FORTRA®

DATASHEET (OFFENSIVE SECURITY)

UDRL and Sleepmask Development

Author: Alex Reid
Levels: Certified
Study time: 9 hours

ZERO POINT SECURITY

This UDRL and Sleepmask Development course, created by Alex Reid and Zero-Point Security, teaches students how to apply low-level Windows knowledge and offensive tradecraft in the writing and development of Cobalt Strike's User-Defined Reflective Loader and Sleepmask components.

Training Content

Getting Started

- Welcome
- Author's Note
- Software Requirements

Introduction to UDRL and Sleepmask

- Defining the Problem
- Component Requirements
- Project setup
- Build Automation
- Testing Payloads

Extending Stardust

- Introduction to Stardust
- API Resolution Via Macros
- Debug Output
- Supporting Forwarded APIs
- Pointer Arithmetic
- Custom User Data
- Global Variables Without NtProtectVirtualMemory
- Removing Padding

Basic Reflective Loading

- Parsing PE Headers
- Mapping Sections
- Populating the Import Address Table
- Processing Relocations
- Transferring Execution

Sleepmask: Basic Ekko Implementation

- Integrating Common Assets
- Preparing sleep_mask
- Masking Heap Memory
- Understanding Ekko
- Implementing Ekko

Sleepmask: Through the BeaconGate

- Understanding the Problem
- Passing Stack Parameters
- Storing Return Values
- Integrating BeaconGate

UDRL: Module Stomping

- Introduction to Module Stomping
- Identifying Sacrifical DLLs
- Basic Implementation

UDRL: Advanced Module Stomping

- Background Research
- Manipulating Section Handles
- Replacing LoadLibraryEx
- Mapping Beacon's Unwind Info
- Finding and Fixing Sleepmask and BOF Unwind Info

Common: Control Flow Guard

- Understanding Control Flow Guard
- Enumerating CFG and Modifying the Bitmap

Evasion: Call Stack Spoofing

- Introduction to Hunt Sleeping Beacons and Call Stack Detection
- Exposing Timer Functionality to the UDRL
- Beacon Entry via NtContinue
- Concealing the Main Thread's Call Stack During Timer Execution
- Disguising Timer Stacks with ROP and JOP
- Suspicious Timers and Where to Hide Them

Evasion: Cleaning the LitterBox

- Introduction to LitterBox and Underlying Tools
- Patriot: Hiding Suspicious CONTEXTs
- Moneta: Freeing the Initial Allocation
- YARA: Addressing Static Signatures
- PE-sieve: Avoiding Entropy Checks

Evasion: Assorted Tradecraft and Code Cleanup

- Extending BeaconGate for Unsupported APIs
- Exception Handling via Wow64PrepareForException
- Exiting Beacon Gracefully
- Ekko via Threadpool Timers
- Code Cleanup and Bug Fixes

Course Completion Cleanup

- Areas for Future Exploration
- Closing
- Course Evaluation
- Certificate of Completion

FORTRA

Fortra.com

