Module A3: Forward Engineering

**Module Description:** Before students begin to reverse engineer software, it is vital that they have some understanding of how binaries are created, what influences their final states, and what piece they play in the larger context of a process (either as an executable or a library/DLL/shared object). The other major goal of this module is to give students a mechanism by which they can begin to map high level programming language concepts (e.g., a loop) into the various forms they can take in a binary, allowing them to more easily recognize common operations (e.g., loops, if statements, functions, variable assignments, arithmetic operations, and I/O) when they see them later in the reverse engineering modules.

**Prerequisite Knowledge:** This is intended to be a follow-up to Module A2: SRE Prerequisites - ARM architectures and assembly languages

**Length of Completion**: Module – More than 4 and less than 10 hours

**Level of Instruction:** This module intended to be an advanced undergraduate or graduate course for technical majors.

**Learning Setting:** This module is intended for in-class.

**Lab Environment:** VMs and containers that include the exercises and challenge materials to ensure consistency and ease of deployment.

**Lab Tasks:** The objective of this lab is to actively investigate a malicious binary to understand how binaries are built and the resulting structures in assembly language.

**Lab Files Needed:** All files are contained in the VMs.

# learning outcomes

MODULE learning oUTCOMES

Upon successful completion of this module, the student should be able to:

1. Explain how binaries are built (including preprocessing, compiling, assembling, and linking)
2. Recognize and differentiate between basic programming language structures in assembly language
3. Design and implement assembly language programs with structures. module Details

**Interconnection:** This module is part of a 15-week Software Reverse Engineering (SRE) Course. The 15 modules are:

* Module A1: x86 and x64 architectures and assembly languages
* Module A2: ARM architectures and assembly languages
* **Module A3: "Forward Engineering"**
* Module B1: Approaches
* Module C1: Techniques for the safe handling of files of unknown origin and /or functionality
* Module C2: Basic static analysis tools.
* Module C3: Disassemblers and Decompilers
* Module D1: Sandboxing and other techniques for the safe execution/opening of files of unknown origin and /or functionality.
* Module D2: Basic dynamic analysis tools.
* Module D3: Debuggers
* Module D4: Network Traffic Analysis
* Module D5: Patching binaries
* Module E1: Obfuscation and Anti-SRE
* Module F1: Non-Binary SRE

**Instructional Files and Online Resources Needed:**

PowerPoint file: SRE-A3-Lecture.pptx

Lab Environment: SRELNX-2.ova (Format may vary)

Lab Narrative: SRE-A3-Lab.docx

Supplementary Reading:

Sikorski, M. & Honig, A. (2012). *Practical Malware Analysis: A Hands-On Guide to Dissecting Malicious Software.* San Francisco:No Starch Press.

Eagle, C. (2011). *The IDA Pro Book: The Unofficial Guide to the World's Most Popular Disassembler (2nd ed.).* San Francisco:No Starch Press.

**Assessment:**

The learning objectives are assessed through the following methods:

1. ASKs: There are questions in the note sections for the instructor to involve the students in the lesson and assess their grasp of the concepts.
2. LABs: There are lab exercises associated with this lesson

# lessons

**Warm Up:** See SRE-A3-Lecture.pptx

**Lesson:** See SRE-A3-Lecture.pptx

**Active Learning Activity:** See SRE-A3-Lab.docx