kaspersky
expert training

Reverse engineering 101

Course program

№	Track	What you will learn	What you will practice	Lesson	Practice	Evaluation
0	Introduction	• About your trainers • Course roadmap • Course structure	—	Intro	—	—
				Intro To The GitHub Repository	—	—
1	Theory	• The basics of the assembly language	• Working with assembly instructions • Understanding function calls • Recognizing arguments	Introduction To ASM	—	—
				The most common ASM instructions	—	—
				Correspondence between C code and ASM	Reading ASM code	Quiz
				Function calls	—	—
				Calling conventions	Understanding a full function	Quiz
				Wrap-up	—	—
2	C-language “Hello World”	• How to approach reverse-engineering of programs written in C language	• Getting familiar with IDA and using the software to navigate inside assembly code • Applying theory learnt in the previous track	Introduction	—	—
				Simplest C Program	First steps with IDA	Quiz
				First steps with IDA		
				Working without debugging symbols		
				Windows API	Arguments and constants	Checkpoint Quiz
				Wrap-up	—	—

№	Track	What you will learn	What you will practice	Lesson	Practice	Evaluation
3	Simple Lambdas	• The difference between values and pointers • Transmitting the arguments by value and by reference • Old-style C-like memory management, where programmer is in charge of keeping proper references	• Reconstructing custom structures Transmitting the arguments • Understanding value and pointer fields in such structures • Practice embedded structures, where field is a pointer to another custom structure	Starting with value fields	Starting with value fields	Quiz
				Adding pointers	Now it's time to add pointers	Quiz
				Shallow and deep copying	Shallow and deep copying	Checkpoint Quiz
				Wrap-up	—	—
4	Stuck in the heap	• The difference between main places to store the data • Conception of stack and heap Automatic, dynamic and static memory • How the programmer's decision where to store data affects the resulting binary	• Creating massive custom structures on stack and heap • Analyzing them in resulting executable binary file to see the difference in these memory types • Reversing the code with static variables, understand their position in executables	Counting hash value	Counting hash value	Quiz
				Meet the heap	Meet the heap	Quiz
				Memory management	Memory management	Checkpoint Quiz
				Wrap-up	—	—
5	Lists and tricky pointers	• How custom C list looks on a binary level • How to handle lists with pointers that point to the middle of next element • Understanding custom data types, which you would met in binaries, further	• Continue to analyze structures, but a bit more complicated this time • Using simple custom list to move towards real C++ STL containers • Practice shifted pointers	Servers chain	Servers chain	Quiz
				List it till the end	List it till the end	Quiz
				Pointer to the insides	Pointer to the insides	Checkpoint Quiz
				Wrap-up	—	—

№	Track	What you will learn	What you will practice	Lesson	Practice	Evaluation
6	C++ And OOP	• How to reverse-engineer programs written in C++ language	• Recognizing C++ classes in assembly form • Experimenting with IDA's disassembler • Working with code coming from the C++ STL.	Introduction	A simple class	Quiz
				Virtual Function Table	C++ inheritance	Checkpoint Quiz
				The decompiler	—	—
				STL library: The string class	—	—
				STL library: The vector class. Wrap-up	—	—
7	Pain In The Containers	• How C++ STL containers look like in executables • How to analyze them in compiled programs, creating proper structures • Upon which basement std::map, std::set are build	• Inserting and searching for operations in std::map, std::set • The difference between map and set on binary code level • Practicing surface std::map, set, pair analysis in binaries	Commands as dictionary	Commands as dictionary	Quiz
				Real Handlers	Real handlers	Quiz
				Make It Set	Make it set	Checkpoint Quiz
				Wrap-up	—	—
8	Introduction to Golang Reverse-engineering	• The basics of the Go language • How to approach reverse-engineering when faced with binaries generated by its compiler	• Using a debugger in order to easily obtain program arguments and return values	Introduction	Golang basics	Quiz
				Debuggers	The decryptor	Checkpoint Quiz
				Working with x64dbg		—
				Reconstructing Go code from the assembly		—
				Controlling execution with debugger		—
				Go Reverse-engineering methodology	—	—

№	Track	What you will learn	What you will practice	Lesson	Practice	Evaluation
9	‘Rusty’ Code	—	• Understanding non-stripped Rust code • Dividing runtime and custom Rust code • Demangling Rust function names	Simplest Esoteric Decryptor	Simplest esoteric decryptor	Quiz
				Vector And Deep Copy	Vector and deep copy	Quiz
				Let’s Be More ‘Rusty’	Let’s be more ‘Rusty’	Checkpoint Quiz
				Wrap-up	—	—
10	A “Real” Malware	• How to analyze a full infection chain on your own	• All tools and knowledge gathered up to now in the course: C, C++, debugging, etc.	Introduction	Stage 1	Quiz
					Stage 2	Quiz
					Stage 3	Checkpoint Quiz
				Outro. Course summary	—	—

Thank you!

kaspersky.com

Discord server: kas.pr/g2j8

Help page: kas.pr/ii9f

kaspersky