Image Tracker

Prepared By

Khyati Thakkar

12MCEI30

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382481

May 2014

Image Tracker

Major Project

Submitted in partial fulfillment of the requirements

For the degree of

Master of Technology in Computer Science and Engineering

(Information and Network Security)

Prepared By

Khyati Thakkar

12MCEI30

Guided By

Ms. Rupal Kapdi (Internal Guide)

&

Mr. Nilay Mistry (External Guide)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382481

May 2014

Certificate

This is to certify that the major project report entitled **"Image Tracker"** submitted by Khyati Thakkar (12MCEI30), towards the partial fulfillment of the requirements for the degree of **Master of Technology in Computer Science and Engineering (Information and Network Security)** of Nirma University, Ahmedabad is the record of work carried out by her under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project, to the best of my knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

Ms. Rupal Kapdi
Internal Guide & Asst. Professor,
Dept. of Computer Science & Engg.,
Institute of Technology,
Nirma University,
Ahmedabad

Mr. Nilay Mistry
External Guide & Asst. Professor,
Dept. of Digital Forensics,
Institute of Forensics Science,
Gujarat Forensics Sciences University,
Gandhinagar

Prof. Sharada Valiveti
Assoc. Professor & M.Tech. INS Coordinator,
Dept. of Computer Science & Engg.,
Institute of Technology,
Nirma University, Ahmedabad

Dr. Sanjay Garg
Professor & Head,
Dept. of Computer Science & Engg.,
Institute of Technology,
Nirma University,Ahmedabad

Dr. K. Kotecha,
Director, Institute of Technology, Nirma University, Ahmedabad

Undertaking for Originality of the Work

I, **Khyati Thakkar (12MCEI30)**, give undertaking that the major project entitled **"Image Tracker"** submitted by me, towards the partial fulfillment of the requirements for the degree of **Master of Technology in Computer Science and Engineering (Information and Network Security)** of Nirma University, Ahmedabad, is the original work carried out by me and I give assurance that no attempt of plagiarism has been made. I understand that in the event of any similarity found subsequently with any published work or any dissertation work elsewhere; it will result in severe disciplinary action.

Khyati Thakkar (12MCEI30)
Date:
Place:

Endorsed by

Ms. Rupal Kapdi
Internal Guide

Mr. Nilay Mistry
External Guide

Acknowledgements

It gives me immense pleasure in expressing thanks and profound gratitude to **Ms. Rupal Kapdi**, Internal Guide & Assistant Professor, Dept. of Computer Science & Engg., Institute of Technology, Nirma University, Ahmedabad and **Mr. Nilay Mistry**, External Guide & Assistant Professor, Dept. of Digital Forensics, Institute of Forensics Science, Gujarat Forensics Sciences University, Gandhinagar and **Mr. Abdul Zummerwala**, Project Scientist, BISAG, Gandhinagar for their valuable guidance and continual encouragement throughout this work. The appreciation and continual support they have imparted has been a great motivation to me in reaching a higher goal. Their guidance has triggered and nourished my intellectual maturity that I will benefit from, for a long time to come.

My deepest thank you is extended to **Prof. Sharada Valiveti**, Coordinator M.Tech. - INS, Dept. of Computer Science & Engg., Institute of Technology, Nirma University, Ahmedabad for an exceptional support and continual encouragement throughout the Major Project.

It gives me an immense pleasure to thank **Dr. Sanjay Garg**, Hon'ble Head of Dept. of Computer Science & Engg., Institute of Technology, Nirma University, Ahmedabad, **Dr. K. Kotecha**, Hon'ble Director, Institute of Technology, Nirma University, Ahmedabad and **Dr. M S Dahiya**, Hon'ble Director, Institute of Forensics Science, Gujarat Forensics Sciences University, Gandhinagar for their kind support and providing basic infrastructure and healthy research environment.

I would also thank **Ms. Twinkle Patel (12MCEI38)**, **Mr. Madhur Tewani (12MCEI29)**, my other colleague friends, the institution and all faculty members of Dept. of Computer Science & Engg., Institute of Technology, Nirma University, Ahmedabad for their special attention and suggestions towards the project work.

- Khyati Thakkar

12MCEI30

Abstract

The Internet is a global network connecting millions of computers. So anyone from any corner of this world can upload or download anything to and from the Internet. Moreover, social networking is being widely used by people to share each and every update of their life. So in this era of communication, lots of cyber crimes are taking place and they are increasing day-by-day and one of them is, social media crime. Image Tracker will help reducing such kind of crimes as it is the functionality which embeds the source computer's unique identifier i.e. MAC address into the image by modifying exif data. So, if any such crime takes place by editing and uploading someone's image then the computer, on which the image has been lastly modified and has been uploded, can be traced down by fetching unique identifier of that computer from the image.

Key words: Image Tracker, Morphing Detection, MAC address within Image, Track Source of Image

Contents

List of Figures

Chapter 1

Introduction

The Internet is a global network connecting millions of computers. Due to the increasing amount of data available online, the World Wide Web has become one of the most valuable resources for information retrievals and knowledge discoveries. So, anyone from any corner of this world can upload or download anything to and from the Internet. Moreover, social networking is being widely used by people to share each and every update of their life. It has been playing a major role in people's lives. But there also exists a community which misuses such information for financial gains or else for any personal benefits. This is a community which never shows up anywhere in the physical world but plays a significant role on the Internet and they are hackers. In this era of communication, lots of cyber crimes are being performed by them and they are increasing day-by-day and one of them is, social media crime.

Significant number of cases have been filed in which some person's images have been modified in some bad manner and then they have been uploaded on the world wide web on some social networking websites and the original culprit haven't been found for long time as there is no technique available which can directly trace down the computer from which the image has been modified and uploaded on the web. The maximum possible detail is the ip address of the ISP (Internet Service Provider) to which the computer was connected. ISP uses DHCP (Dynamic Host Configuration Protocol) to allocate ip addresses to hosts but that ip address changes everytime the host disconnects and reconnects to the ISP. The ip addresses and the time when they were used by which computer are stored in the form of log files on ISP but ISP maintains these log files for specific amount of time only, due to limited storage capabilities. After that, the log files may get destroyed. So, it is very difficult to know which ip address was used by which computer at what time and thus, the computer used by culprit cannot be traced so easily.

1.1 Problem Definition

Image Tracker helps reducing such kind of crimes as it is the functionality which embeds the source computer's (on which the image has been lastly modified) unique identifier i.e. MAC address into the image by modifying the image's exif data. So, if any such crime takes place by editing and uploading someone's image then the computer, from which it has been uploaded, can be traced down by fetching unique identifier of that computer from the image.

1.2 Chapter Organization

This section gives brief information of all the chapters. Chapter 1 gives the introduction about the topic and the report. Chapter 2 gives the details about why this project was necessary to be carried out. Chapter 3 contains the literature survey of GIMP basics, the existing methodologies of GIMP plug-in development and exif tags. Chapter 4 contains various available techniques of backtracking to the source of the image. Chapter 5 gives details about challenges found in existing techniques and the proposal of this project. Chapter 6 explains the methodology of implementation for the plug-in that is to be developed. Chapter 7 shows the results of the plug-in implemented for embedding MAC address into the image. Chapter 8 concludes with current status and the future work. Chapter gives details of research papers published.

Chapter 2

Motivation

There are many image tracking services online like TinEye, PicScout, PhotoSecrets, ePHOTOzine etc. But they only provide the URL, by backtracking, from where the image has been used and not the exact source from where it has been uploaded. Some of them are purely commercial websites which shares and tracks images for financial gains. The objective of this work is to address the issues and propose solutions as compared to the present works for similar kinds of problems. This is due to the motivation behind the work to make the proposed technique innovative, simple and less computational.

2.1 Domain Study

This project is about working on images and there are lots of websites as well as lots of tools which work on images. To narrow down, first thing is to select a proper tool for image editing and that tool should be open source so as to make changes to it. The popular tools for image modification are:

1. Adobe Photoshop
2. Picasa
3. Paint
4. Paint.net
5. Microsoft Office Picture Manager

6. GIMP

But none of them, except GIMP, are open source. Picasa and Paint were open source in earlier times but they are not anymore due to some security reasons. So, the most desired tool which is open source as well as popular is GIMP.

Chapter 3

Literature Survey

GIMP (GNU Image Manipulation Program) is an image retouching and editing tool and is released as free and open-source software.[17] There are versions available for most operating systems including Linux and Microsoft Windows. GIMP has tools used for image retouching and editing, free-form drawing, resizing, cropping, converting between different image formats etc. Animated images such as GIF and MPEG files can be created using an animation plug-in.

To build a new plug-in for GIMP, we need to first study its source code and the methods for developing plug-in. The GIMP source code lives in the gimp repository on the GNOME git server. The GNOME git server hosts a couple of GIMP related repositories:

Module	Description
babl	Pixel format conversion library
gegl	Generic Graphical Library
gimp	GIMP and the standard set of plug-ins
gimp-data-extras	GIMP Data files such as brushes, gradients, patterns and the like
gimp-gap	GIMP Animation Package, a set of plug-ins that provide video editing functionality
gimp-help-2	GIMP User Manual
gimp-perl	GIMP Perl bindings and a bunch of nice gimp-perl scripts
gimp-plugin-template	GIMP Plug-In Template, a starting ground for plug-in developers
gimp-plugins-unstable	GIMP plug-ins from the past, a collection of unstable and unmaintained plug-ins
gimp-ruby	GIMP Ruby-based scripting plug-in
gimp-tiny-fu	GIMP Tiny-Fu, a drop-in replacement for Script-Fu
gimp-web	the GIMP web site, available at www.gimp.org
gimp-web-devel	the source of the pages you are reading right now

Figure 3.1: GIMP Repositories

[1]

From all of these, GIMP source resides in the GIMP folder. In that, the code of our concern resides at following paths:

- gimp-2.8.6 \app \plug-in
- gimp-2.8.6 \libgimp
- gimp-2.8.6 \plug-ins

3.1 GIMP Architecture

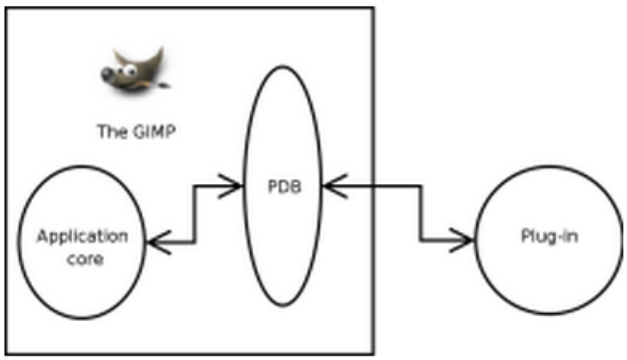

Figure 3.2: GIMP Architecture

[2]

Here are the steps of how it works:

1. The GIMP script interface is centered on the Procedural database (PDB). [2]
2. At startup, The GIMP looks into a predefined set of places for scripts and plug-ins, and asks each new script to identify itself. [2]
3. The plug-in declares itself to the PDB at that time, and passes informations like the position it wishes to get in the menu hierarchy, input parameters, and output parameters. [2]
4. When a script or a plug-in wants to use our plug-in, it gets through the PDB, which manages communicating parameters in one direction and the other in a transparent way. [2]
5. Internal functions that wish to get exposed to plug-ins have to be packaged first in the core, that will register them in the PDB, and secondly in the libgimp that will allow the function to be called as a normal one. [2]

3.2 GIMP Plug-in Development

There are two methods to develop a GIMP plug-in:

1. Develop a C plug-in using a library called libgimp and an associated utility named gimptool.

2. Develop a Python plug-in using the Python-Fu interactive console which is in-built in GIMP.

3.3 Exif Tags

EXIF stands for Exchangeable Image File Format, and is a standard for storing interchange information in image files, especially those using JPEG compression.[15] Most digital cameras now use the EXIF format.

The Japan Electronic Industries Development Association (JEIDA) produced the initial definition of Exif.[14] The metadata tags defined in the Exif standard cover a broad spectrum:

1. **Date and time information.** Digital cameras will record the current date and time and save this in the metadata.[14]

2. **Camera settings.** This includes static information such as the camera model and make, and information that varies with each image such as orientation, aperture, shutter speed, focal length, metering mode, and ISO speed information.[14]

3. **A thumbnail,** for previewing the picture on the camera's LCD screen, in file managers, or in photo manipulation software.[14]

4. **Descriptions.**

5. **Copyright information.**

For editing exif data of an image using Python in GIMP, Pexif library is compulsory. Pexif is a Python library for parsing and more importantly editing EXIF data in JPEG files. The library contains some examples as well as some usable standalone PexifScripts.[12] Pexif can be downloaded from $http://code.google.com/p/pexif/downloads/list$. The latest version is pexif-0.13.

There is one more external library exiftool, which can be used as an external library as currently GIMP has no support for manipulating EXIF data so we will use exiftool as our external library. Exiftool can be downloaded from *http* : *//www.sno.phy.queensu.ca/ phil/exiftool/*.

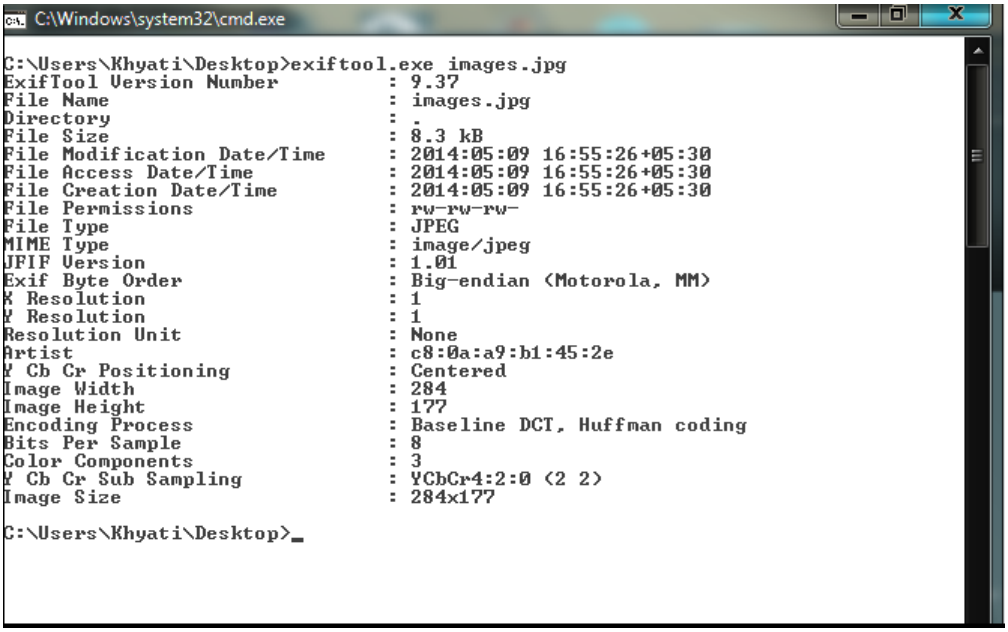

Figure 3.3: Snapshot of Exif data of an Image

[14]

Figure 3.3 is the snapshot captured while checking the exif data of some random image using exiftool.

Chapter 4

Study of Available Techniques

4.1 TinEye

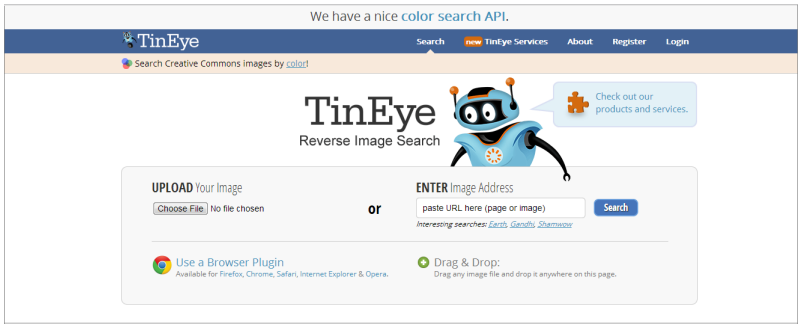

Figure 4.1: TinEye GUI

TinEye is the online service which helps searching an image but it has several limitations:

1. **Limited Index Size:** Currently, the Tineye database is at about 700 million images. While that is an impressive number, one has to remember that Photobucket alone has over 5 billion images according to their numbers. The site does not seem to detect duplications on Photobucket, Flickr or other popular image sharing sites, focusing instead on blogs. Thus, many images that are known to have many copies return no results.[3]

2. **No Case Tracking:** Currently, with Tineye, there is no way to track cases of plagiarism or copying so that they are not acted upon a second time.[3]

3. **No Alerts System:** Where writers have Google Alerts and even CopyAlerts, there is currently no system in Tineye that will alert artists to new copies of their work being posted.[3]

4.2 PicScout

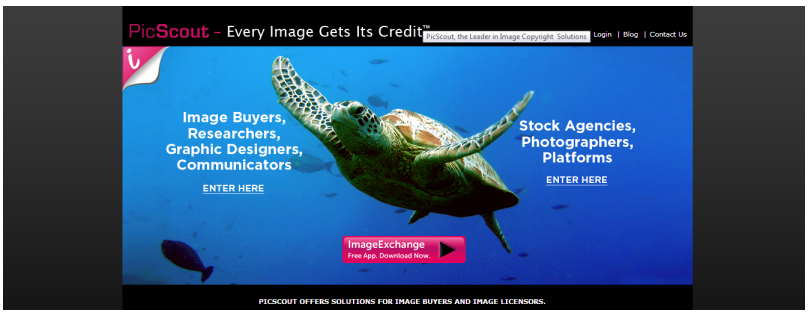

Figure 4.2: PicScout GUI

PicScout also has some limitations, which are as follows:

1. Uses ImageExchange and ImageTracker applications so it is a dependent application.
2. Only monitors business usages.
3. Only covers North America, UK and Germany and not India.

4.3 PhotoSecrets

Figure 4.3: PhotoSecrets GUI

PhotoSecrets is also an online image exchange service but it consists of two important features:

- Copyright your photos
- Sell copyrighted photos

Thus, it's purely a commercial website.

Chapter 5

Proposal

5.1 Challenges

There are several challenges found in the existing techniques, such as:

1. The problem with online image tracking service is, if you are looking for an image which is rarely available on the Internet or else not at all available then it is almost impossible to track it back.
2. There are very few image editing tools which are open source.
3. There are no tools available which gives any information about the origin of the image.

5.2 Proposed Solution

For developing a functionality which helps identifying the source of image, the unique identifiers of the computer need to be fetched successfully. Moreover, the process of narrowing down on a single tool which is open source as well as popular is also important. And after selection of tool, electing a proper method to develop the technique also matters a lot. The technique is developed for almost all image formats for e.g. .tiff, .jpg, .png etc. The objective of the proposed technique is to develop a cost-effective alternative for commercially available services and tools.

Following is the step wise proposal for the project:

1. Narrowing down on a single tool taking into consideration both the open source and most widely used features.

2. Selecting a proper method to develop plug-in for selected tool.

3. Checking whether the method can fetch unique identifier of computer.

4. If yes, then developing final code.

Chapter 6

Implementation Details

6.1 Implementation Methodology

GIMP is the most widely used open source tool for image editing and it has in-built interactive Python-Fu console so developing a Pyhton plug-in for GIMP would be more convenient and proper.

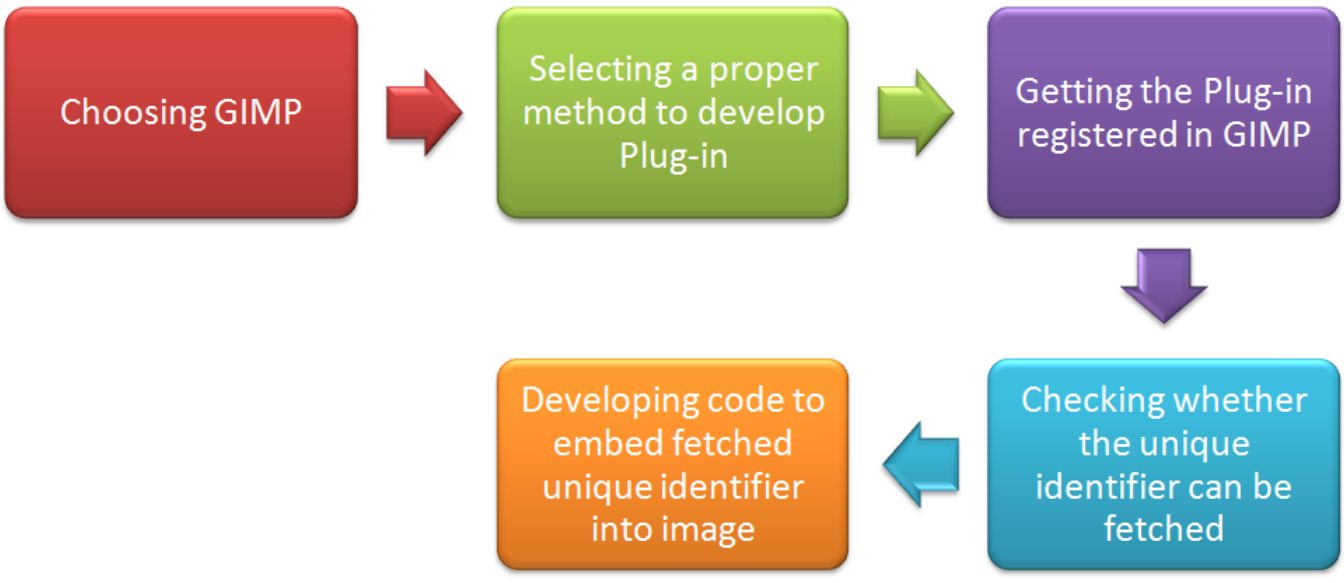

Figure 6.1: Implementation Methodology

So, after getting the Python plug-in successfully registered in GIMP, we have to verify whether it can fetch the MAC address of the computer.

If so, we would start developing code for embedding fetched MAC address into image by manipulating exif data of the image.

Chapter 7

Results

1. The first step is getting our plug-in registered in GIMP. Following are the steps shown for it:

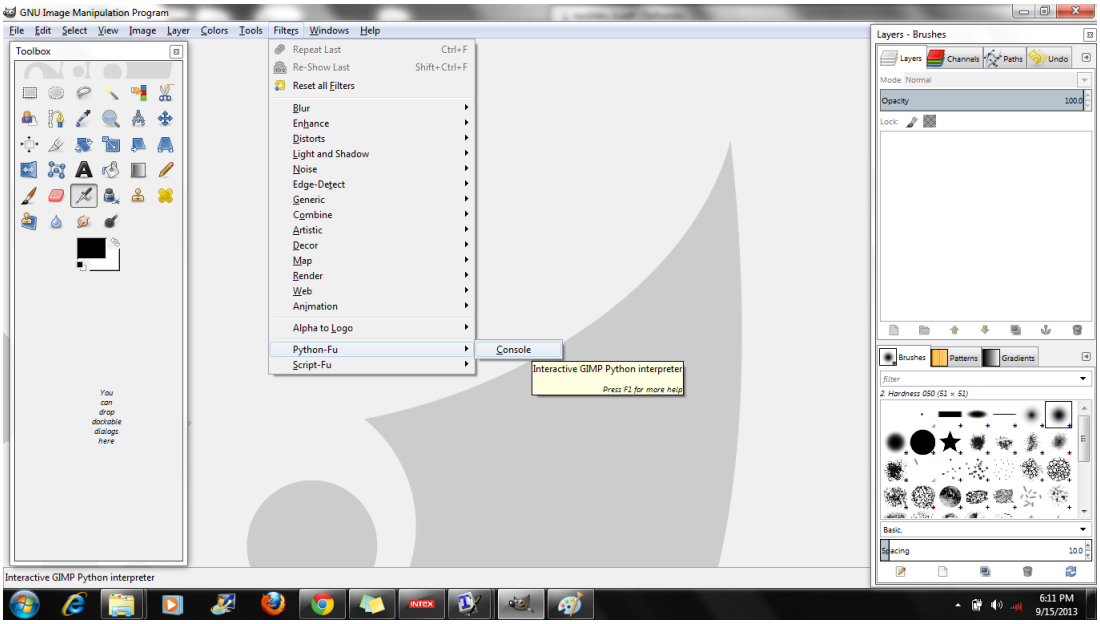

Figure 7.1: How to go to Python-Fu Console

Go to $Filters->Python-Fu->Console$ to get Python interactive console in GIMP.

The command shown in Figure 7.2 is to test whether Python console responds properly.

This is a sample code to get the new plug-in registered in GIMP shown in Figure 7.3.

Figure 7.4 shows which path to be selected to save new plug-in.

Figure 7.5 shows that the new plug-in has been successfully registered in GIMP and it is showing at the time of loading GIMP.

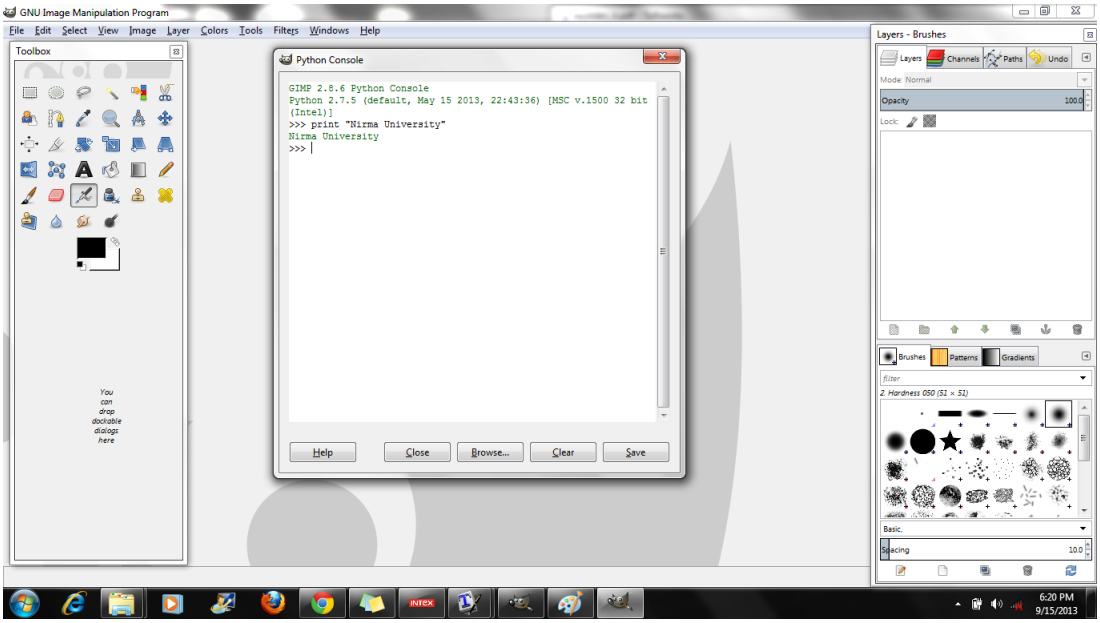

Figure 7.2: Python-Fu Console

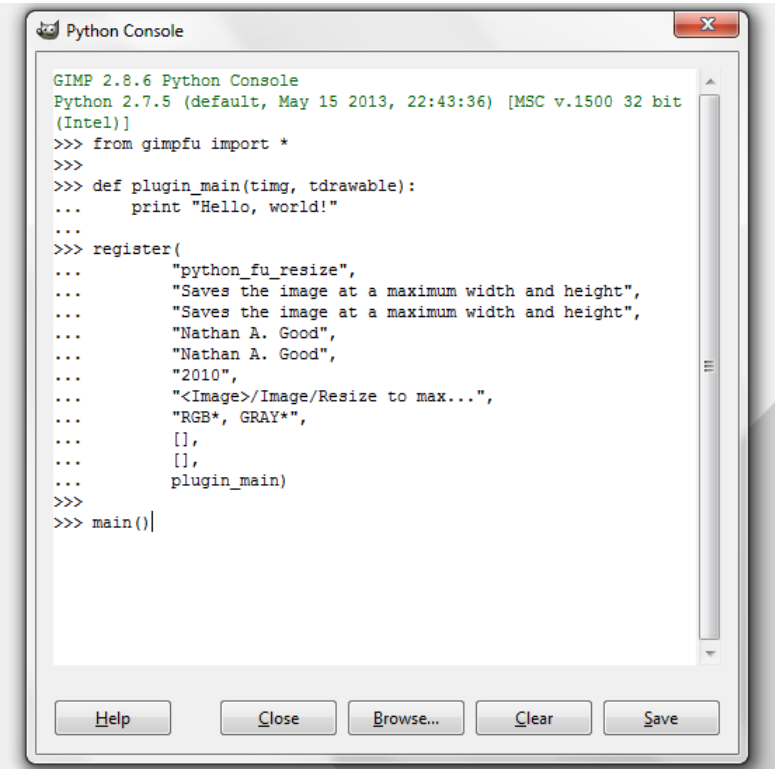

Figure 7.3: Code for New Plug-in in Python

2. After getting the new plug-in successfully registered in GIMP, the next step is to check whether the MAC address can be fetched from Python console in GIMP.

   Figure 7.6 shows the code used to fetch MAC address in Python.

   Figure 7.7 shows the MAC address fetched from the code above it.

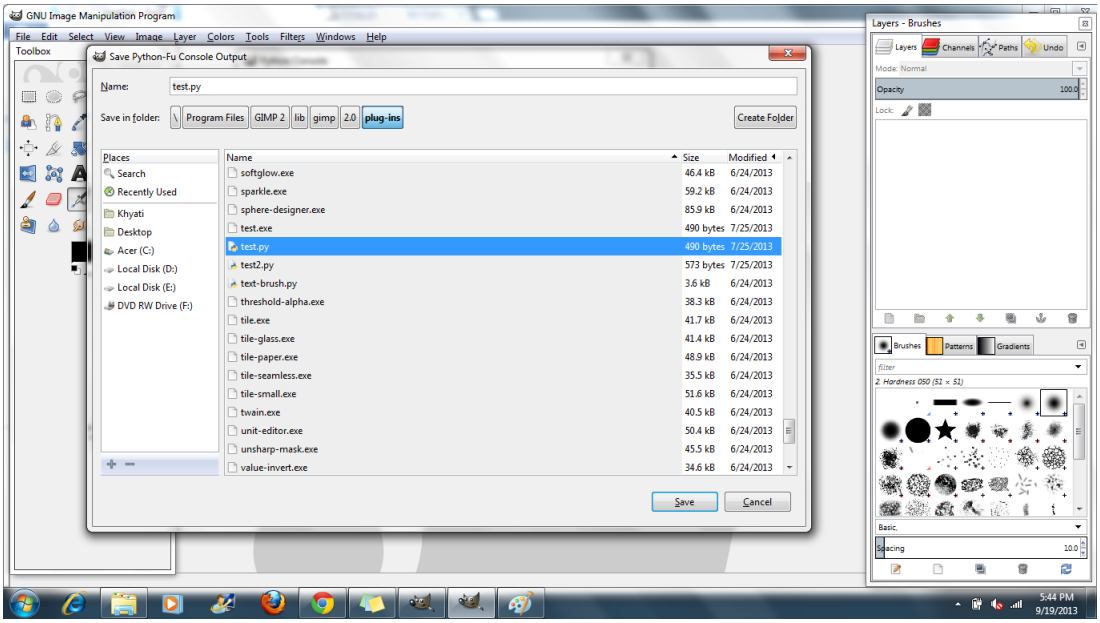

Figure 7.4: Saving New Plug-in

Figure 7.5: Presence of New Plug-in in GIMP

Looking at the getnode documentation, it says that it will return a random long if it fails to detect the mac: "If all attempts to obtain the hardware address fail, we choose a random 48-bit number with its eighth bit set to 1 as recommended in RFC 4122."[16]

One other noticeable thing is that uuid.getnode() can fake the MAC address by returning

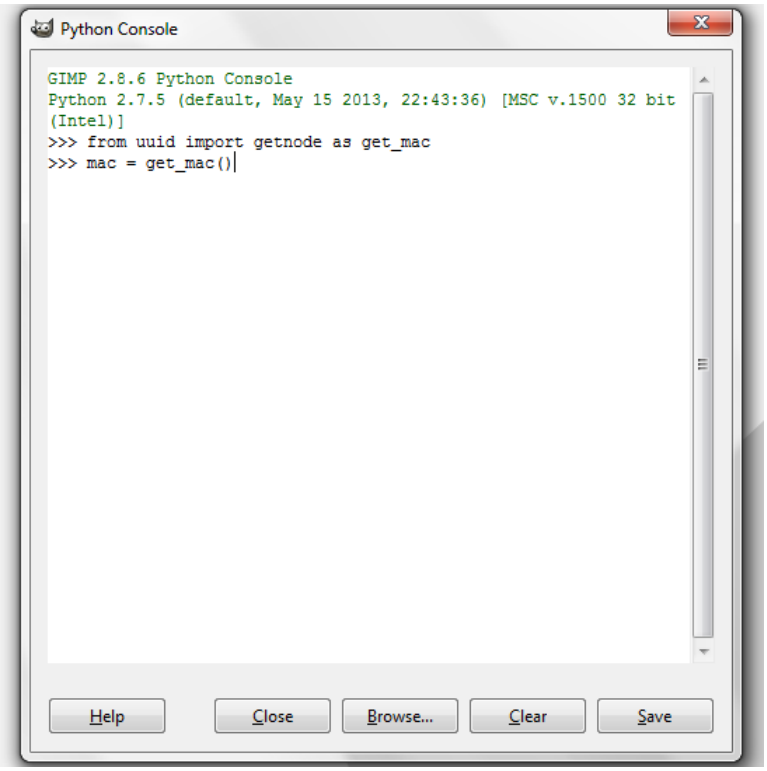

Figure 7.6: Code to fetch MAC address in Python

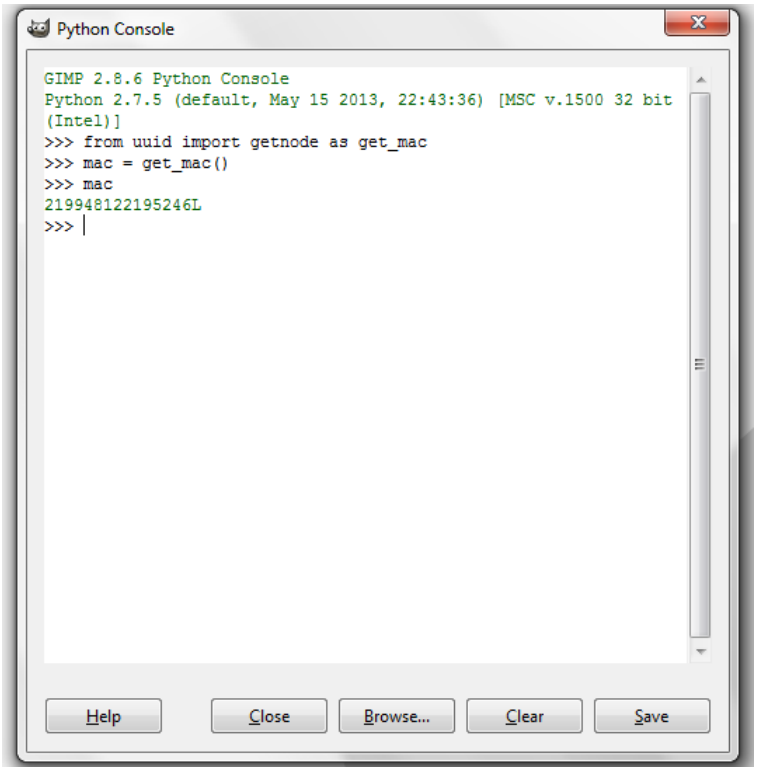

Figure 7.7: Returned MAC address in the form of 48-bit number

a random 48-bit number which may not be what is expected. Also, there's no explicit indication that the MAC address has been faked, but it can be detected by calling getnode() twice and seeing if the result varies. If the same value is returned by both calls, you have the MAC address, otherwise you are getting a faked address.

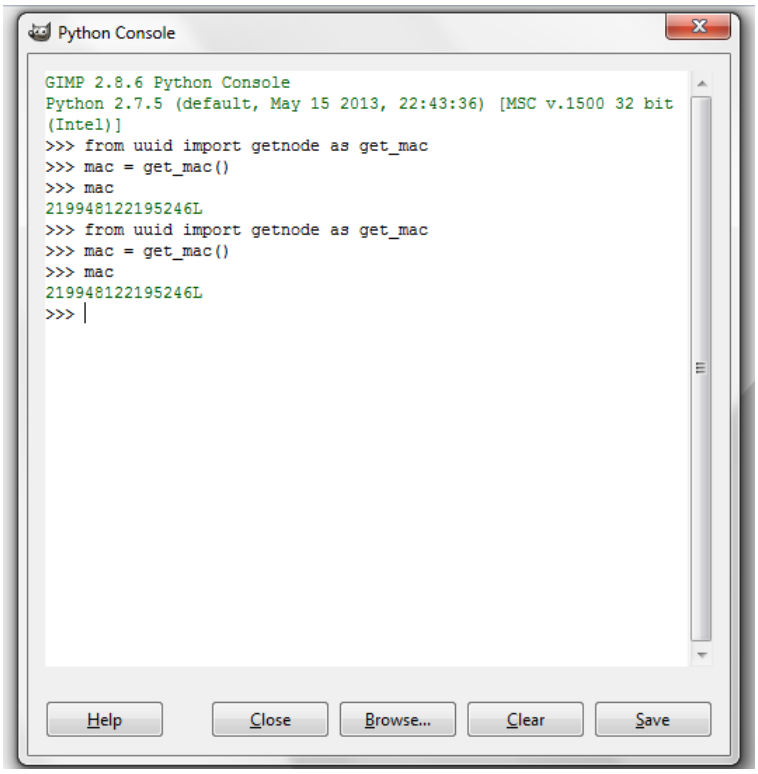

Figure 7.8: Confirming MAC address

Looking at figure 7.8, it is confirmed that the MAC address has been returned and it's not fake as same value is returned by both calls.

uuid.getnode represents current MAC address as an integer, this one-liner code in figure 7.9 formats this number in a standard MAC address form.

Thus, the MAC address of computer is fetched successfully.

3. The next step after fetching the MAC address is to develop code which embeds this MAC into image.

   The one way to do this is to use exif tags and to make use of exif and to edit or modify them in GIMP, the use of pexif library is must. Following are the steps to install pexif

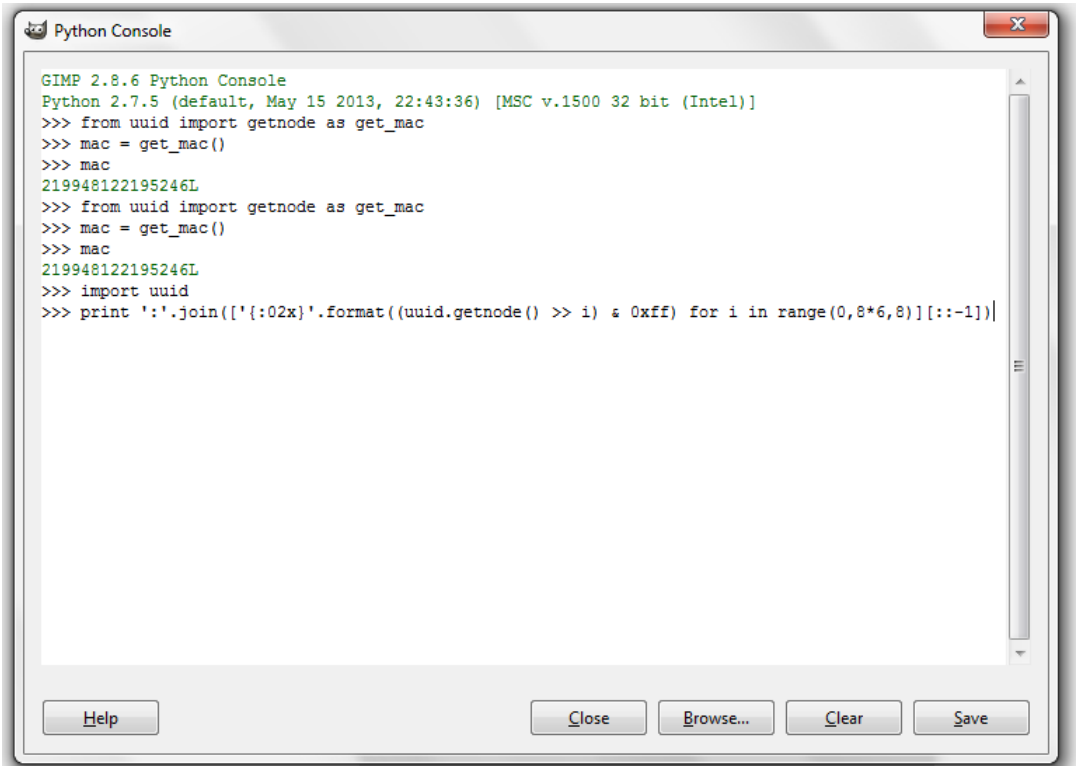

Figure 7.9: Code to print MAC address in human readable format

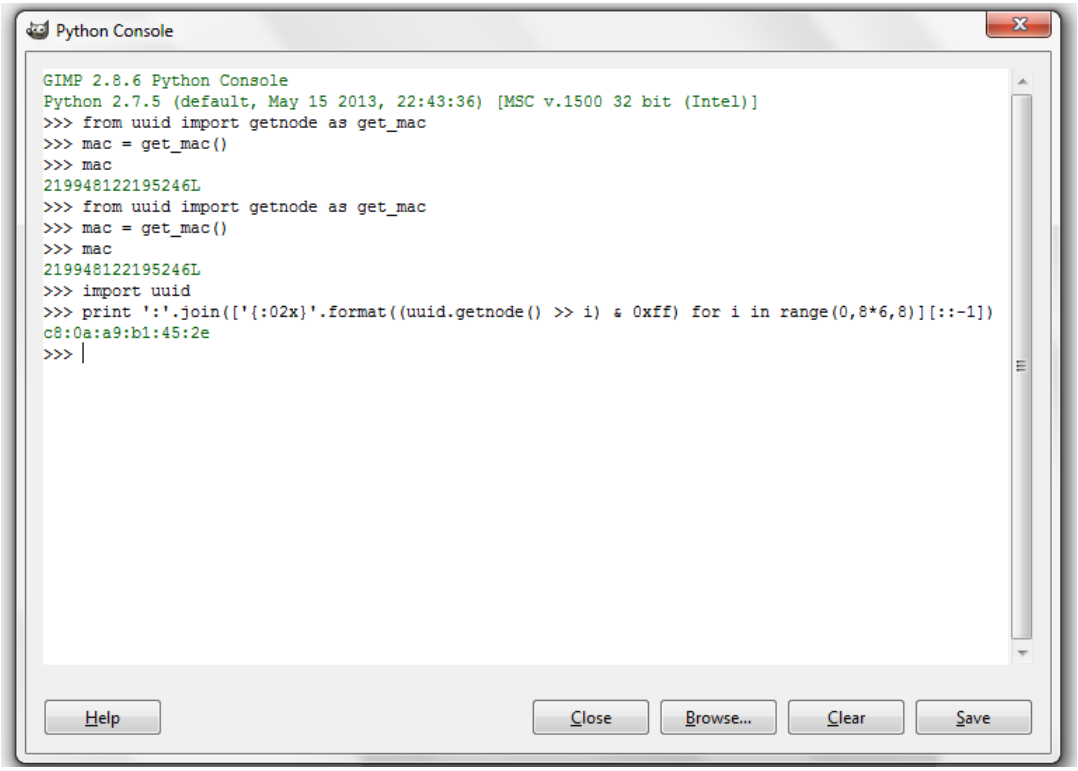

Figure 7.10: Returned MAC address in human readable format

library.

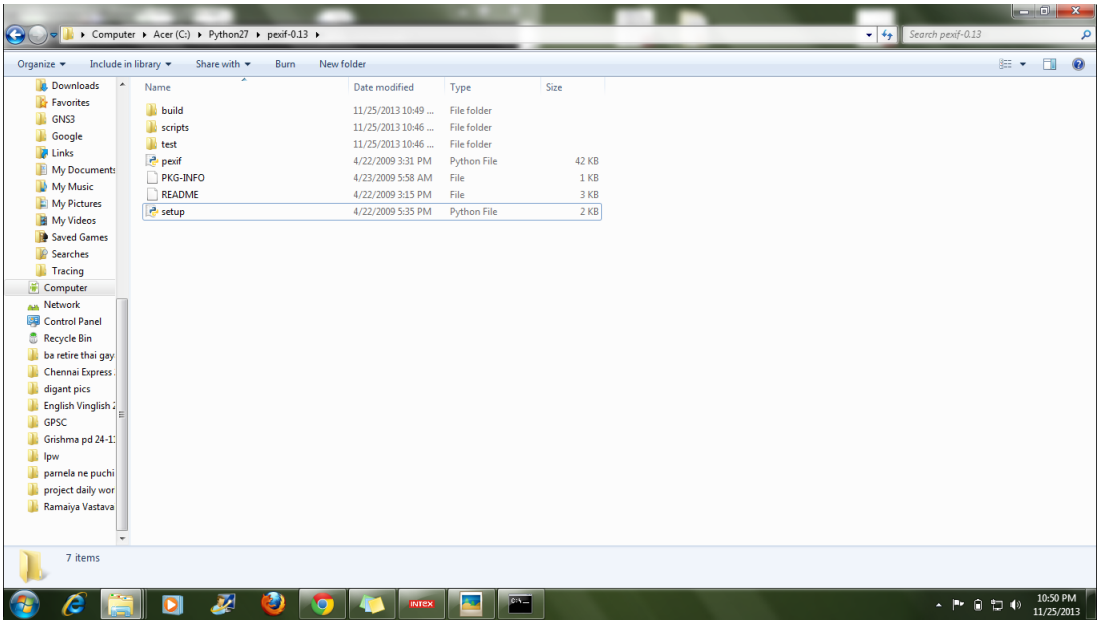

Figure 7.11: Path to extract Pexif source

Figure 7.11 shows the path where the pexif source should be extracted so that it can be installed successfully.

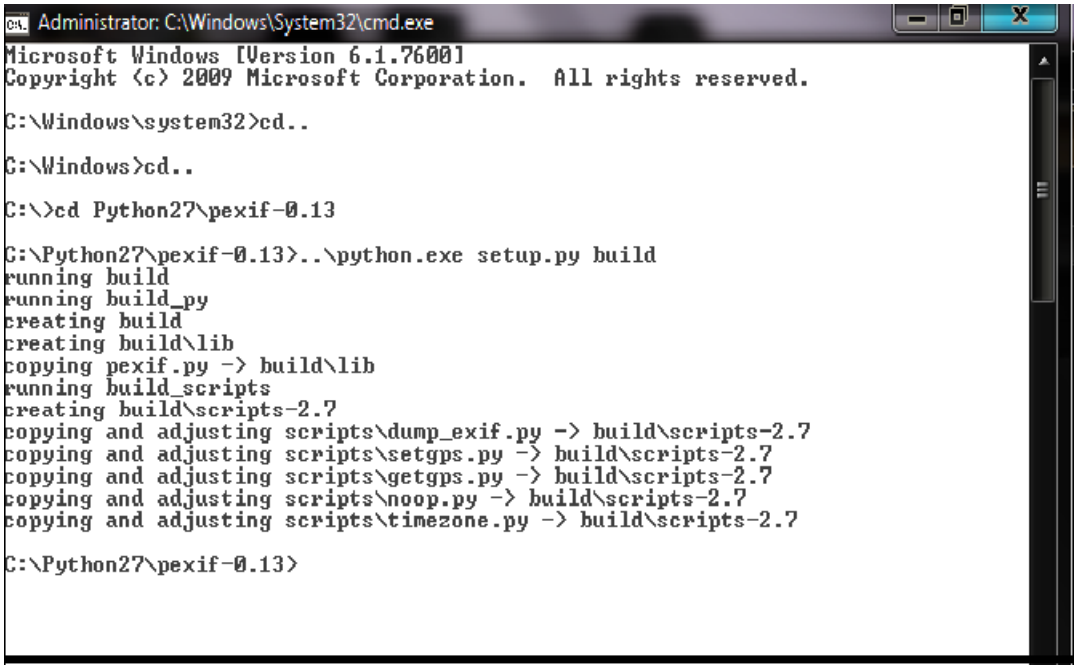

Figure 7.12: Compiling and building Pexif

After successfully compiling and building, all the scripts are built in build\scripts-2.7.

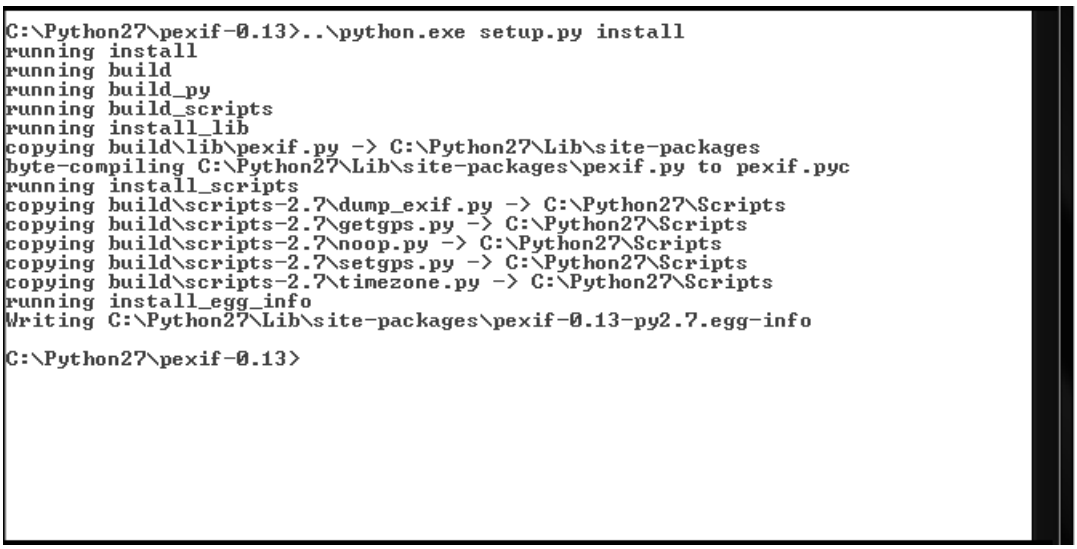

Figure 7.13: Installing Pexif

Returning to command prompt shows that pexif is successfully installed. After successful installation, the scripts are copied into C:\Python27\Scripts, which is shown in figure 7.13.

Figure 7.14 shows the final plug-in code that embeds the MAC address into the image in its Artist field which is one of the unused Exif field.

We have to copy and paste that code into **"C:\Program Files\GIMP 2\lib\gimp\2.0\plug-ins"** path so that we can see the plug-in existence in GIMP.

Moreover, the highlighted line shows that this plug-in uses an external script i.e. exiftool.exe which is located at **"C:\Windows"** path, which helps GIMP in modifying Exif data as GIMP does not have support for manipulating Exif data.

Figure 7.16 shows the existence of plug-in in GIMP File menu. The plug-in also has shortcut Ctrl+S assigned to it.

4. The final step is to show resulted images that have MAC addresses embedded into them.

   Here, it is shown for different image file formats like .jpg, .png, .gif and .tif. The plug-in works for all of these image file formats.

   Following are the command prompt snapshots that show the comparison between the original image and the image after using the plug-in.

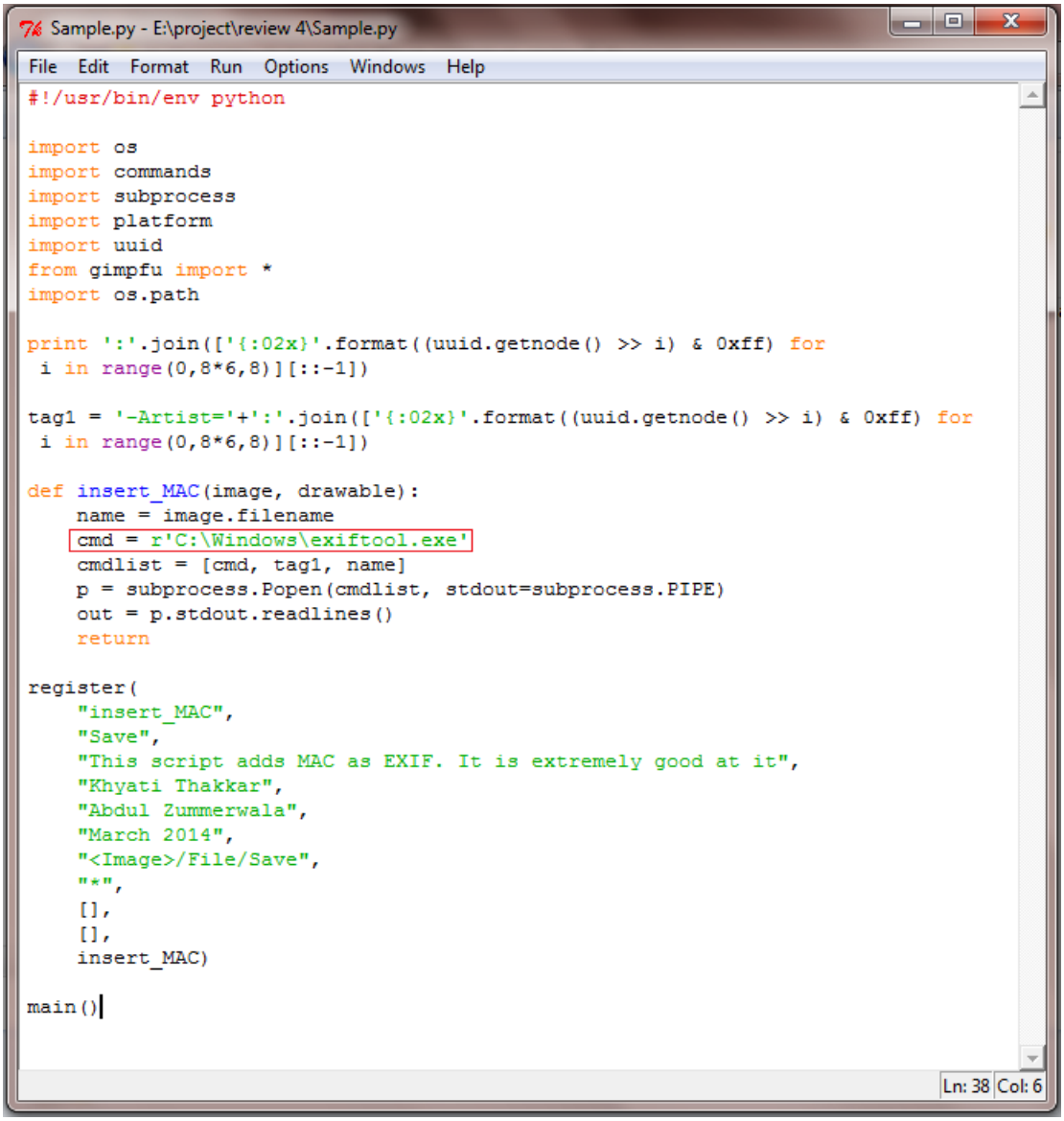

Figure 7.14: Final Plug-in Code

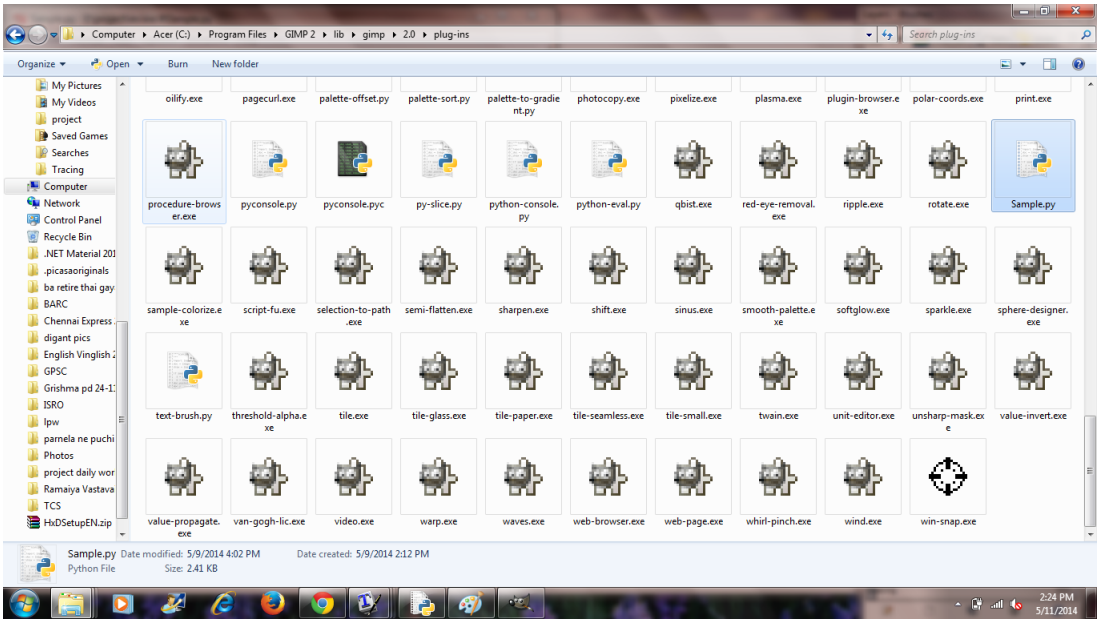

Figure 7.15: Copying Plug-in Code in GIMP

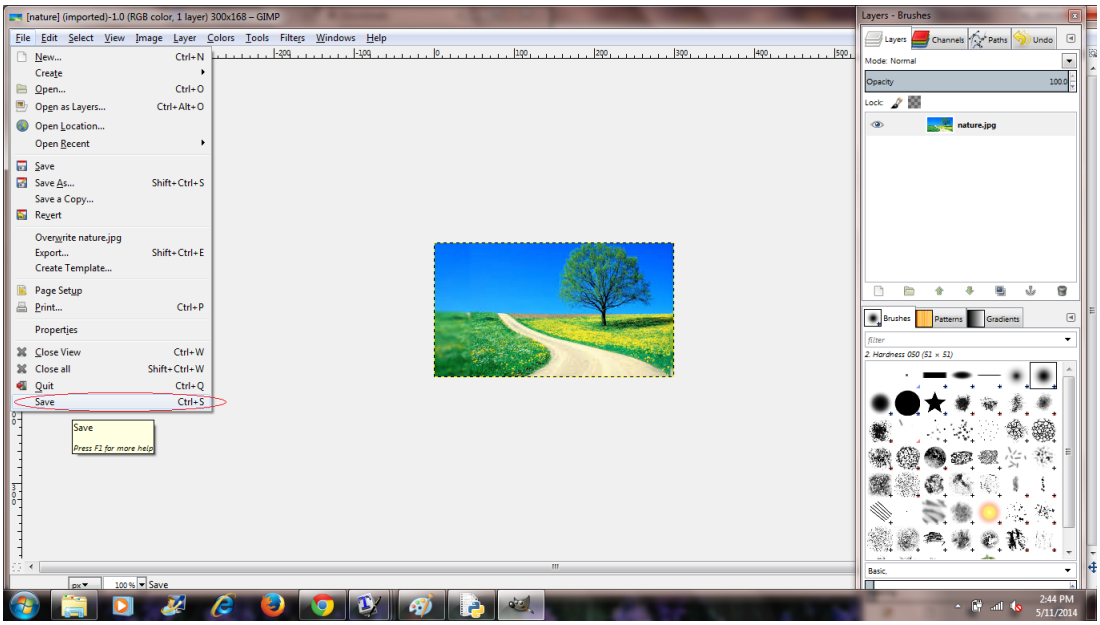

Figure 7.16: Plug-in Existence in GIMP

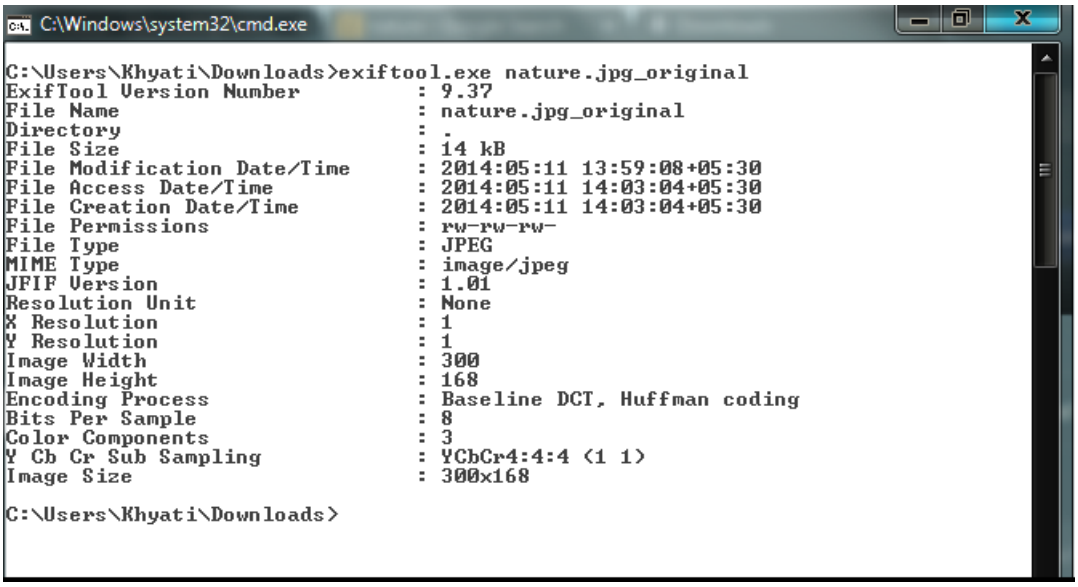

Figure 7.17: Original Image in .jpg Format

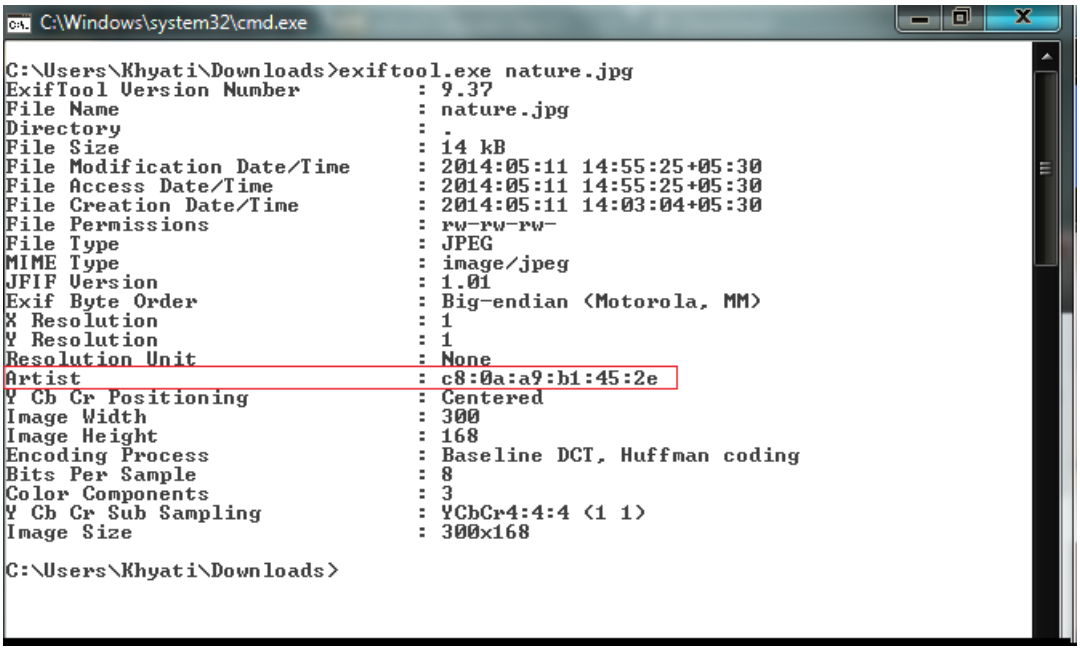

Figure 7.18: Edited Image in .jpg Format

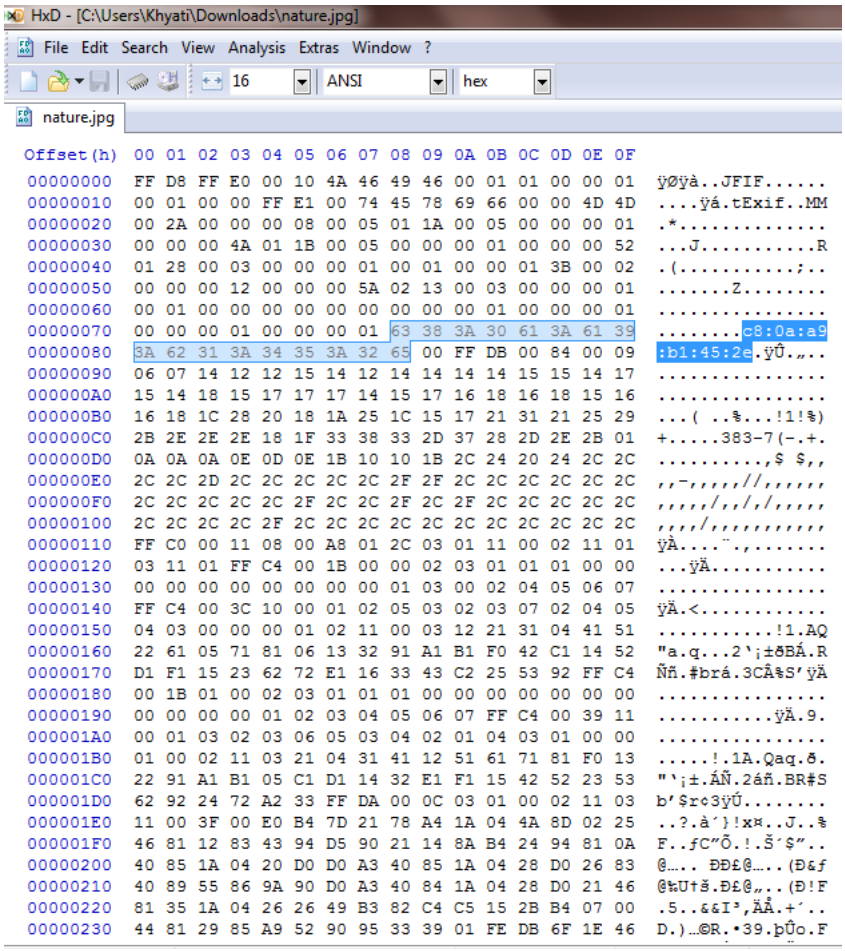

Figure 7.19: Edited Image in .jpg Format in Hex Editor

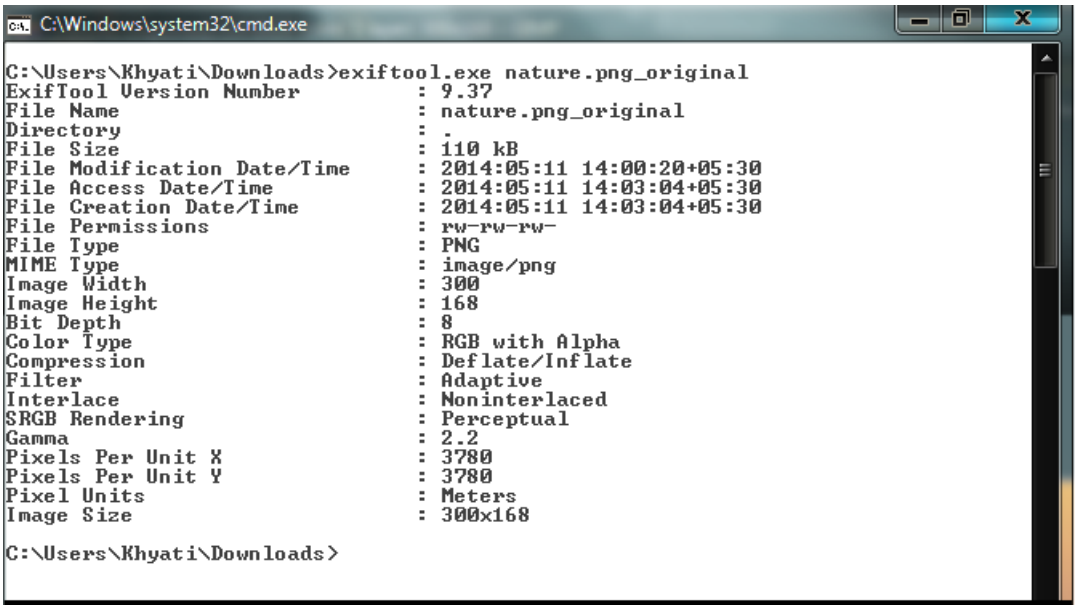

Figure 7.20: Original Image in .png Format

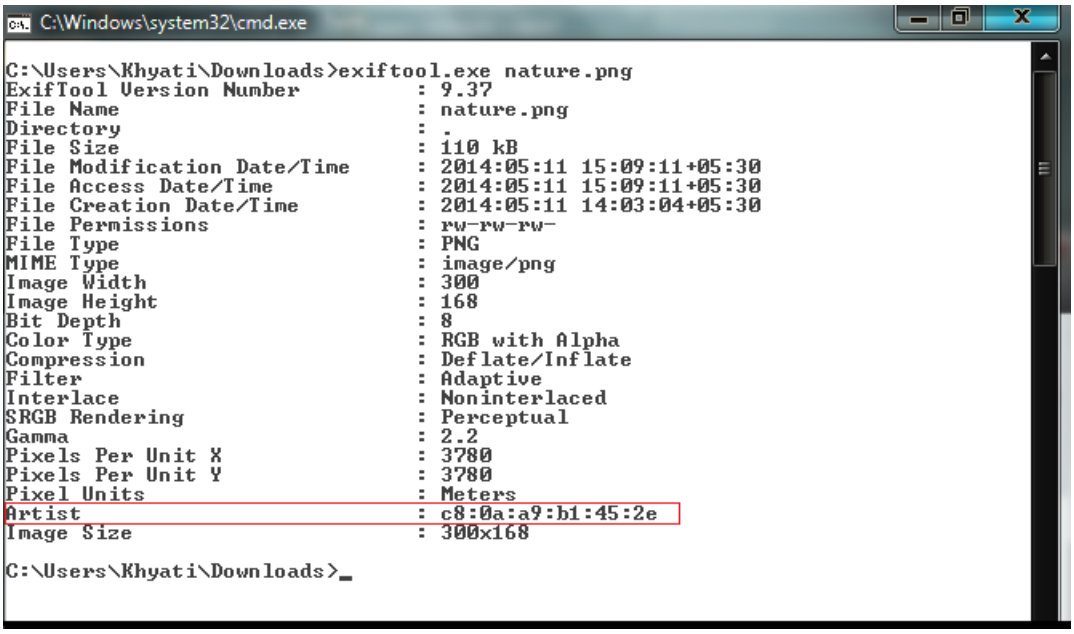

Figure 7.21: Edited Image in .png Format

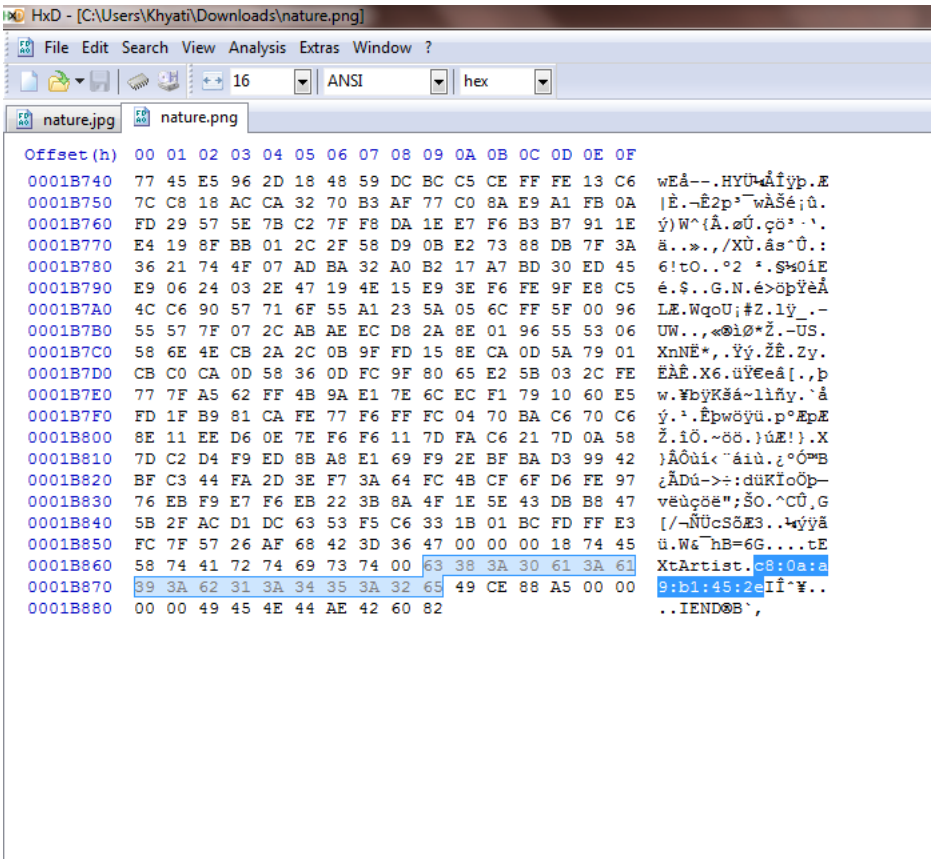

Figure 7.22: Edited Image in .png Format in Hex Editor

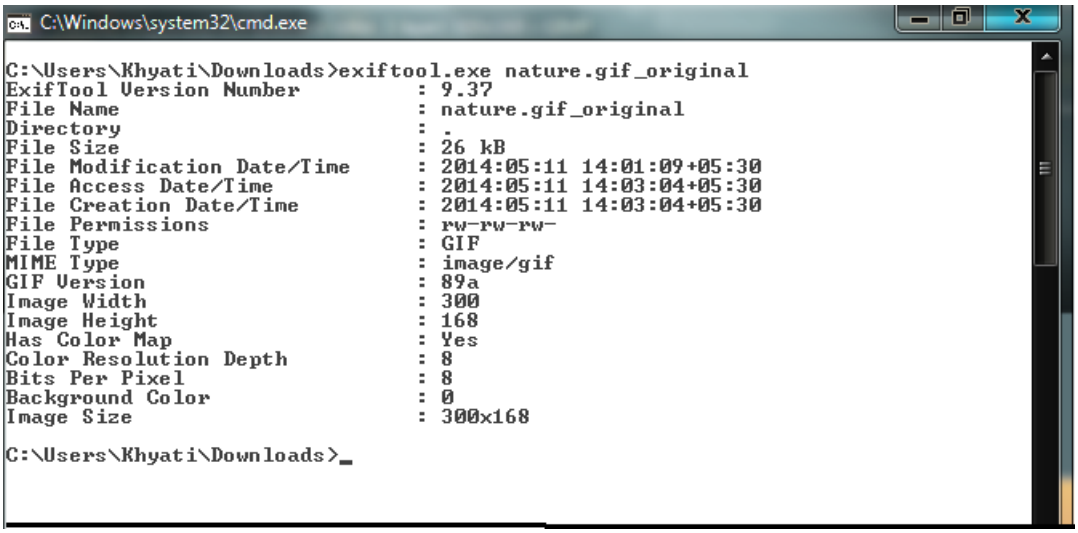

Figure 7.23: Original Image in .gif Format

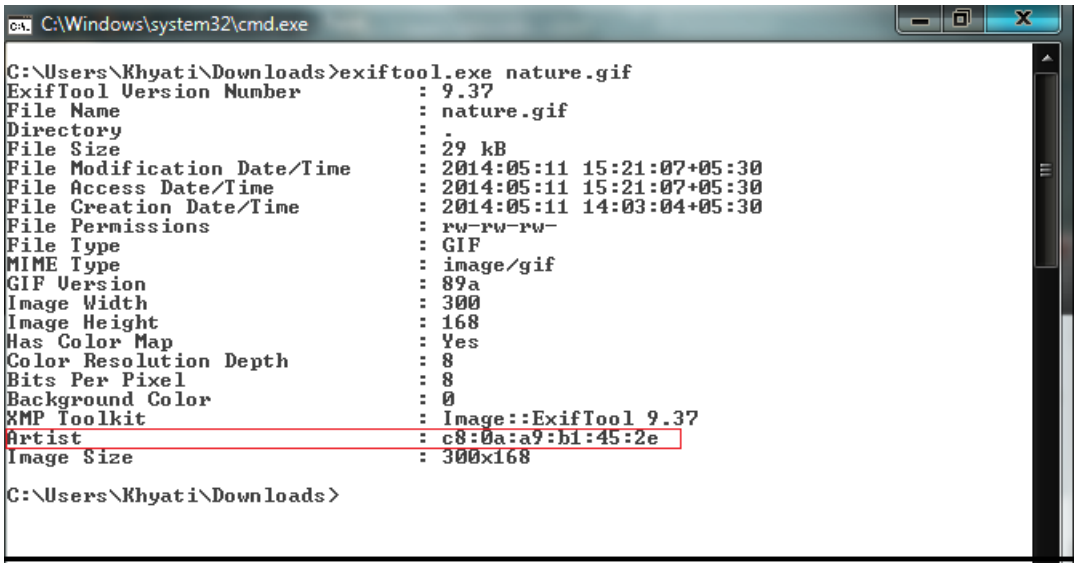

Figure 7.24: Edited Image in .gif Format

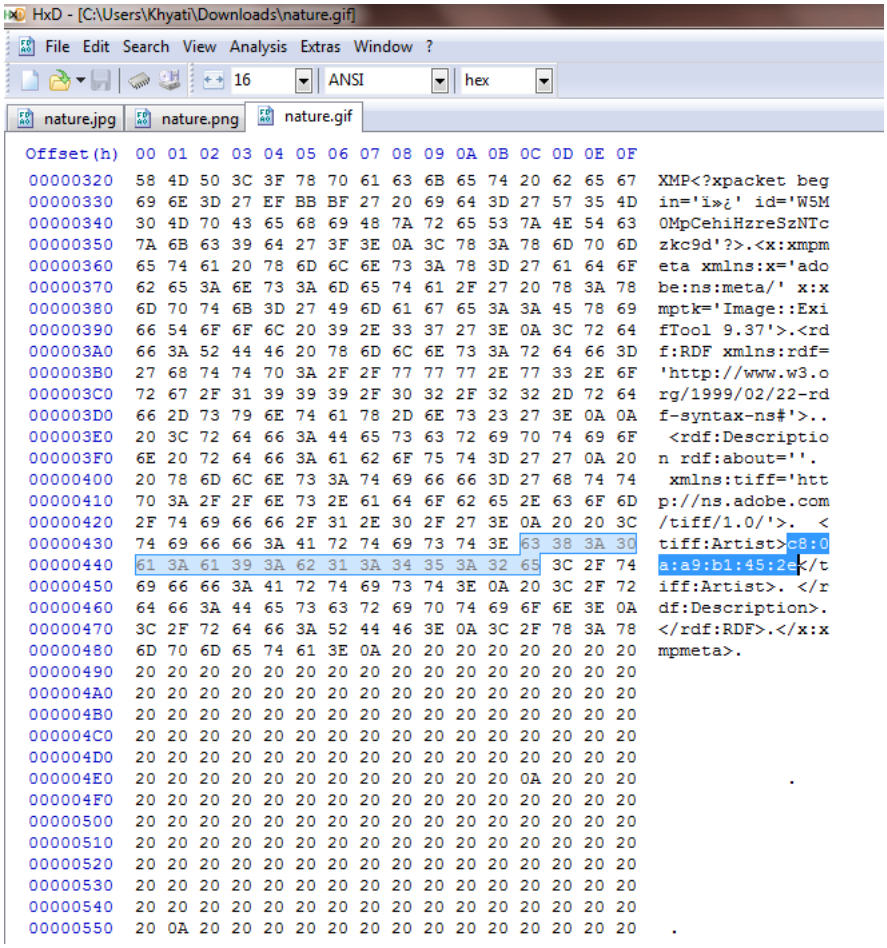

Figure 7.25: Edited Image in .gif Format in Hex Editor

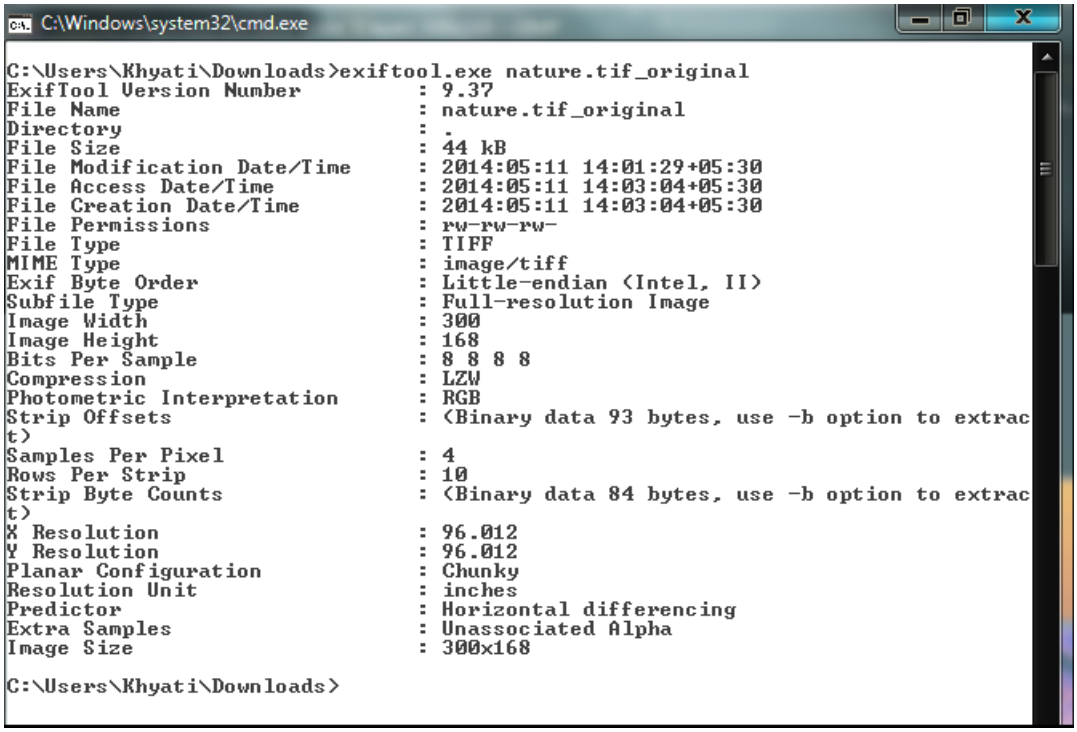

Figure 7.26: Original Image in .tif Format

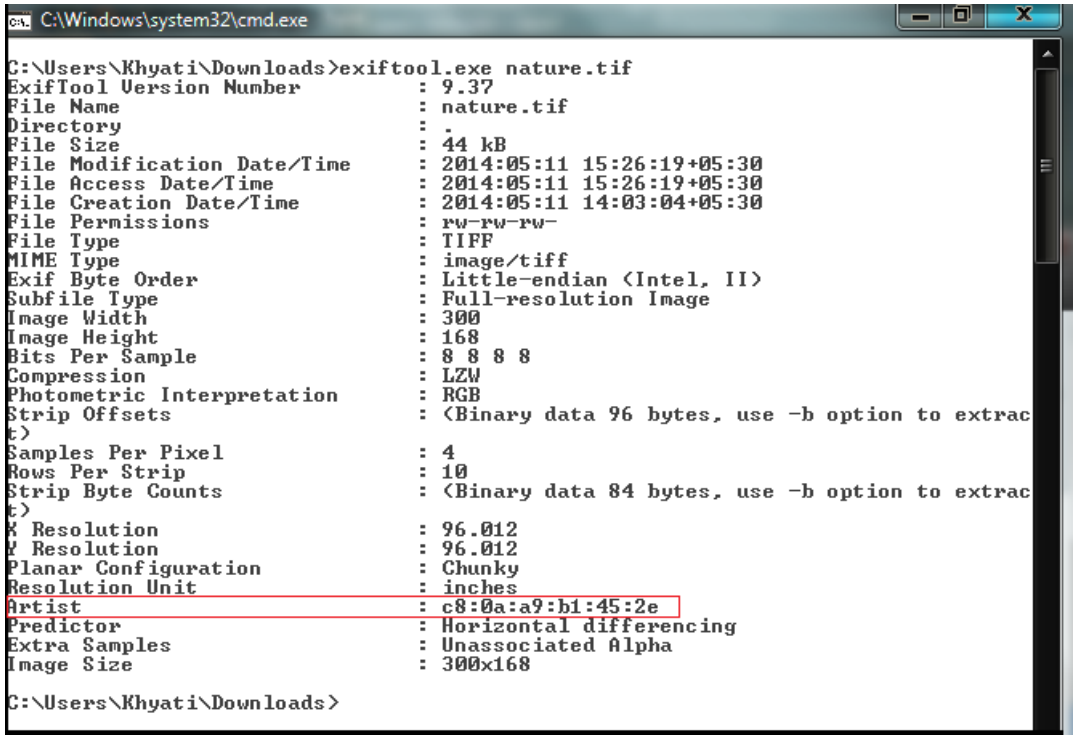

Figure 7.27: Edited Image in .tif Format

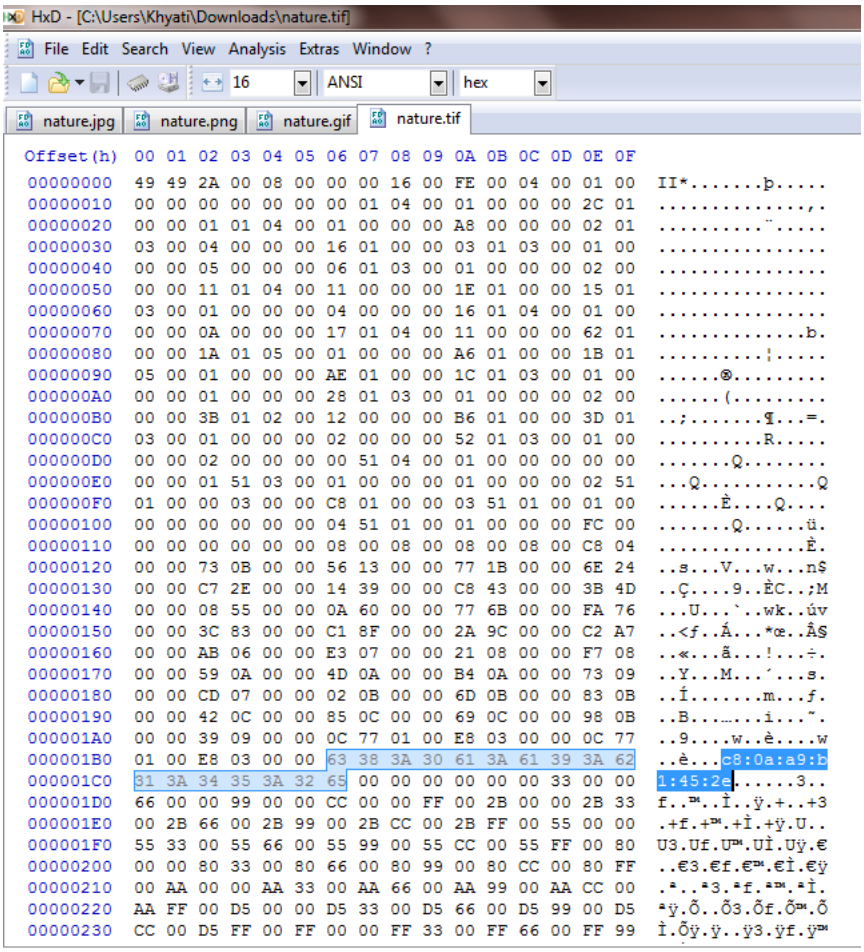

Figure 7.28: Edited Image in .tif Format in Hex Editor

Chapter 8

Conclusions

8.1 Current Status

The plug-in has been developed which embeds the computer's MAC address into the image(which is edited on that computer using GIMP) by editing the image's Exif data and result can be seen by fetching the Exif data of that image with command **exiftool.exe filename.file-format** or by opening the image in any hex editor.

8.2 Future Work

Currently, the plug-in only embeds the MAC address of computer on which the image has been lastly modified. The project can be extended to modify the plug-in code in such a manner that the image can be made to contain the series of all the MAC addresses of the computers on which it has been modified.

Chapter 9

Paper Published

Paper Published in: SAPIENCE'14 International Conference on Security and Authentication on 27-28th March, 2014 organized jointly by the department of Computer Applications, Sree Narayana Gurukulam College of Engineering (SNGCE) and School of Computer Science, Mahatma Gandhi University (MG University), Kerala.

Paper Title: A Novel Approach to Find the Artifacts of GodMode

Paper Authors: Ms. Khyati Thakkar, Ms. Twinkle Patel, Mr. Madhur Tewani, Mr. Kishan Varshney, Mr. Nilay Mistry and Dr M S Dahiya

Paper Abstract: Windows operating system (OS) is highly used by computer users across the globe. Windows OS is coming up with various secret tricks and tweaks to get into the administration control. "GodMode" is one of the hidden secret through which one can access the whole control panel administration though the user is either guest or standard user with limited privileges. By using GodMode, users can access any of the control panel settings from a single folder, which is a concept actually based upon NTFS virtual folder. Forensic Investigation of such hidden secret can give proper and supportive conclusion to the investigator, whether control panel settings opened directly or through the GodMode, also which user created the GodMode and controlled the system. This paper has some approaches given through which investigator can identify whether GodMode operations are done over the system to breach the privileges of administrator or not.

Bibliography

[1] $http://developer.gimp.org/git.html$

[2] $http://developer.gimp.org/writing-a-plug-in/1/index.html$

[3] $http://www.plagiarismtoday.com/2008/08/19/tineye-protecting-images-preventing-orphans/$

[4] $http://www.linuxplanet.com/linuxplanet/tutorials/6720/1$

[5] $http://www.linuxplanet.com/linuxplanet/tutorials/6720/2$

[6] $http://www.python-forum.org/viewtopic.php?f=6\&t=177$

[7] $http://code.activestate.com/recipes/550811-jpg-files-redater-by-exif-data/$

[8] $http://www.endlesslycurious.com/2011/05/11/extracting-image-exif-data-with-python/$

[9] $http://www.blog.pythonlibrary.org/2010/03/28/getting-photo-metadata-exif-using-python/$

[10] $http://stackoverflow.com/questions/159137/getting-mac-address$

[11] $http://code.activestate.com/recipes/578277-get-mac-address-of-current-interface-in-one-line-o/$

[12] $http://code.google.com/p/pexif/$

[13] $http://code.google.com/p/pexif/downloads/list$

[14] $http://en.wikipedia.org/wiki/Exchangeable_image_file_format$

[15] $http://www.exif.org/$

[16] $http://www.ietf.org/rfc/rfc4122.txt$

[17] $http://en.wikipedia.org/wiki/GIMP$

[18] $http://www.sno.phy.queensu.ca/\ phil/exiftool/$

[19] $https://raw.githubusercontent.com/akkana/gimp-plugins/master/save-export-clean.py$

[20] $http://docs.gimp.org/2.6/en/gimp-concepts-shortcuts.html$

[21] $https://github.com/akkana/gimp-plugins$

[22] $http://webcache.googleusercontent.com/search?q=cache:http://blog.defron.org/2013/01/gimp-script-save-as-png.html$

[23] $http://shallowsky.com/software/gimp-save/$

[24] $http://www.gimphelp.org/python_save_as_jpg.shtml$

[25] $http://www.gimphelp.org/DL/Python_scripts/save_as_jpg.py$