Practical Reversing II – Unpacking EXE

Nagareshwar Talekar

www.SecurityXploded.com

Disclaimer

The Content, Demonstration, Source Code and Programs presented here is "AS IS" without any warranty or conditions of any kind. Also the views/ideas/knowledge expressed here are solely of the trainer's only and nothing to do with the company or the organization in which the trainer is currently working.

However in no circumstances neither the trainer nor SecurityXploded is responsible for any damage or loss caused due to use or misuse of the information presented here.

Acknowledgement

- Special thanks to **null** & **Garage4Hackers** community for their extended support and cooperation.
- Thanks to all the trainers who have devoted their precious time and countless hours to make it happen.

Reversing & Malware Analysis Training

This presentation is part of our **Reverse Engineering & Malware Analysis** Training program. Currently it is delivered only during our local meet for FREE of cost.

For complete details of this course, visit our Security Training page.

Who am I

Nagareshwar Talekar

- Founder of SecurityXploded
- Reverse Engineering, Malware Analysis, Cryptography, Password Forensics, Secure Coding etc.
- Email: tnagareshwar at gmail.com

Course Q&A

- Keep yourself up to date with latest security news
  - http://www.securityphresh.com
- For Q&A, join our mailing list.
  - http://groups.google.com/group/securityxploded

Contents

- What is EXE Packing?
- Purpose of Packing EXE
- What is Unpacking?
- Detection of Packer
- Execution of Packed EXE Program
- Standard Process of Unpacking EXE
- Unpacking UPX using OllyDbg
- DEMO - Unpacking UPX
- Anti Anti-Debugging Plugins
- References

What is EXE Packing/Protecting?

- **EXE Packing:**

  Compressing the Executable to a smaller Size

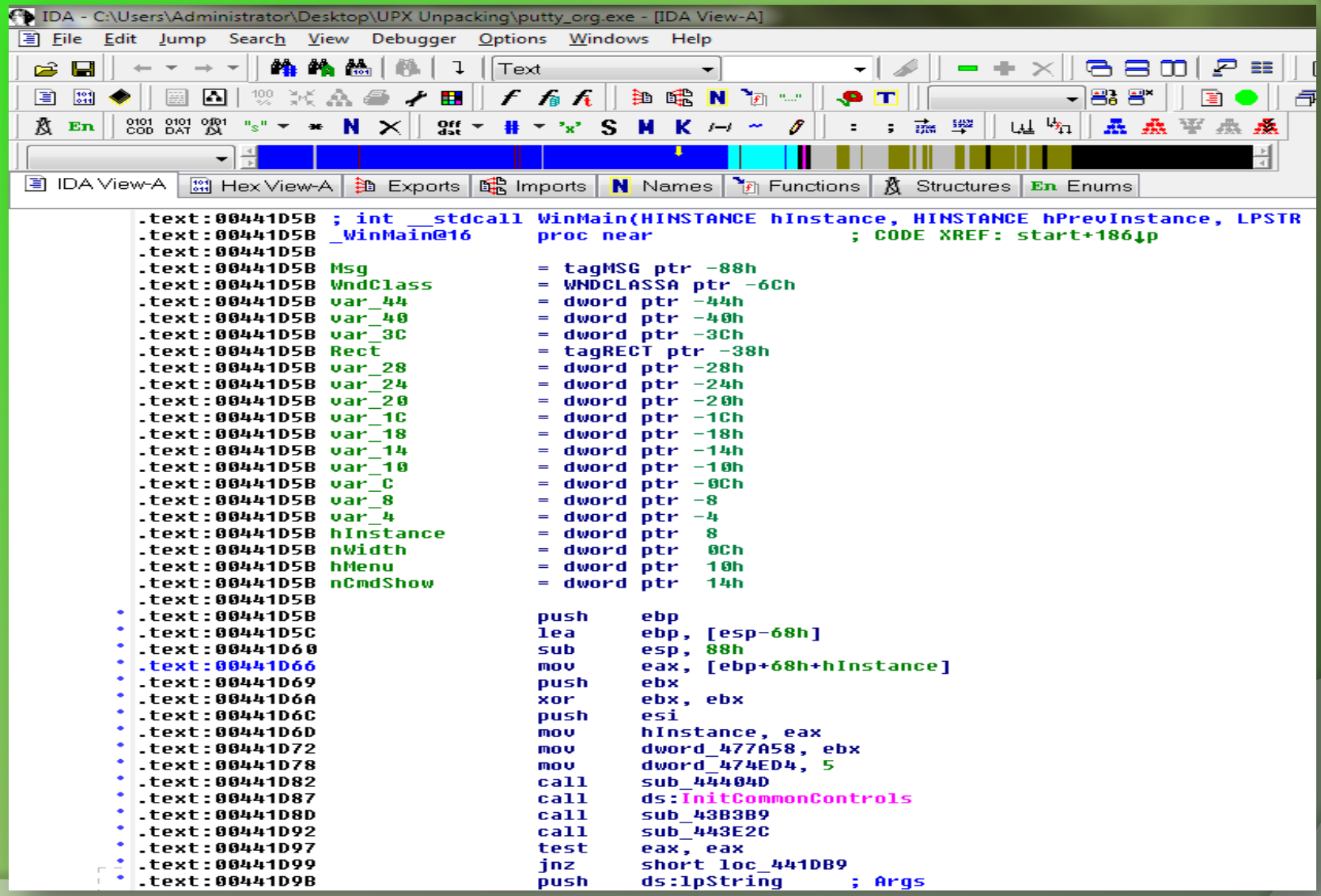
- **EXE Protecting:**

  Encrypting with Anti-Debugging Techniques to prevent Reversing

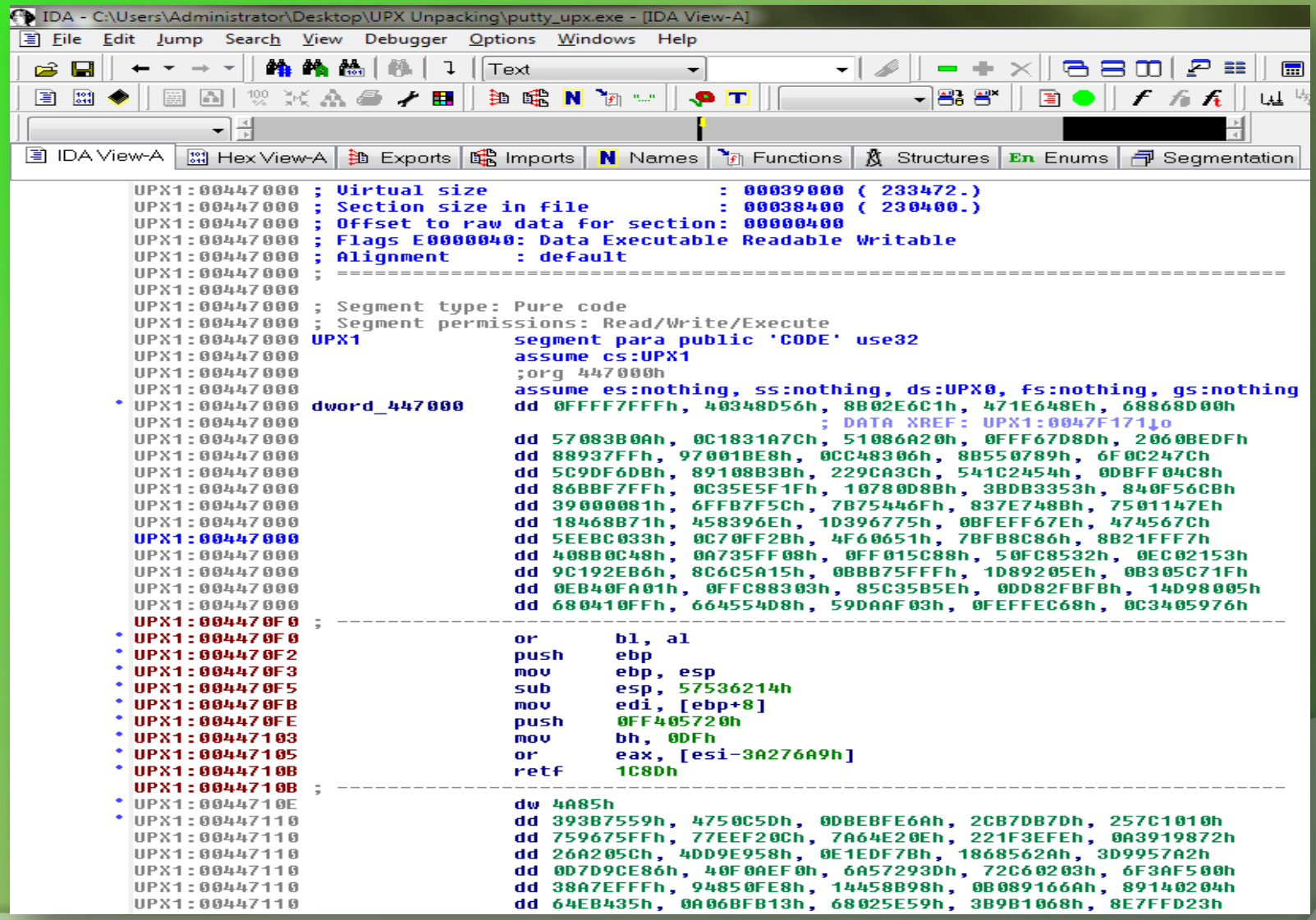
✓ In Reversing world, both Packer & Protector is commonly referred as **Packer**.

Examples of Packers: UPX, AsProtect, Armadillo etc.

EXE - Before Packing



EXE - After Packing



Purpose of Packing EXE

- **Prevent Reverse Engineering [Crack License, Secret Code etc.]**
  - **Defeat Static Disassembling**
  - **Make Dynamic Debugging Difficult**
- **Reduce the size of Executable file**
- **Bypass Anti-virus Detections with multi-level Packing**

✓ **It is used by Software Vendors to prevent Serial Cracking and Malware Authors to prevent analysis by AV Researchers.**

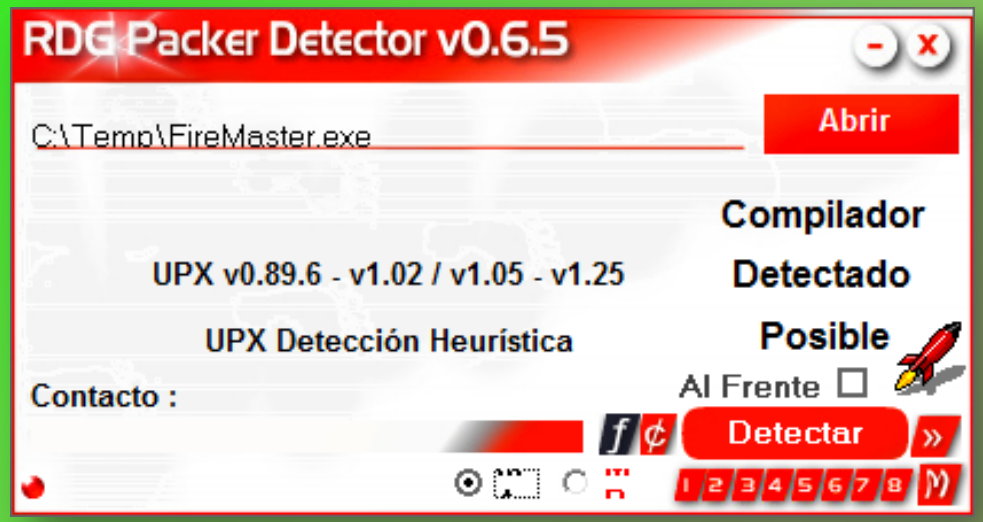
What is Unpacking?

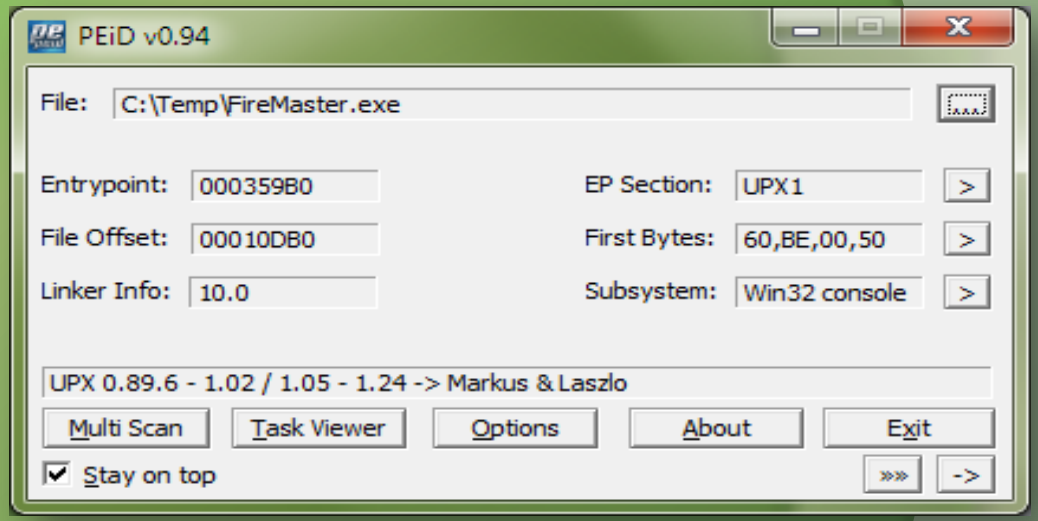
- **Extracting the Original Binary from the Packed Executable File.**
- **Automatic Unpackers available for popular Packers.**
  - **May not work with different versions**
  - **Not available for Complex packers**
- **Involves Live Debugging by Defeating Anti-Debugging techniques**

Detection of Packer

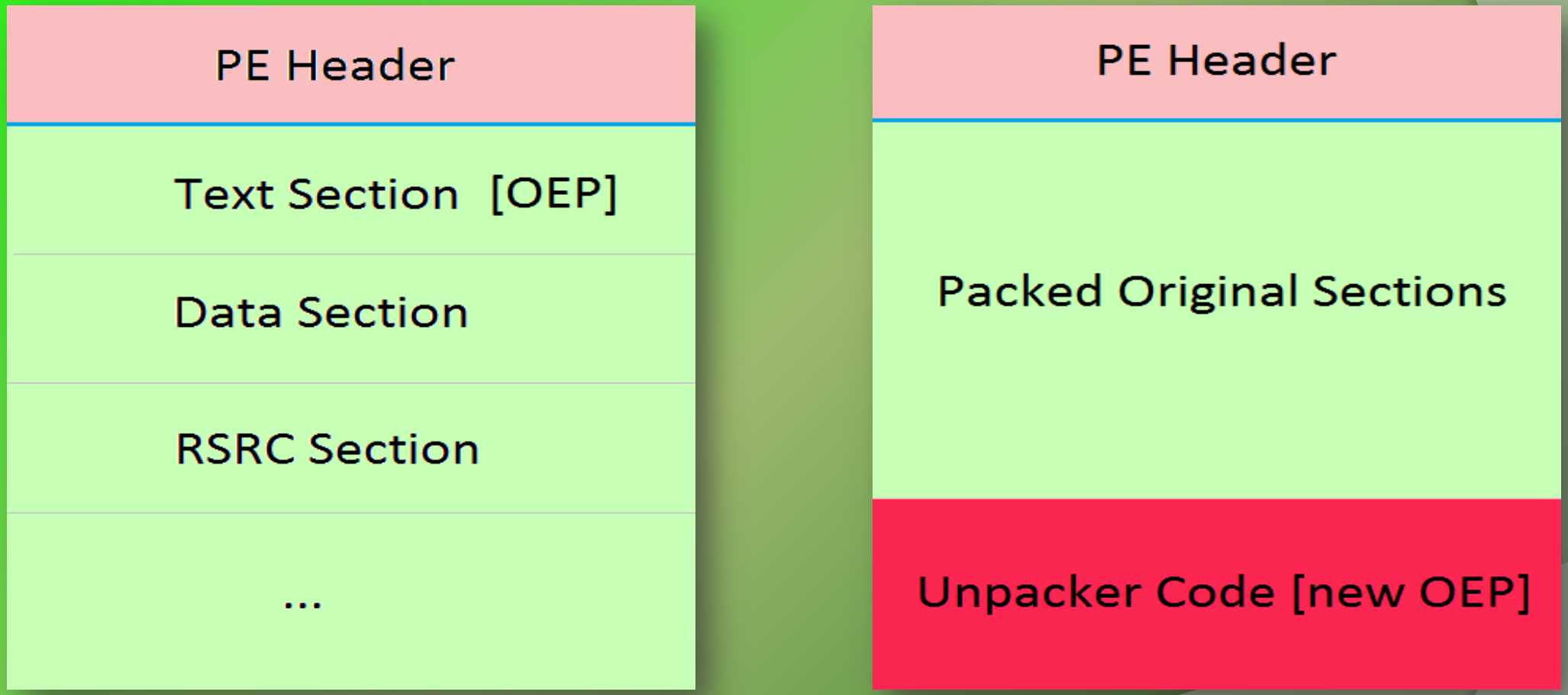
- **Packer Detectors like PEiD, RDG, ExeScan etc**
  - **Detect the popular Packers**
  - **Show the version of Packer also**
- **PE Viewer Tools like PEditor, PEview**
  - **Look at Section Table**
  - **Look at Import Table**

Packer Detectors





Structure of Packed EXE



Before Packing

After Packing

Execution of Packed EXE Program

- Execution starts from new OEP
- Saves the Register status using PUSHAD instruction
- All the Packed Sections are Unpacked in memory
- Resolve the import table of original executable file.
- Restore the original Register Status using POPAD instruction
- Finally Jumps to Original Entry point to begin the actual execution

Standard Process of Unpacking EXE

- Debug the EXE to find the real OEP (Original Entry Point)
- At OEP, Dump the fully Unpacked Program to Disk
- [?] Fix the Import Table using ImpRec Tool
- [?] Fix the PE Header

Unpacking UPX using OllyDbg

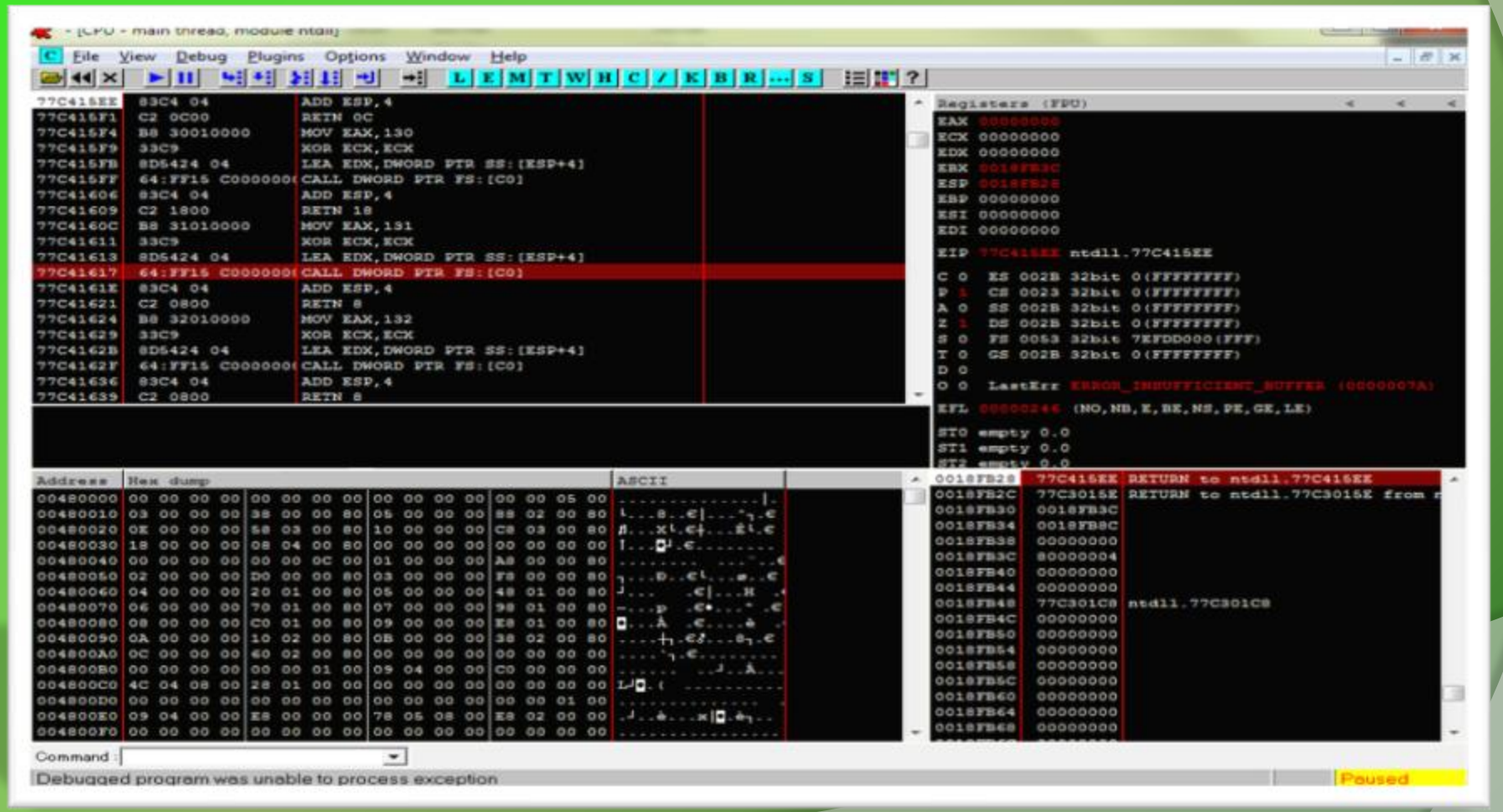
- Load the UPX packed EXE file into the OllyDbg
- Start tracing the EXE, until you encounter a PUSHAD instruction.
- At this stage, put the Hardware Breakpoint (type 'hr esp-4' at command bar) so as to stop at POPAD instruction.
- Other way is to manually search for POPAD (Opcode 61) instruction and then set Breakpoint on it.

Unpacking UPX using OllyDbg (contd)

- Next press F9 to continue the Execution.
- You will break on the instruction which is immediately after POPAD or on POPAD instruction [based on the method you have chosen]
- Now start tracing with F7 and soon you will encounter a JMP instruction which will Jump to OEP in the original program.
- At OEP, dump the whole program using OllyDmp plugin.

DEMO - Unpacking UPX

http://vimeo.com/42197903



Anti Anti-Debugging Plugins

Here are useful OllyDbg Plugins for Anti Anti-Debugging

- **Olly Advanced**
- **Hide Debugger**
- **NtGlobalFlag**
- **Anti Anti BPM**

Useful Tips

- Always use simple EXE for Unpacking exercises
- Use same EXE for all – You will know the OEP & other magic numbers
- Use Windows XP for better (less annoying) debugging experience.
- Have Patience, Its an Art and takes time.
- For best results, do it in the Moon Light ☺

What's Next?

- **Try Unpacking AsPack, AsProtect, PESpin, YodaP etc**
- **Try Unpacking Packed DLL (Google - Neolite DLL Unpacking)**
- **Try Advanced Packers: Armadillo ☺**

Reference

- Complete Reference Guide for Reversing & Malware Analysis Training

Thank You !

www.SecurityXploded.com