

Forum:[Hack Forums](#)

Thread: [how to reverse engineer video games?](#)

Phantom_NiNja - August 26, 2026, 5:04 pm

I am trying to learn how to reverse engineer programs and games.

I am working on airline tycoon and theme hospital.

However, some people online claimed to have done it.

I looked into them and it looks like all they did stole the game assets.
then just rewrote it from scratch.

I am trying to reverse engineer the games. I got a disassembler debugger.
but need to know what to use to watch what values get stored and passed around.
what's a good tool to do this and then make patches to the code?

I just want to learn for educational purposes.

Terry A. Davis - August 27, 2026, 3:54 pm

First off, I think this would've been better in the RE section.

Now that is out of the way. This might shock you, but Cheat Engine can be very helpful here.
Figure out if the game uses ASLR - that will stunt things, not make it impossible, but make it a bit harder, by any metric of harder.

You can use Cheat Engine to find values, and track what reads/modifies them, then from there, you can take that address into something like IDA.

Can you articulate what more you want to do, how much you know about RE, and everything of that nature? I don't particularly want to hold your hand since my 'tism has already got me going on 2 RE projects, but I can happily share what I know, and explain my knowledge to help you accomplish what you might want to do.

PELock - August 27, 2026, 4:35 pm

Start with good tools <https://www.pelock.com/articles/reverse-engineering-tools-review>

Terry A. Davis - August 29, 2026, 11:31 pm

Since you've made other threads, and not returned to this one, I presume you've lost interest?

Phantom_NiNja - August 30, 2026, 2:45 pm

(Aug 29, 2026 11:31 PM)Terry A. Davis Wrote: [\[CITE\]](#) Since you've made other threads, and not returned to this one, I presume you've lost interest?

Nope, I still have interest on how to reverse the game.

I used x32dbg and x64dbg disassemblers.

I just want to learn the concept and what people do to reverse engineer fully.

I tried to do this with simcity 4 rush hour and airline tycoon. The simcity 4 is hard because so many things are obfuscated

the airline tycoon the strings are not obfuscated. You can easily see only system functions like functions from windows api.

other than that the game functions seem to be from other libraries and don't know how I can see them.

I am guessing I have to disassemble them right?

I just want to know what is best practice and is there way I can run the program like IDA another paid disassembler?

I want to use free tools.

I want to run the game in the x32dbg and watch the code and registers change. I think it's better to watch the program start and flow thru.

but don't know if it calls a function will it hop to the dll code?

I am new to reverse engineering but I have used cheat engine and do know I can scan the memory and change numbers.

I know how to jump from logic that checks if you have a legit key or copy.

I know the basics.. do know the keygen how to make them but it requires you to reverse the code that generates the keys.

how ever I want to be able to watch the program run step by step and read the values to figure out what's going on.

Most functions are not names it's normally hex values in memory locations that gets loaded pointers and sometimes put on the stack.

I know assembly language. in x86. I just want to know if there's a way I can run the code without having it run on my system.

Just want to watch each step of line of code gets executed.

I currently been looking at things statically. I want to learn how to do this the right way. maybe read a book or if there's tutorials that makes it easier to understand.

Terry A. Davis - August 30, 2026, 2:57 pm

Quote:Nope, I still have interest on how to reverse the game.

I used x32dbg and x64dbg disassemblers.

They're the same application for different [architecture](#), if you're dealing with modern games, you'll mostly be using x64dbg. So for now, we'll simply call it x64dbg, we'll differentiate only when needed to do so explicitly.

Quote:I just want to learn the concept and what people do to reverse engineer fully.

I tried to do this with simcity 4 rush hour and airline tycoon. The simcity 4 is hard because so many things are obfuscated

I would suggest not doing it to AAA games for that exact reason, obfuscation makes things very difficult, because not only are you fighting what the compiler thought to do, you need to fight the

obfuscation to work out what it is originally doing. Maybe pickup the non-modern version of Windows Calculator for example, and see if you can find the calculation functions, and work backwards from there to see if you can hook how it sets the Windows Title and set your own.

Quote:the airline tycoon the strings are not obfuscated. You can easily see only system functions like functions from windows api.

other than that the game functions seem to be from other libraries and don't know how I can see them.

I am guessing I have to disassemble them right?

You do typically have to do that, yes. Since DLLs are dynamic libraries, they need to be disassembled separately, but unless what you're after is in those, you should treat them as the black box they are, if the code is not in the target EXE, it is not of your concern unless it is what you're looking for, E.G. If you wanted to change a game's MP system away from SteamWorks you would look at where the EXE sets it all up and change it there, not inside the SteamAPI DLL. Static libraries are different, they get included in the EXE and are undisguisable from normal code unless they're something very common, then you might notice patterns (Look into FLIRT signatures)

Quote:I just want to know what is best practice and is there way I can run the program like IDA another paid disassembler?

I want to use free tools.

IDA is free if you know where to look ?. It is the premier tool, some people like Ghidra, but I never really liked it much, the Java interface always through me off. IDA / Ghidra do have features to launch through a debugger, but mostly a static tool for viewing and parsing through, they let you reconstruct a fair bit of the programs structure and logic, but obviously you'll never get back to original source code.

Quote:I want to run the game in the x32dbg and watch the code and registers change. I think it's better to watch the program start and flow thru.

but don't know if it calls a function will it hop to the dll code?

It should do yes, I don't have x64dbg in front of me now, but it should say in the titlebar which module it is currently executing from, so if you load the EXE then launch it, I believe you start typically in ntdll, then as you run through it will take you to the EXE's entry point and such and so forth.

Quote:I am new to reverse engineering but I have used cheat engine and do know I can scan the memory and change numbers.

I know how to jump from logic that checks if you have a legit key or copy.

I know the basics.. do know the keygen how to make them but it requires you to reverse the code that generates the keys.

how ever I want to be able to watch the program run step by step and read the values to figure out what's going on.

That is what a debugger is for, much like code you write, you set the breakpoints where you are, and see registers to watch what changes, at the lowest level variables are just registers, so they will change and set depending on what is going on, maybe learn a bit about assembly and how that all works at a high level so you are better prepared to see what you expect in the disassembler, or watch windows etc.

Quote:Most functions are not names it's normally hex values in memory locations that gets loaded pointers and sometimes put on the stack.

I know assembly language. in x86. I just want to know if there's a way I can run the code without

having it run on my system.

Just want to watch each step of line of code gets executed.

No, that would be static analysis, you can step through it yourself manually, but dynamic is always better since you see the flow as expected, use breakpoints as your friend.

Quote:I currently been looking at things statically. I want to learn how to do this the right way. maybe read a book or if there's tutorials that makes it easier to understand.

I think you'll learn more just farting around.

<https://www.penguin.co.nz/books/the-ida-pro-book-2nd-edition-9781593273958>

Is one I thumbed through that oddly helped a bit.

Find a very simple project, like I said, Calculator, see if you can find where it sets the title, and hook it to something you want instead.

Don't use the modern WinApp version, I am talking vintage calculator.

Phantom_NiNja - August 30, 2026, 3:06 pm

(Aug 27, 2026 03:54 PM)Terry A. Davis Wrote: [\[CITE\]](#) First off, I think this would've been better in the RE section.

Now that is out of the way. This might shock you, but Cheat Engine can be very helpful here.

Figure out if the game uses ASLR - that will stunt things, not make it impossible, but make it a bit harder, by any metric of harder.

You can use Cheat Engine to find values, and track what reads/modifies them, then from there, you can take that address into something like IDA.

Can you articulate what more you want to do, how much you know about RE, and everything of that nature? I don't particularly want to hold your hand since my 'tism has already got me going on 2 RE projects, but I can happily share what I know, and explain my knowledge to help you accomplish what you might want to do.

I have used cheat engine for a long time. I don't think it's a good too to reverse engineer games. I feel it's good in changing values in the game while it runs.

I never was an advance user where making scripts for it. I only used it to change money values for airline tycoon and simcity 4 rush hour.

I want to run a game without running it on my computer. Meaning just want a tool to execute the code and show line by line what is going on.

If it uses dlls it should jump over to that code.

Right now I know basics. I know assembly code. I written a basic OS in assembly and some programs in it.

I just want to get used to using a disassembler to learn how programs work. Right now using x32dbg and I can read the .exe file

but I haven't got it to run a program. I know it has the functionality to it but tried it a few times and my computer just overheated and crashed.

I got a new CPU. So, don't know if it's going to give me problems.

You don't have to hold my hand. I just want to know how people do it. I know about IDA but don't want to spend the money.

Most people use that. I know about ghidra.

I just want to know how to reverse engineer programs of today's. obfuscated programs. I already know how cracked programs are patch.

I done fake examples to do it. Those are really easy. It's mainly reading the logic. It's typeing in a wrong key and getting a string that says error or sorry wrong key. You take that string and used tools like what I use to search for the string in the strings section and then you look around that code at the assembly to figure out the logical jumps. You just patch it by adding in a jump to the hex value of the either the function or the label where after you put the correct key it takes you too.

I know the easy basic. The easy stuff. I want to learn how to people tackle down the obfuscated harder programs. Or if code is encrypted

and gets decrypted on the fly. I want to learn how to reverse engineer programs of today. I know it's hard and isn't easy but want to know the methods.

what do people look for in the code that gives them hints as to what encrypting they're using and how that encryption gets decrypted.

all I know is that for obfuscated stuff you need to find the code that deobfuscates the code at run time. for encryption you need to find the code that decrypts the code at run time. I know this is the goal. But it's easier said than done.

even when games used online servers. Hackers found ways to create their own programs to act like the servers and bypass all the checks and executes the code.

I just want to know what I should do to tackle reading the code of airline tycoon and how to reverse engineer it.

I want to make my own mods to the game. I always wanted to do this as a kid.

I think by doing this it will help me better with writing my own assembly code.

I want to do this just to learn. I been reading books but the books get you so far and won't show you latest concepts or what the industry is using.

Normally the industry uses the latest stuff and then years later you get books publish talking about how it works. By then the industry is using something else. You cannot learn latest stuff. It's always stuff that becomes outdated.

like QR codes we been using it for years There's lots of tutorials about it and lots of program that can make it. In 2030 it's going to replace the bar codes.

However, QR is old tech. Now it's going to be an industry standard in 2030 so there's books on it and other materials on it that anyone can use it.

My point is that you need to reverse engineer to learn the latest tricks people are using in the industry.

Terry A. Davis - August 30, 2026, 3:30 pm

Quote:I have used cheat engine for a long time. I don't think it's a good too to reverse engineer games. I feel it's good in changing values in the game while it runs.

I never was an advance user where making scripts for it. I only used it to change money values for airline tycoon and simcity 4 rush hour.

I want to run a game without running it on my computer. Meaning just want a tool to execute the code and show line by line what is going on.

If it uses dlls it should jump over to that code.

Right now I know basics. I know assembly code. I written a basic OS in assembly and some programs in it.

If you truly have written your own OS in ASM, you should know how to do what you're asking, or have an intuition to do it if you didn't write code it.

Quote:I just want to get used to using a disassembler to learn how programs work. Right now using x32dbg and I can read the .exe file but I haven't got it to run a program. I know it has the functionality to it but tried it a few times and my computer just overheated and crashed. I got a new CPU. So, don't know if it's going to give me problems. Not sure how that would've happened with what you asked it to do, disassembly isn't that resource intensive since it's just deciphering hex into instructions. So you might have had more underlying issues than just this.

Quote:You don't have to hold my hand. I just want to know how people do it. I know about IDA but don't want to spend the money. Most people use that. I know about ghidra.

I just want to know how to reverse engineer programs of today's. obfuscated programs. I already know how cracked programs are patched. I done fake examples to do it. Those are really easy. It's mainly reading the logic. It's typing in a wrong key and getting a string that says error or sorry wrong key. You take that string and used tools like what I use to search for the string in the strings section and then you look around that code at the assembly to figure out the logical jumps. You just patch it by adding in a jump to the hex value of the either the function or the label where after you put the correct key it takes you too. That is exactly what cracking is. For obfuscation there is no golden key, you need to work out all the tricks it uses, and counter it, no special tricks to teach.

Quote:I know the easy basic. The easy stuff. I want to learn how to people tackle down the obfuscated harder programs. Or if code is encrypted and gets decrypted on the fly. I want to learn how to reverse engineer programs of today. I know it's hard and isn't easy but want to know the methods. what do people look for in the code that gives them hints as to what encrypting they're using and how that encryption gets decrypted. You run it, set breakpoints, and work through it one line at a time, there is no golden key unless the obfuscation is already cracked by people.

Quote:all I know is that for obfuscated stuff you need to find the code that deobfuscates the code at run time. for encryption you need to find the code that decrypts the code at run time. I know this is the goal. But it's easier said than done.

even when games used online servers. Hackers found ways to create their own programs to act like the servers and bypass all the checks and executes the code. Yes - they do that by looking at expected behavior and mirroring the logic, there is no simple this is how you do it, since stuff like that is hard. You need to look at how the game handles inbound outbound stuff, and work backwards, to see what you can understand from there.

Quote:I just want to know what I should do to tackle reading the code of airline tycoon and how to reverse engineer it.

I want to make my own mods to the game. I always wanted to do this as a kid.

I think by doing this it will help me better with writing my own assembly code.

You said you made your own OS, even basic, which means you know how it works, though I figure you're being a bit ... let's just say bold. So you should know what you need to do, start with working out the file format the game uses to load content, find functions that load said content, and work backwards of how you'd make your own thing to do it, to decipher obfuscated code, you'll need to find how the game does it internally, since ultimately even obfuscated code can be defeated, without knowing what tool they did to do it, you're out of luck for any silver bullet.

Quote:I want to do this just to learn. I been reading books but the books get you so far and won't show you latest concepts or what the industry is using.

Normally the industry uses the latest stuff and then years later you get books publish talking about how it works. By then the industry is using something else. You cannot learn latest stuff. It's always stuff that becomes outdated.

Building blocks teach you more than the results, there is nothing for what the industry teaches, you need to have that understanding of how things work at the low level to see how building block stuff was built on-top of it.

Quote:like QR codes we been using it for years There's lots of tutorials about it and lots of program that can make it. In 2030 it's going to replace the bar codes.

However, QR is old tech. Now it's going to be an industry standard in 2030 so there's books on it and other materials on it that anyone can use it.

My point is that you need to reverse engineer to learn the latest tricks people are using in the industry.

Which is only done by doing.

Phantom_NiNja - August 30, 2026, 5:04 pm

(Aug 30, 2026 03:30 PM)Terry A. Davis Wrote: [\[CITE\]](#) If you truly have written your own OS in ASM, you should know how to do what you're asking, or have an intuition to do it if you didn't vibe code it.

Not sure how that would've happened with what you asked it to do, disassembly isn't that resource intensive since it's just deciphering hex into instructions. So you might have had more underlying issues than just this.

That is exactly what cracking is. For obfuscation there is no golden key, you need to work out all the tricks it uses, and counter it, no special tricks to teach.

You run it, set breakpoints, and work through it one line at a time, there is no golden key unless the obfuscation is already cracked by people.

Yes - they do that by looking at expected behavior and mirroring the logic, there is no simple this is how you do it, since stuff like that is hard. You need to look at how the game handles inbound outbound stuff, and work backwards, to see what you can understand from there.

You said you made your own OS, even basic, which means you know how it works, though I figure you're being a bit ... let's just say bold. So you should know what you need to do, start with working out the file format the game uses to load content, find functions that load said content, and work backwards of how you'd make your own thing to do it, to decipher obfuscated code, you'll need to find how the game does it internally, since ultimately even obfuscated code can be defeated, without knowing what tool they did to do it, you're out of luck for any silver bullet.

Building blocks teach you more than the results, there is nothing for what the industry teaches, you

need to have that understanding of how things work at the low level to see how building block stuff was built on-top of it.

Which is only done by doing.
I think I get it now.

No, writing your own OS is easier than reverse engineering a program. Why? what you just said is like saying why can't you reverse engineer the program. Just email the programmer and ask for the source code.

You wouldn't need to reverse engineer programs if they handed you the source code. Just because I know assembly and know how to write programs. Doesn't mean I should be able to reverse engineer easily. It doesn't work that way. I am asking for how people tackle such problems.

Normally people would advice work doing x y c first. I watched a video where he said step one is look at the strings. He said what he does instead of reading the code he said to look at the strings for hints. He used an old program where the strings were not obfuscated.

He ran an example showing how to write your own keygen but the example is easy because in the strings you can figure out where the key gets generated. He explains that software companies make keygens to assign a key to each copy. So, the example was a fake program. it would take program files burn it to cd-rom before it burns it. It used a key generator to make a key and then encrypt it and embed it into the .exe file then it had a function and a key used to decrypt it. It would store it in some dll file. The guy showed to search the strings for clues. This narrow down where to look for the code we need.

he would run the program and find the function that just makes the key. He copy's that code and then decrypts the key.

He then would take the key and use various standard encryption methods. He would then encrypt it and see the results to see if it matched what he saw before the decryption. It matched and he would then use both to make a keygen program. It was an easy fake program to just demonstration what you need to do.

My point seeing various videos on it people have their own method on talking such a project. That guy says first check the strings to see if it gives you clue. He gives a check list on where to look next. Yet, most videos only show those methods works well if you already know how the program works or if it's a fake program it would deliverable be made easily to reverse. They tell you that real current programs do a lot of various obfuscation methods to make it a lot harder and they said could take months or years to be able to do it.

just because I know how to make a program in ASM and an OS doesn't mean reverse engineering is easy for me. I am trying to learn how to tackle it.

Since, reversing a game has a lot of code. It's not an easy task. It's a lot easier to write your own programs than it is to look at machine code that is heavily obfuscated to reverse engineer it and succeed.

Terry A. Davis - August 31, 2026, 5:27 pm

Quote:I think I get it now.

No, writing your own OS is easier than reverse engineering a program. Why? what you just said is like saying why can't you reverse engineer the program. Just email the programmer and ask for the source code.

Respectfully disagree - though I guess its your interpretation of what you consider an OS, writing an ASM program that puts your CPU into real mode isn't that hard, but having something multiuser ringed, and some nice to haves like busy box or a C compiler is hard.

Quote:You wouldn't need to reverse engineer programs if they handed you the source code. Just because I know assembly and know how to write programs. Doesn't mean I should be able to reserve engineer easily. It doesn't work that way. I am asking for how people tackle such problems.

Normally people would advice work doing x y c first. I watched a video where he said step one is look at the strings. He said what he does instead of reading the code he said to look at the strings for hints. He used an old progarm where the strings were not obfuscated.

You're correct, though one lends itself to the other, you know enough ASM to know how to do to things, in RE there is no one solution for all, just because I can look at programs I RE often doesn't mean I can do the same elsewhere, and especially for obfuscation, you're not going to find an easy way to undo it since that's the whole point in it.

Quote:He ran an example showing how to write your own keygen but the example is easy because in the strings you can figure out where the key gets generated. He explains that software companies make keygens to assign a key to each copy. So, the example was a fake program. it would take program files burn it to cd-rom before it burns it. It used a key generator to make a key and then encrypt it and embed it into the .exe file then it had a function and a key used to decrypt it. It would store it in some dll file. The guy showed to search the strings for clues. This narrow down where to look for the code we need.

he would run the program and find the function that just makes the key. He copy's that code and then decrypts the key.

He then would take the key and use various standard encryption methods. He would then encrypt it and see the results to see if it matched what he saw before the decryption. It matched and he would then use both to make a keygen program. It was an easy fake program to just demonstration what you need to do.

My point seeing various videos on it people have their own method on talking such a project. That guy says first check the strings to see if it gives you clue. He gives a check list on where to look next. Yet, most videos only show those methods works well if you already know how the program works or if it's a fake program it would deliverable be made easily to reverse. They tell you that real current programs do a lot of various obfuscation methods to make it a lot harder and they said could take months or years to be able to do it.

That is what RE is, you need to work it out, that's the whole point in it. It can certainly take months to years, because compilers, obfuscators, people all play roles in making it difficult to do what we want to do via this, it's a consequence of the nature of how things are designed I'm afraid.

Quote:just because I know how to make a program in ASM and an OS doesn't mean reverse engineering is easy for me. I am trying to learn how to tackle it.

Since, reversing a game has a lot of code. It's not an easy task. It's a lot easier to write your own programs than it is to look at machine code that is heavily obfuscated to reverse engineer it and succeed.

You use tools you are using, and work through functions one at a time, find things, work things out, document, test, evaluate, and continue. There is no magic button a disassembler can click to do it for you, since most of the time it needs to be ran through once to work it all out. Ergo, use a debugger, step through the program, set breakpoints where you think things happen, trigger them,

evaluate, and so on.

The strings approach is valid, but not always the way, sometimes just looking at cmp instructions is what you need to do to find what you're looking for, sucks horribly, but if the program is obfuscated, maybe look at a high level what it is doing to get to the original code so you can try unpack it. Or look via known obfuscation checkers to see, maybe also don't start on a hard project so you don't get discouraged.