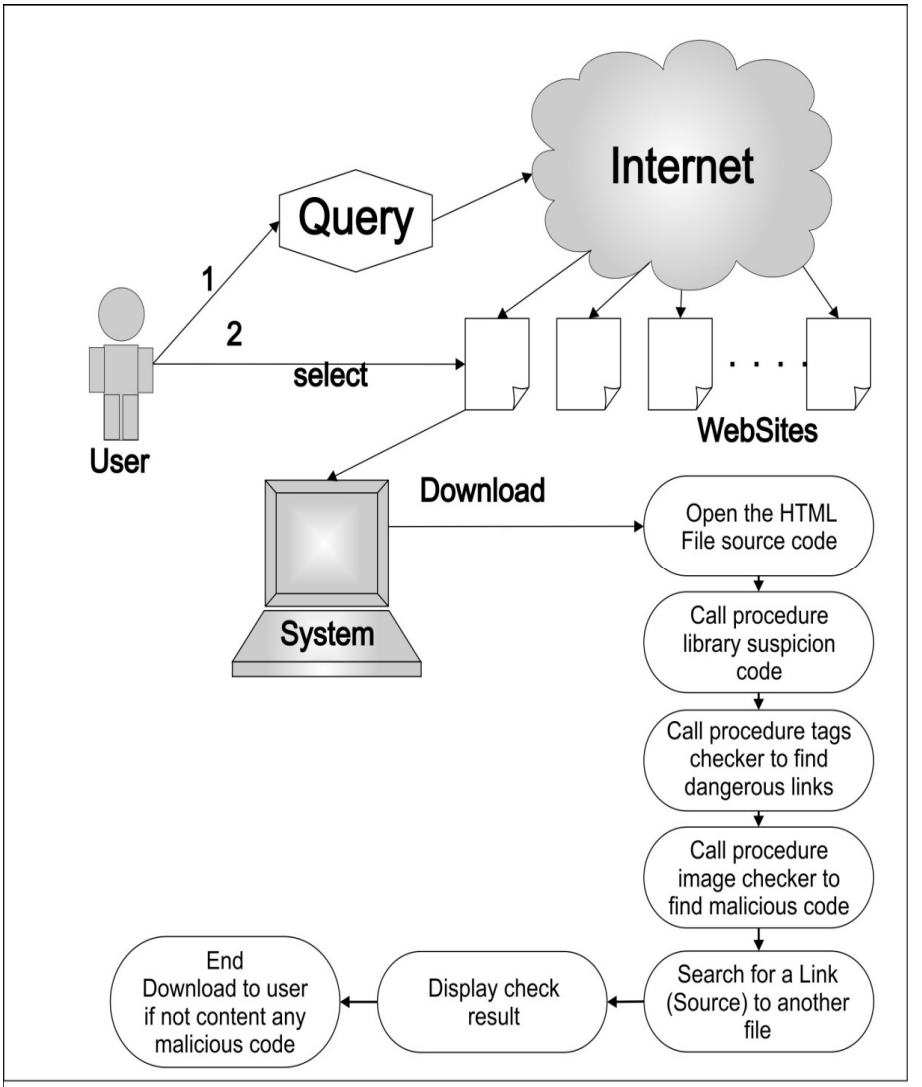


Figure-2 proposed system Description

In Figure (2), initially HTML file may be downloaded through the internet, the procedure is to open the HTML file and then call the main checker to do the main processing.

The main checker consists of three procedures that are called sequentially to check the following: -

1- Procedure suspicious library calls the suspicious library checker to check the main suspicious file. There is a flexible database, which contains the name of the known suspicious calls {6}.

2- Procedure tag checker calls the tag checker to find the dangerous tags. This procedure searches for dangerous tags (links) that can call and execute any malicious code {7}.

3- Procedure image checker calls image checker to find if in the image pixel malicious code could be found because the images are very important hyperlink to another page or site. This type of images seems unlike images, thus the purpose of existence of these images is suspicious {2}.

After applying all the rules of the three procedures, a new check is done on the HTML file, this new processing searches in the HTML files for a link (source) to another file or document, if any link exists, the same procedures will apply to the linked file, otherwise the system will show the check results of the HTML file. After that the system is to open a new HTML file and repeat the same processing described above, when all the HTML files are checked and results are obtained the system finishes the current work, waiting for checking another file or it exits {3},{8}.

4- System Components

The system essentially consists of five main components as shown in figure (3)

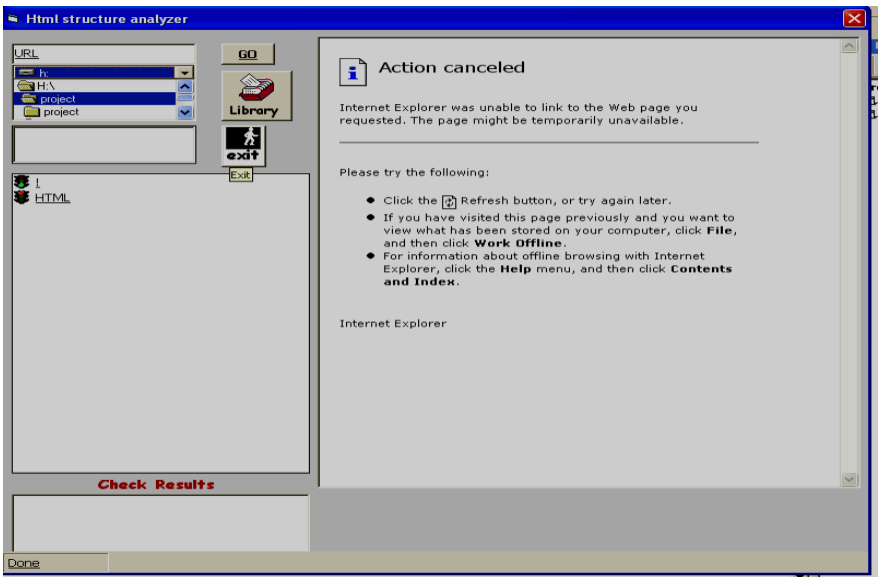


Figure (3) The Program Interface Before any File Check

1- URL Link: This option is to choose a URL name to be navigated. In this process take the name of the URL and call the main checker. Enter the name of website after that click (GO) button. It displays the website and system being opens webpage through the Internet to check all tags in existing files that belong to the same family as HTML this selectable choice is **on line** to internet.

2- Local File Selection: This option allows us to choose one of the existing HTML files that we want to test and check. If we choose to check the HTML files, the system will call the main Checker to do its work on the selected choice. Open the HTML files, after the system checks the file and its link, if it exists, the system will display the check result to the user in case the HTML file contains malicious codes, a warning will appear to tell the user that this file includes a threat to computer system.

In addition, the system gives information about the type of threats that may be contained in an HTML document. Finally the system views the page via customized Internet Explorer as user desires.

3- Library Updating: This selection is to let the user update the file of the suspicious library calls list by calling the updating form to show or update the suspicious files this library can update add or delete malicious code.

4- Exit Selection: To exit from the system the user must click on the Exit Select. We can choose this select when we do not want to check the existing files, or in the case there are no files to be checked.

5- System Development

There are three checkers are follows

5.1 Suspicious Library Checker

The following flowchart in Figure (4) shows library checkers that applied to the HTML files represents the suspicious library checker that checks the entire file if it contains a calling process to one or more of the suspicious files that they have been running some malicious codes which infect the system hard registry, HTML files, or resources.

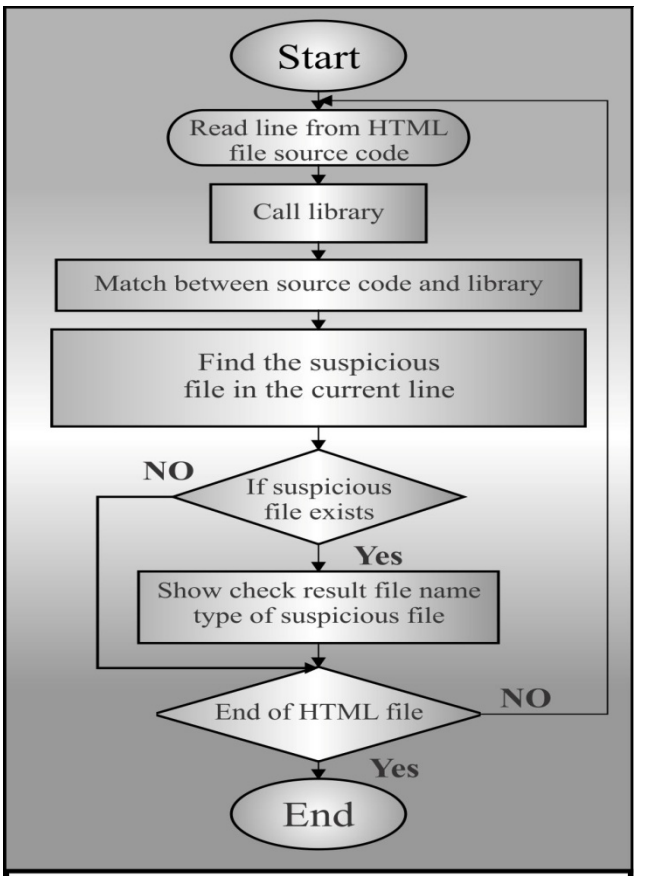


Figure (4) Suspicious Library Calls Checker Flowchart

This checker begins by searching for the suspicious files in the current line of the HTML codes and matches them with the suspicious library calls. If the files exist in the suspicious library calls list, the system will show a message which contains the HTML file name, warning, and suspicious file name.

In this case or in the case the suspicious file is not found in the HTML file, the system repeats the process until the end of the suspicious library calls list.

Input: Open library file, Open HTML file
Output: check result (file name and the type of malicious code)

Repeat
Step1: Call procedure library checker (read library name)
Step2: Read line from HTML file
Step3: Find the existence of the HTML line, matching between
the library content of kinds Malicious code and tags
Step4: show check result, If malicious code found in the current line
Step5: Loop for next line in HTML file
Step6: Until (EOSL)/*End of suspicious library calls list*/

Figure (5) Proposed Algorithm of Suspicious Library Calls Checker

in Figure (5) there is a Proposed representation of the suspicious library calls checker algorithm. This library is the database content several kinds of malicious code can be modified or update (add or delete) malicious code.

5.2 The Tag Checker

As we know HTML language consists of several elements called tags. These tags are considered the basic element that the HTML document stands for.

Some of these tags are unsecured elements, which call and run some unwanted malicious files or codes.

In this checker procedure we search for this dangerous tags and test their attributes and values to decide if these tags are considered secure or no.

In the following flowchart there is a general description of the tag checker processing which is represented in Figure (6).

With the current opened HTML file the tag checker takes its order in the system work. And in table (1) shows types of tags involved, the HTML file may contain malicious code or viruses.

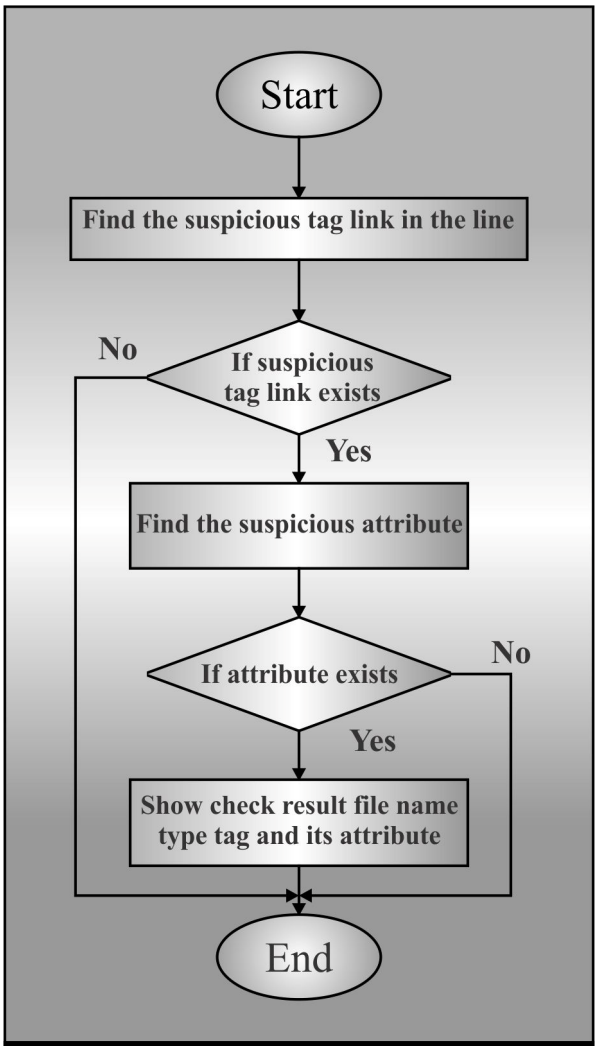


Figure (6) Suspicious Tag Checker Flowchart

The processing of this procedure begins with search for the suspicious tags (links), which are described in table (1).

If the tag in the current line exists in the suspicious tags list, the next step will find the suspicious attribute of this tag.

Now after we find the suspicious tag, the important part of this tag is its attribute and the value. The value of the attribute will decide the legality of the tag. For example, the "link" tag is considered a suspicious tag if its "Scr" attributed takes the value of yes i.e <link SRC = "filename.htm" Application = "yes" Another example, the "<A" tag also is considered suspicious tag with its attribute "HREF", i.e. <A HREF = "something">. When all the tags (links) of Hypertext file are checked, the system will show a message, which contains the following information:

- ♦ The HTML file name.
- ♦ A warning that this file includes one or more of suspicious tags.
- ♦ The suspicious tags and their attributes.

Table (1) The Suspicious Tag Link and Attributes of HTML Document

Tag	Attributes
<A	HREF=
<Link	HREF=
<IMG	SRC=

Figure (7) describes an algorithm of the suspicious tags checker, and works under the order of the algorithm.

Input: HTML file
Output: Check result (file name and the type of Malicious Code)
Repeat
Step1: Find the existence of first tag in the line of HTML File
Step2: If it exists then begin
Step3: Find the existence of the suspicious attribute
Step4: if the attribute link exists (HREF) then add the link to link list and write a warring message for the tag and attribute
Step5: Show check result (file name, type of malicious code)
Step6: Loop for next line HTML
Step7: Loop for next tag (links) in the tag list
Step8: loop for Additional links, if it exists in the link list
Until (EOSL)

Figure (7) Algorithm of Suspicious Tag Checker

5.3 The Image Checker

As shown in Figure (8) below, the image checker is starting to do it work at this time. After the suspicious library calls checker and suspicious tag checker finish their work, the image checker begins to do its own checking property.

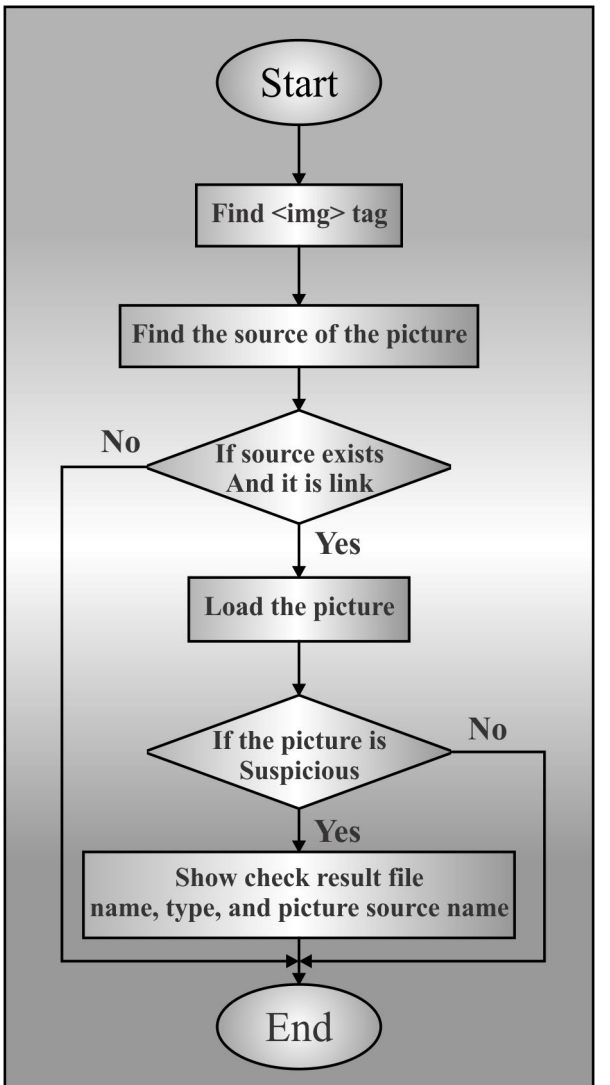


Figure (8) Suspicious Image Checker Flowchart

The last procedure is working to find the <IMG> tag; this tag as described in table (1), represents the HTML file element, which can incorporate an image in the Hypertext document.

If the procedure finds the <IMG> tag, it will find the source of the picture.

If the source of picture is link to other web page, the system must load the

Picture, otherwise it returns to the main checker.

After loading the picture, the system will check the image; this type of image may refer to unsafe link using this image as a link to cause some malicious or dirty work on the user system. Then the system gives its output message, which contains the following information.

- The HTML file name.
- A warning that a suspicious image is found in the current file.
- Source name of the image.

- Figure (9) describes an algorithm of the special image checker, which causes a dangerous link to a malicious file. And works under the same order of the algorithm.

Input: HTML File
Output: Check result (file name and the type of Malicious Code)
Step1: Search for image tag in the current HTML line
Step2: Find the source of the picture
Step3: if exists then find the source attribute for the tag
Else it exit
Step4: Load the picture
Step5: If image link content malicious code then
Show check result

Figure (9) Proposed Algorithm of Suspicious Image Checker

- Now the main checker, which it calls all the previous checkers,

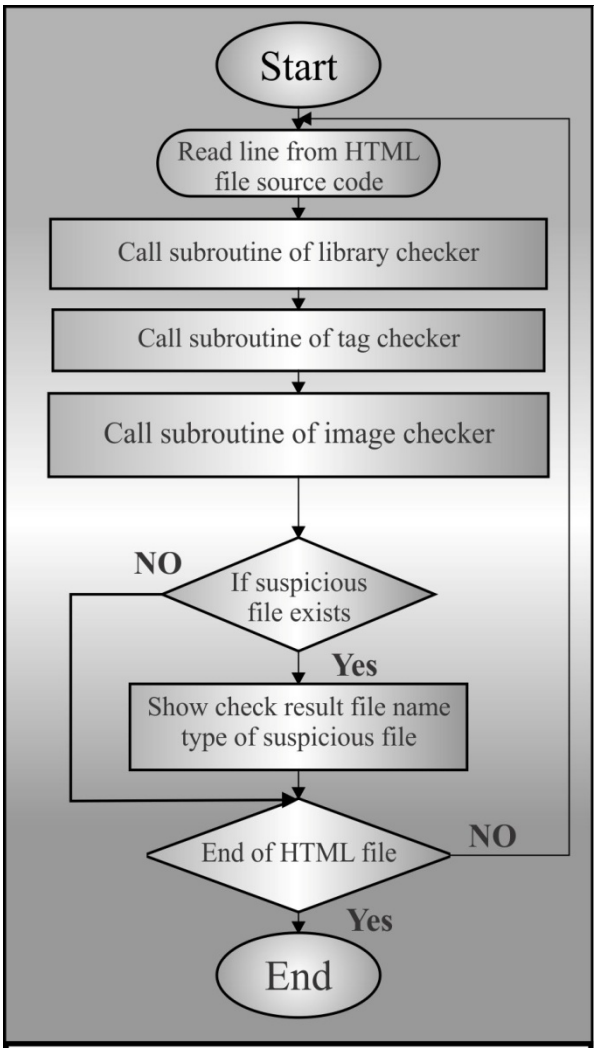


Figure (10) Main Checker Flowchart

Figure (10) to check the current file and all sources (link) find in it. This figure represents the general algorithm of the main checker which operates the checker procedures and then gives the system result as an output message.

There are six steps below explain the main checker

1- Open HTML file.
2- Call subroutine of library checker.
3- Call subroutine of tag checker.
4- Call subroutine of image checker.
5- Search to find source code of each line in HTML file.
6- If it exists then return to subroutine to check, return to read another line and if all lines end then return to read another page.

6. Test Results

In this section make test results for the Analysis and detection system from malicious Hypertext Languages.

- **Online Testing (URL)**

  By connect (on line) in the Internet entering URL and press Go, in the Figure(11) show this test.

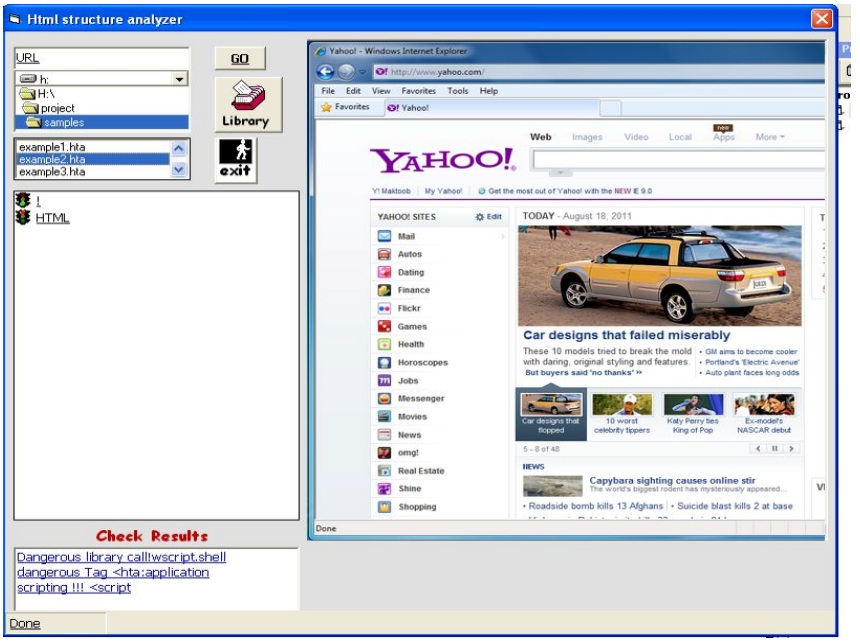
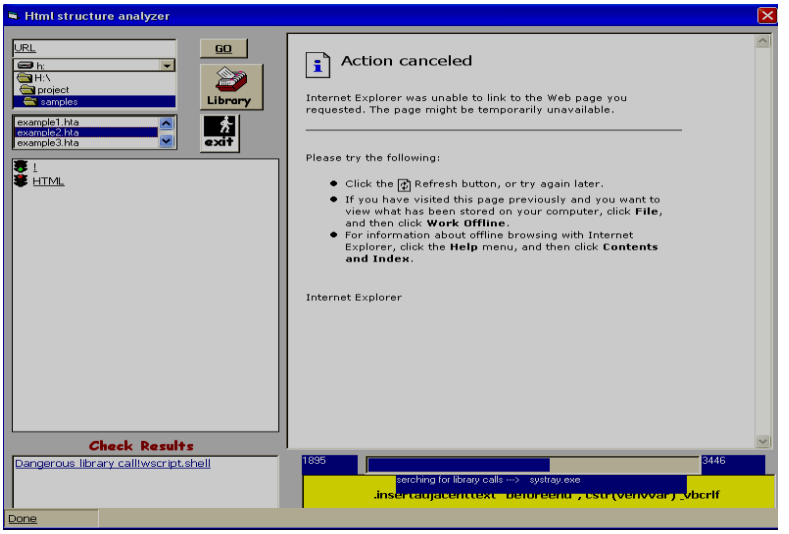


Figure (11) Online the Internet by URL

- **Offline Testing (HTML File Examples)**

Here we check an HTML file (example), .In the following Figures show all the stages of the system checkers that display a sequence output interfaces with a final result.

The Program Interface Checking File Example as shown in figure(12).



Figure(12)

Result of Checking example with Suspicious Library Calls and Tag Checkers as shown in figure (13).

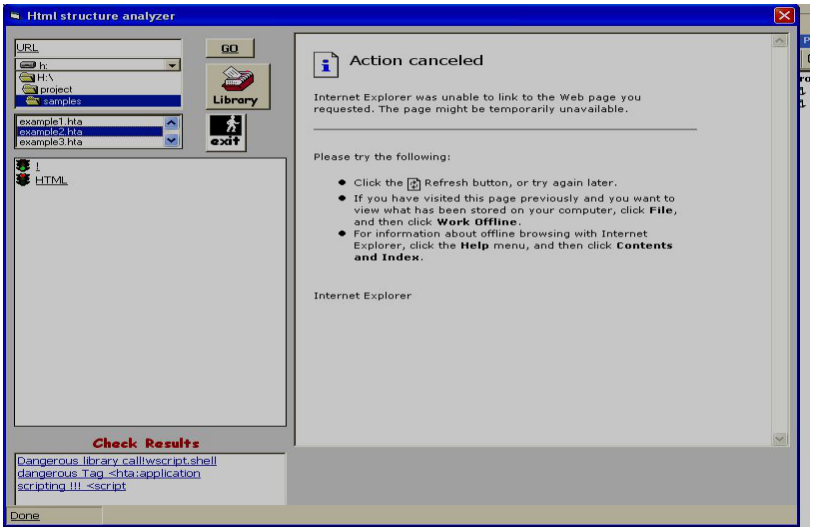


figure (13)

Figure (14) Shows Suspicious Library Calls to Add New Malicious in Library

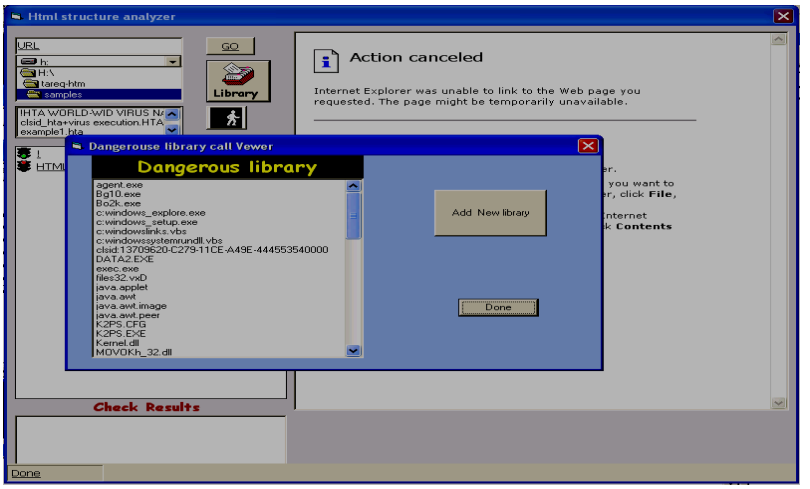


Figure (14)

Figure (15) shows the end result of checking the HTML file (example) which contains suspicious tags that do something malicious. The suspicious tags display the output interface of the program.

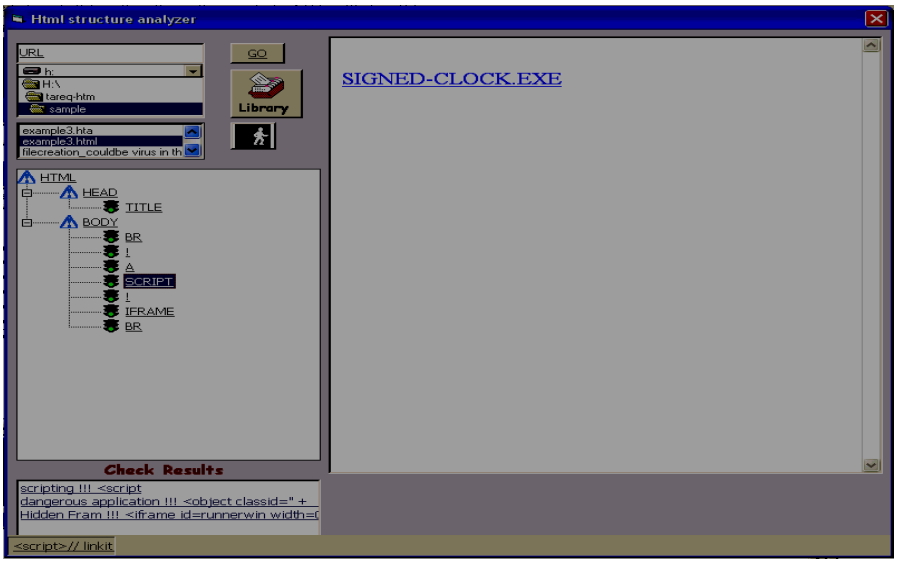


Figure (15)

Figure (16) shows the end result of checking the HTML file (example) which contains a suspicious image and link to another source to the output interface. There is a warning that an image exists in the current file.

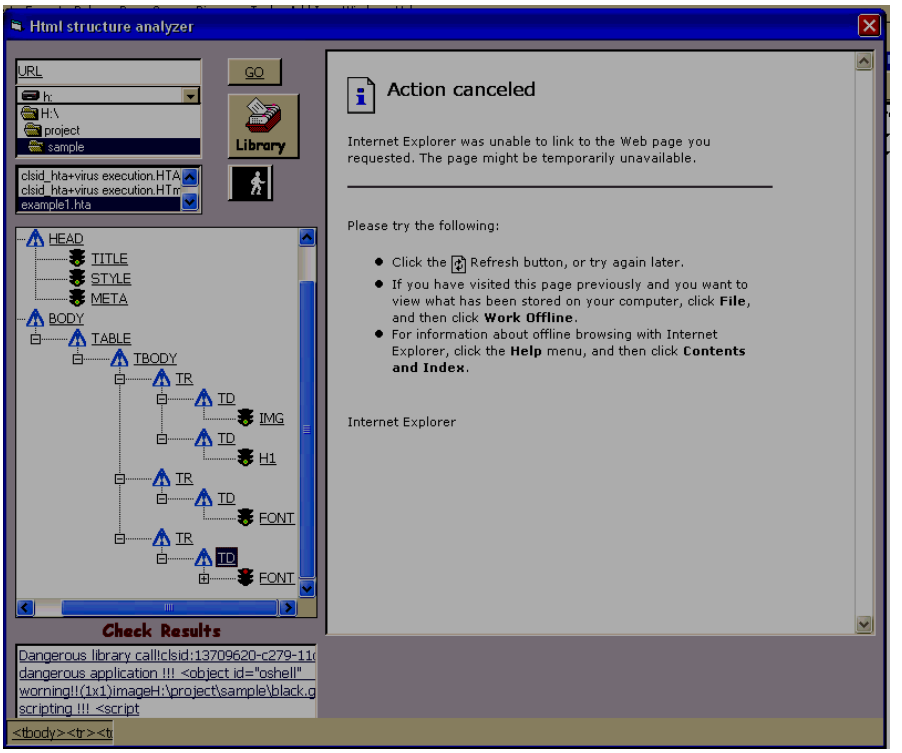


Figure (16) End Result of Checking the HTML File (Example) which Contains a Suspicious Image

7.1 Conclusions

Through this use of the proposed system, we came out with the following conclusions:-

1- A good analysis of the website structure is very important step to discover malicious codes.
2- As long as the Internet websites are developed and grown malicious code problems grow with it and become more difficult to detect it.
3- The probability of a malicious code into a website is greater if that website contains many links to other web pages.
4- Building an updateable database is important to include all the variety of malicious codes and downloading updates for database.
5- Most of the suspicious files in malicious codes and viruses in tag HTML and pictures used as link to other web page is hiding, and are used to host the malicious code.

7.2 Suggestion for Future Works

1. Making the (Malicious Code Database) automatically updateable from Internet by new malicious code signatures.
2. Building a miniature model of the system as a web crawler and sending it over the Internet to clean it up.

3. Running several copies over the client/server network each one of which at the client's computer to increase speed over a network.
4. Adding a security check entry into the database, updateable only by the system to prevent other malicious code from manipulating the database.

References

1- Merike Kaeo,"***Designing Network Security***", Second Edition, Macmillan India Ltd.2004

2- Mark C. John," ***Searching the World Wide Web***", 1998

http://www.neci.nec.com/~lawrence/papers/search-ic98/search-ic98.pdf

3- Terry Escamilla,"***Intrusion Detection***", John Wiley& Sons, Inc, 1998

4- John E. Howland, "***Introduction to Internet Security***", Department of Computer
Science, MS.C. Trinity University, 2002

5- Hasnien S. Al-Ani, "***protecting Systems from Malicious Hypertext Language***",
MS.C., University of Technology,

6- 2003 Zoran Nikoloski , Narsingh Deo, and Ludek Kucera, "***Correlation Model***
for Worm Propagation on Scale-free Graphs", MS.C., Charles university,

2005.

7- Sura Mazin,"***Detection of HTML Viruses Using a Monitoring System***", MS.C. ,
Iraqi Commission for Computers and Formatics/ Informatics Institute for Postgraduate Studies, 2006

8- TrendMicro, "**Security *Testing, Firewall, and Checkers***", Proc., 2002
www.carnet.hr/cuc/cuc2000/radovi/prezentacije/trendmicro.com,

9- K.Smith,"***Securing an Internet Name Server***",2000
http://www.praxis.bond.edu.au/prism/papers/refereed/paper5.pdf

10- CERT advisory, "***malicious code and application***", O'reilly &Associates Inc. 2000.

11- Roger A. Grimes, "***Malicious Mobile Code***", O'reilly &Associates Inc., 2001.

كشف البرامج الخبيثة في مواقع الانترنت

م.م.هشام رعد جعفر*

المستخلص

كما نعلم بان الانترنيت اصبح شبكة واسعة الاستخدام ، تحتوي على انواع عديدة من الاوساط المتعددة كالوثائق و البرامج و المواقع ، لهذا السبب انه من الاصعب ادامتها كبيئة خالية من التخريب ، هذه البرامج الخبيثة تكون مخفية داخل مواقع ذات مظهر برئ ، هذه المواقع بدورها واسعة الانتشار ، وهي صفة مهمة للبرامج الخبيثة كي تعتمد عليها للوصول الى اكبر عدد من المستخدمين.

النظام المقترح مبني بصورة عامة لايجاد وكشف هذه البرامج الخبيثة في مواقع الانترنيت ، ولان هذه البرامج والمبرمجين المخربين منتشرون بكثرة فان الحاجة لهذه الاداة اصبحت حقيقة واضحة، الموقع المستهدف يتم تحليله لكشف وجود هذه البرامجيات الخبيثة فاذا لم يجد هذه البرامجيات فيه فيتم تتبع الوصلات بصورة مفردة لفحص هل البرامج موجودة بصورة غير مباشرة، هذه الطريقة مفيدة في المواقع ذات الانتشار السريع.

يتم تشغيل النظام على الخادم وعندما يقوم المستخدم بطلب تحميل صفحة فهذا الطلب يتم ارساله اولا الى الخادم ثم يتبع هذا المسار ويتفحص المحتويات لذلك الموقع متتبعا كل الوصلات الثانوية فاذا كان الموقع نظيف (اي خالي من البرامج الخبيثة) فان الخادم يسمح باستخدام ذلك الموقع والا يكشفها ويكشف نوعها ويهيئ تقرير للمستخدم عن الاصابات ونوع الاصابة كرسالة تحذيرية في الموقع الخاص بالفحص.

هذا النظام تم بناءه باستخدام نظام فيجوال بيسك .

* وزارة التعليم العالي والبحث العلمي- دائرة البعثات والعلاقات الثقافية