Reverse Engineering

Folmer Heikamp
Thijs van Ede
Anirudh Ekambaranathan

What and Why RE?

- Disassembling and decompiling
- Gain insight in program workflows
- Analysis of malware
- Vulnerability research

Dynamic vs Static analysis

- Executing code
- Debugger
- Disassembling binary files
- Decompiler

Static analysis methods

- Linear sweep
  - Prone to chain of errors
- Recursive traversal
  - Discovers dead code

Countermeasures to RE

- Dynamic analysis:
  - Anti debugging and emulation
  - Anti dumping through encryption
- Static analysis:
  - Obfuscation
  - E.g. optimisation, multithreading

Obfuscation Techniques

- Inserting junk bytes
- Introduce branch functions
  - Change the return address

Countering Obfuscation Techniques

- Create functions based on prolog
- Create CFG from functions
- Apply block conflict resolution
- Gap completion

- Result: significant improvement in disassembly

Questions?

References

C. Willems, C. Felix, F.A. Freiling. *Static Disassembly of Obfuscated Binaries*

C. Kruegel, W.Robertson, F. Valeur, G. Vigna. *Static Disassembly of Obfuscated Binaries*

J. Kinder, H. Veith. *Jakstab: A Static Analysis Platform for Binaries*

C. Linn, S. Debray. *Obfuscation of Executable Code to Improve Resistance to Static Disassembly*

U. Bayer, A. Moser, C. Kruegel, E. Kirda. *Dynamic analysis of malicious code*

J. Bergeron, M. Debbabi, M.M. Erhioui, B. Ktari. *Static Analysis of Binary Code to Isolate Malicious Behaviors*