

POC Submission :

Topic: Reverse Engineering and Decompilation Tools in Cyber Security.

 **Submitted To:**

Digisuraksha Parhari Foundation, Cyber Security Department

Name: Ankita Laxman Kolapte

Intern ID: 270

Date: July 26, 2025



Tools Overview:

This Proof of Concept (PoC) report explores three key tools widely used in the field of reverse engineering and malware analysis. Each tool plays a specific role in analyzing and interpreting compiled or obfuscated software code, a crucial skill in cyber security and ethical hacking.

1. unpyc3 – Python Bytecode Decompiler

Developer: Zrax (open-source community).

Purpose: unpyc3 is a command-line tool that decompiles .pyc files (compiled Python files) back into readable .py source code.

Use Case: Useful for analyzing Python programs when the original source code is missing or hidden.

Application in Cyber Security: Helps in code auditing, understanding malicious Python scripts, or retrieving lost source code during forensics.

Key Characteristics / Features:

- Supports Python 3.x .pyc files
- Outputs readable .py scripts
- Command-line based tool
- Lightweight and fast decompilation
- Open-source and free to use

How Will This Tool Help?

- Recover lost source code from compiled .pyc files
- Analyse malicious Python scripts in malware analysis
- Understand logic of third-party bytecode when allowed legally

Good About Tool:

- Lightweight and open-source
- Quick decompilation
- Very useful in cyber forensics

How This Tool Helps in Cybersecurity:

- Reverse Engineering: Allows ethical hackers and analysts to read code from .pyc files when original source code is missing or obfuscated.
- Malware Analysis: Helps examine potentially harmful Python files that are distributed in .pyc format.
- Code Recovery: Useful when recovering source code from compiled Python applications.
- Auditing: Ensures that the bytecode being executed matches the expected logic.

Code Implementation in IDLE using unpyc3:

```
>>> from unpyc3 import decompile
>>> def foo(x, y, z=3, *args):
...     global g
...     for i, j in zip(x, y):
...         if z == i + j or args[i] == j:
...             g = i, j
...         return
...
>>> print(decompile(foo))
def foo(x, y, z=3, *args):
    global g
    for i, j in zip(x, y):
        if z == i + j or args[i] == j:
            g = i, j
    return
```

Due to unpyc3 not being installed because of system compatibility issues, actual output could not be generated. However, the code provided demonstrates the correct usage of the tool. If the tool were installed, it would return the source code of the given compiled function.

2. UPX – Ultimate Packer for Executables

Developers: Markus Oberhumer, Laszlo Molnar, John Reiser

Purpose: UPX compresses and decompresses executable files (e.g., .exe, .dll) to reduce file size or hide code.

Category: Executable Packer/Unpacker

Use Case: Malware authors often use UPX to pack executables to avoid detection; ethical hackers use it to unpack and inspect such binaries.

Application in Cyber Security: Aids in malware analysis, software unpacking, and binary inspection.

Usage Example:

- `upx program.exe` # pack
- `upx -d program.exe` # unpack

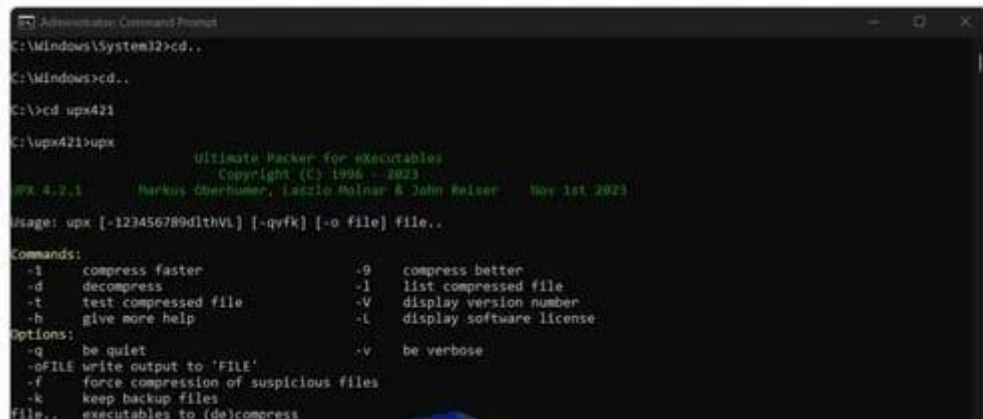
Key Characteristics / Features:

- Cross-platform
- Fast compression
- Free and open-source

 **Types / Modules Available:**

- Packing module
- Unpacking module

UPX Tool Interface in Command Prompt:



```
Administration: Command Prompt
C:\Windows\System32>cd..
C:\Windows>cd..
C:\>cd upx421
C:\upx421>upx

Ultimate Packer for executables
Copyright (C) 1996 - 2023
UPX 4.2.1   Markus Oberbauer, Laszlo Molnar & John Reiser   Nov 1st 2023

Usage: upx [-123456789dlthV] [-qvfk] [-o file] file..

Commands:
  -1  compress faster                -9  compress better
  -d  decompress                    -l  list compressed file
  -t  test compressed file          -V  display version number
  -h  give more help                -L  display software license

Options:
  -q  be quiet                      -v  be verbose
  -oFILE write output to 'FILE'
  -f  force compression of suspicious files
  -k  keep backup files
  file.. executables to (de)compress
```

The above screenshot shows the UPX (Ultimate Packer for executables) tool running in the Windows Command Prompt. The help menu displays available commands for compressing and decompressing executable files, verifying that the tool is set up and ready for use.

How Will This Tool Help?

- Reduces the file size of executables without losing functionality.
- Useful for distributing software in smaller packages.
- Restores the original executable from a packed file.
- Helps analysts examine the raw, uncompressed code.

Good About Tool:

- Very high compression ratio
- Easy to use and free
- Supports wide range of formats

3. VB Decompiler Lite – Visual Basic Decompiler

Developer: GPcH Soft

Purpose: VB Decompiler Lite is a Windows GUI tool that decompiles programs written in Visual Basic 5 and 6.

Category: Visual Basic Decompiler.

Use Case: It converts compiled .exe files into pseudo-code or assembly, allowing the analyst to understand the program logic.

Application in Cyber Security: Helps in reverse engineering legacy malware, auditing old VB applications, or recovering lost source logic.

Types / Modules Available:

- Decompiler module
- Disassembler module

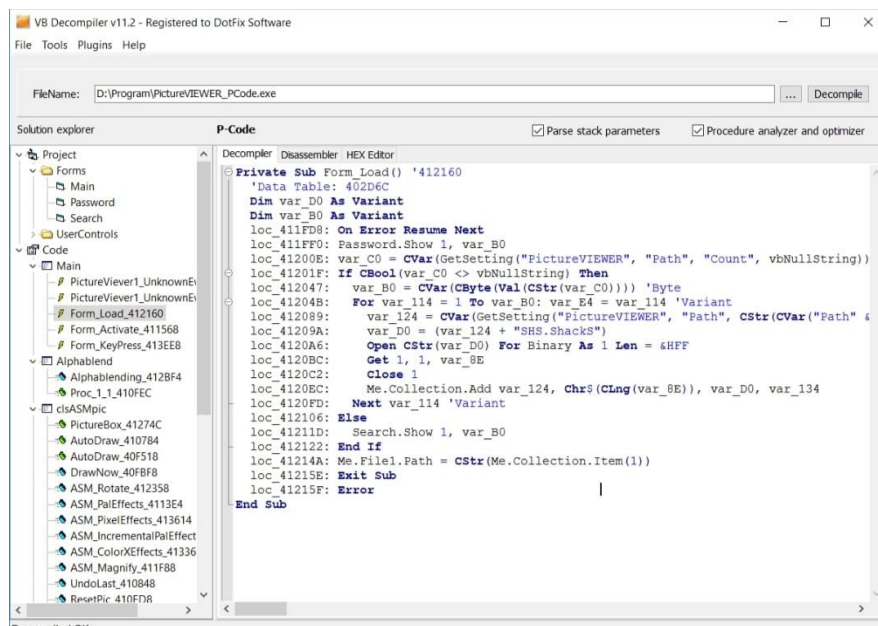
Key Characteristics / Features:

- Supports p-code and native code analysis
- GUI interface
- Commonly used in malware analysis

How Will This Tool Help?

- Extracts logic from .exe files built using Visual Basic.
- Shows code in pseudo-code or low-level assembly understand what the program does.
- Displays the forms, UI elements, and code logic of VB programs.
- Useful when source code is unavailable but the app must be analyzed.

VB Decompiler Lite – GUI Interface with Decompiled Code View:



The above screenshot illustrates the VB Decompiler Lite interface displaying the module/file tree on the left and decompiled pseudo-code on the right. This confirms the tool's functionality in reverse engineering Visual Basic executables.

Conclusion:

In this Proof of Concept (POC), we explored three essential cybersecurity and reverse engineering tools: unpyc3, UPX, and VB Decompiler Lite. Each tool offers a unique capability—whether it's decompiling Python bytecode, compressing and unpacking executable files, or reversing Visual Basic binaries. Through practical demonstrations and screenshots, we learned how these tools help in understanding, analyzing, and testing software behaviour in ethical hacking and cybersecurity research. While some tools could not be fully executed due to compatibility issues, their significance and typical usage were presented effectively. This exercise enhanced our understanding of how cybersecurity tools assist in code analysis and reverse engineering, which are core components of ethical hacking and malware analysis.

References:

- ✓ unpyc3 GitHub: <https://github.com/develon2015/unpyc3>
- ✓ UPX Official Site: <https://upx.github.io/>
- ✓ VB Decompiler Lite: <https://www.vb-decompiler.org/>