

Applying Process Observation Techniques

Offensive and Defensive Tool Construction

Table of Contents

[Objectives 4](#_Toc477267494)

[Background Reading 4](#_Toc477267495)

[Important Information 4](#_Toc477267496)

[Introduction 5](#_Toc477267497)

[Problem 1 5](#_Toc477267498)

Offensive and Defensive Tool Construction

Applying Process Observation Techniques

Objectives

This lab focuses on the following objectives:

* Use soft hooking with Python Debugger.
* Use hard hooking with Immunity Debugger.
* Demonstrate the Python to IDA Pro API.

Background Reading

Read chapter 6 in the *Gray Hat Python* textbook. The following links are also useful:

* <https://www.sans.org/reading-room/whitepapers/malicious/basic-reverse-engineering-immunity-debugger-36982>
* <https://sgros-students.blogspot.ca/2014/05/immunity-debugger-basics-part-1.html>

# Important Information

* For *every* lab and home assignment, store all your work in your personal repository in a subdirectory named **mXX**, where XX is the module number. Carefully name the program as described in each problem.
* Your programs are extracted from your repository by a Python script. If there are any errors in the program name, then your instructor will never see your program, and you will receive a mark of zero.
* Push your work to the server often, and ensure that you push the final version of a program by the deadline specified, because the script extracting them can be run at any time after the deadline.

# Introduction

In this lab, we will explore the Immunity Debugger further and demonstrate soft and hard hooking techniques. Read section 6.1 in the *Grey Hat Python* textbook, which describes soft hooking with the PyDbg debugger. In this problem, we will place soft hook in the Firefox browser.

The PyCommands are found in **/c/Program Files/Immunity Inc/Immunity Debugger/PyCommands**.

To edit files at that location, you must start your code editor (e.g., Emacs) with admin privileges.

# Problem 1

The goal of this problem is to intercept non-encrypted login and password information in the Firefox browser. We will access an encrypted web page at https://itss.biomea.com/login.html [replace with relevant SAIT local website]. The page requests the user ID and a password, and the Firefox browser sends this information in encrypted format over the Internet. We want to exploit the username and password information before it is encrypted and sent to the server.

The general concept follows section 6.1 in the *Grey Hat Python* textbook, but you will write the exploit as an Immunity Debugger PyCommand script named **m08p01.py**.

The result should show the plaintext username and password in the Immunity Debugger log file (right-click the **Log** window, choose **Log to File**, and then enter filename **m08p01.log** at the appropriate Module 8 location in your Git repository).

**Hints:**

* The dll where the **PR\_Write()** function lives is called **nss3.dll** and is part of the Firefox distribution.
* There are several very useful functions in the Immunity Debugger libraries:

callStack()

getProcedure()

getStackDump()

readMemory()

getAllModules()

getName()

setLoggingBreakpoint()

getAllSymbolsFromModule()

**Note:** The rest of the problems for this module are available in the homework assignment. See your course schedule for details.