

## Forum:[Hack Forums](#)

Thread: [Prerequisites to learn Rootkit Arsenal?](#)

**mr.cracker** - April 20, 2016, 10:06 am

I know c, shellcoding and windows console based programming using API's.

I want to learn Rootkit Arsenal. It has a lots of complicated subjects.  
What should I know before go into it?

Do I wanna learn how to program device drivers?  
Does this book is fine - "Developing Drivers with the Windows Driver Foundation"  
Do I wanna read windows Internals?

Can anyone give me some good suggestions please?

**mods\_3** - May 9, 2016, 5:39 pm

Learning to write professional rootkits is a privilege few of us have. Its main goal is monitor activities and to conceal its presence and maybe other processes, file/folders, registry, just to name a few.

That gives you a lot of power, but with power comes responsibility. Be wise. Rootkits are usually used by big firms on their machines to enforce corporate digital policies.

There are three basic types of rootkits:  
Kernel Mode Rootkit

User-Mode Rootkit

MBR (or Bootkit)

The book "Rootkit Arsenal" is by far one of the best choices (by experience). It educates you from A..Z academically and practically (by examples) but only for User-Mode rootkits on x86 machines. It does not cover bypassing techniques of Patch Guard on x64 systems. However, the assembly will have to be slightly modified to accommodate and port the code to x64.

As currently reading the book, it states that the prerequisites are C (programming language) and Intel Assembly. That is it, a little Win32 API is a plus.

Developing rootkits is a new realm, like learning to learn for Macintosh and using Objective-C and libraries. So, no other specific requirements are needed.

Just read the book VERY carefully. Do not try to copy/paste examples and wrap your App up. Learn and read, and only then, I promise you will be writing professional-resilient and stable rootkits soon enough :)

Just remember, the way you would like to go is really deep. You need to have time, which I consider another prerequisite.

Two last things:

Rootkits are not malware in nature. It's a tool that can be used for either good or evil.

Some Material:

For learning C, there are tons of material to learn from.

For Assembly language, I personally recommend the book "Art Of Assembly" (last time I checked, it was freely available online).

Hope that helps! Cheers!

**mr.cracker** - May 10, 2016, 1:13 am

Wow! You are just amazing. I got the answer which i needed. Thanks a lot.

One more question,

So we dont want to learn driver development to master in rootkits. Is it?

**mods\_3** - May 10, 2016, 2:19 pm

I think you still haven't got the big picture here. Writing code for a kernel driver utilizes only your skills with the framework (WinDDK). You have to learn the framework to do something useful, like a rootkit.

Rootkits are basically drivers (applications, if you wish) that use subversion techniques using by manipulating undocumented or very poorly documented functions and structures and then hooking key functions in Kernel for concealment.

For instance, you know the syntax of C and want to learn Java, the single most important thing is to learn the JDK, plus adjust some coding habits with a slightly changed syntax.

When coding in Kernel Space, all you have to learn is the framework (WinDDK--analogous to JDK) and C (analogous to C->Java syntax).

The good new is... the book "Rootkit Arsenal" contains all the material needed (from a malicious POV), starting from history, then by WinDDK development, Hardware architecture (Intel x86 only), and rootkits: how they work and everything about them. It also contains real-world implementation, in a step-by-step approach.

Here you have got the advantage of already knowing C.

A note: You WILL need to dig deep into Windows internals and undocumented functions and features. There is good material for that. For examples here is a list of good ones:

The Definitive Native API Guide

Microsoft Windows Internals

Introduction to NT Internals

This is, of course, to be able to use the undocumented information to hide processes, registry entries, files/folders, you name it.

If your code is running in Kernel space, you basically own that machine, especially if you were writing Kernel-Mode drivers.

Also, google, you may/will find updated things. Plus there is a dedicated undocumented APIs.  
Sorry, can't post links yet. I really needed some rep+ :)

Lastly, reading about reversed malware can be pretty much informative too. For example, google "Ouroboros - The Snake Rootkit.pdf". It was professionally analyzed and I have learned new techniques and stuff from it.

Hope that helps.

MoDs

Signing out...

**mr.cracker** - May 10, 2016, 11:32 pm

@mods\_3. Thanks a lot lot...this is very much informative.. Am gonna start it...

**Root Kit** - August 8, 2016, 3:54 pm

Thanks for this too.

:D :D