# 写在前面

这是一篇有关逆向能力学习的前期准备的文章。

先解释下逆向这个词，它指对编译后的代码（机器码或者字节码等）进行分析，理解含义的操作。破解主要就是靠逆向的能力，下文不区分两者。

为了让读者有个心理准备，大概说下学习可能花费的时间：按照每天学习2个小时计算，从零开始学C语言到满足我要求估计要花至少3周以上，C++可能要花同样的时间（非必需）；接下来汇编语言与底层环境的了解估计2周；调试器的学习时间较短忽略不计；最后Windows程序设计估计4周。要注意的是我说的这个学习包括了上机调试的时间，同时还不能把每一个例程全都试一遍，那样时间也不够。我假设读者的悟性较好，知道在学习过程中哪些东西较为困难，把自己难以理解的东西放到实机上编写、试验，最后领会。还有就是即使每天学习时间有6个小时，也不是说把总的天数除以3就可以的，记不住、领会不了的东西学过了也是白搭。

在这些内容的学习过程中，我不会回答任何关于环境配置等方面的问题——如果连学习环境都没办法自己搭建好，还是算了吧。学完后的自测题如果实在无法理解，可以问我。不过题目自己做会了就好，不用给我看。

文中使用的书籍如果你有财力可以去买正版，否则都可以从网上下载到。

# 解决问题的能力

作为一个技术人员，这是最基本、最重要的能力。不管是分析别人的程序还是自己写程序，总会遇到问题。问题发生之后怎么去理解，从哪个方向努力，采用何种手段去解决，只有掌握了这些才可以有进一步的作为。

理解问题需要经验，只有遇到问题多了之后才能快速的理解一个新问题的性质、分类，继而影响到努力的方向。而关于解决手段，具体到程序员身上，大概包括以下几个方面：

* **工具的使用**。这通常是解决问题的第一步，尝试使用熟悉的工具检查问题所在，比如写代码时你可能需要对大量的代码文件进行搜索、查阅，逆向时可能需要使用WinHex之类的东西查看未被调试程序的内存等等。
* **搜索引擎的使用**。当你发现这个问题对你来说是一个全新的、未知的问题时，通常就需要使用搜索引擎了。你要从问题中提取出合适的关键字交给搜索引擎，并从它返回给你的大量无用页面中找出有用的那些，进去学习。
* **论坛、社交网站的使用**。而当你发现这个问题即使对于搜索引擎也是一个全新的问题时（这大部分时候都是由于你没有正确的提取关键字，或者漏过了正确的结果网页），就只能求助于他人了。各种编程、破解方面的论坛，社交网站都可以作为你提问的渠道。而这种时候，你所需要的就是描述问题的能力，把你的问题清晰、简洁的描述出来，呈现给网线另一端的大神们。

以上是一些概括性的说明，下面说几点心得：

* 搜索前，提取问题的关键字的时候，不要带上特化的东西。比如编译器提示了一个错误，先尝试只搜索错误代号，不行再加上后面的通用描述。和你代码本身有关的内容别加进去。
* google比百度好用。所以你可能需要翻墙的能力。如果没翻过墙的话，只提示一个词，goagent。
* 如前所述，如果搜索引擎搜不到你的问题答案，很有可能（特别是对于初学者）是你关键字提取的不好，多换几次试试。
* 编程方面，英文资料通常比中文的要丰富。如果中文找不到，尝试用英文搜索。
* 破解方面，国内的资料通常更给力一些，主要集中在一些相关的论坛，bbs.pediy.com，www.unpack.cn等。在这些论坛里面注册个帐号，没事刷刷存在感，多向别人学习。
* 提问的话，编程方面推荐StackOverflow，提问前请仔细阅读说明。

就说这些吧，解决问题的能力更多的还是在实践中磨练出来的。最后，如果你基础不够，本篇学习指导本身可能就有你看不懂的内容，请使用搜索引擎或者论坛、SNS。如果还是没办法理解的话，还是放弃吧。

# 基础能力

就是一些使用电脑的基本能力，首先dos命令要了解一些，运行程序时加个/? –help –h之类的参数调出帮助等。然后一个图形界面软件运行之后，怎么去发掘它的各种功能，这也是基本，方法其实就是看顶上的菜单，看右键弹出菜单等等。F1是帮助快捷键应该都知道吧，虽然现在很多新程序都不喜欢用这个了。键盘操作要熟练，写代码的时候基本都是靠键盘，包括调整光标、设置缩进之类的，老去摸鼠标严重降低效率。

# 语言基础

虽然说的是编程语言，但是还是必须强调一点，作为一个程序员，英语是最基本的能力。听、说、写先放一边，读的能力必须非常好。因为你需要经常阅读各种英文的手册、文档、资料。涉及到galgame的话，日语有些时候也会用到。

搞逆向虽然理论上只需要汇编语言基础，但是因为我们逆向的程序通常并不是由汇编直接写出来的，大多是通过C/C++编译后的，所以掌握这门语言还是比较有用处的。C语言的学习参考书太多了，而且事实上光看书也别想真正掌握，这里就不推荐某本特定的书了（我自己用的是谭浩强的《C程序设计》，感觉还不错），总之学到能跟着例程写个几百行的代码并且自己能看懂就好。C++的话，时间充裕可以学一下，虽然有用，但是太复杂了，学习周期会很长。

接下来就是汇编语言。因为汇编语言事实上内容非常少，所以随便找本书学一下就好，最好是32位的，16位的也没关系。目标是了解计算机的底层环境与所有常用指令的含义。屏幕显示、输入控制相关的部分不需要学，windows上用不到。

那么来定一下学习目标吧，在学习完成后能完成以下问题或任务就算合格了。如果你学习完后纯靠自己就能解决，算优秀；如果要靠搜索引擎，算良好；如果要靠问别人，只算及格。当然评分没什么价值，自己知道就好。

## 自测题

如果你是用VC、BC等图形界面的工具学习的，请在命令行下面分别编译一个调试版和发行版的hello world。

为什么.h文件中通常不定义变量？（在这之前请先理解定义和声明的区别）如果这么做可能会导致什么样的错误？怎么解决？

typedef int BOOL;和#define BOOL int有区别吗？（若有的话有什么区别？下同）

给定一个struct，在不将其定义出来的前提下，请打印出其中某个域所在结构中的偏移（不准自己手算）。

某程序只有一个类，该类只有一个方法，该方法是不是虚函数有区别吗？（C++题目）

十六进制数怎么转换成二进制数？

请解释x86 cpu的8个通用寄存器一般分别作什么用处。

jnz和jne指令有区别吗？test和and指令呢？

分别解释push、call和ret指令对内存和寄存器做了哪些修改。

# 调试

这是我们逆向分析的基础。虽然逆向分析分为动态和静态两种，但是静态分析由于无法获取程序运行状态，而现有的计算机程序又极大地依赖于运行状态，所以静态分析所能获取的信息远不如动态分析。

动态分析一般就是调试。使用OllyDbg、WinDbg等调试器为基础。不过这里我们要学习的是普通的源代码调试，也就是说调试你自己写的程序。这里推荐使用VC自带的调试器。当然你要用gcc、WinDbg什么的也无所谓。

调试器的用法的话，随便一本讲Vistual C++的书上都会讲到，事实上我也没有用书学过，所以也没法推荐什么。甚至你可以按照我下面自测题的内容直接从网上搜索，全部弄清之后也就差不多学会了。

## 自测题

首先在你代码某行下个断点断下来。

查看某个局部变量的地址，并在内存中查看（打开内存窗口查看）。

下条件断点并断下来。

直接执行到在这之后的某一行（不是单步执行过去，而是命令调试器直接“执行到这一行”）。

使用监视变量的功能。

查看调用栈。

修改某个变量的值。

修改下一行要执行的语句。

查看反汇编代码（调试中在代码窗口右击，选择“转到反汇编”），并重复上述步骤。

在反汇编代码中观察函数的调用过程。

# 编写Windows程序

我们要调试的是Windows程序，所以必须了解Windows程序的结构。学写Windows程序是最简单的办法。这里推荐罗云彬的《琢石成器——Windows环境下32位汇编语言程序设计》。

这本书涉及到Windows相关的内容比较广，最好都看看，否则除了5、8、9、15、16、18章之外，其他都必须看。以这里的目录为准<http://book.douban.com/subject/3788437/>。

需要注意的一点是，这本书是以Win32汇编为基础讲解的，这种编程方法除了在某些特殊情形下，一般是不会用的，所以没有必要以它为编写Windows程序的基础。以学习性质编写、实验即可，但是代码一定要自己手打。

还有我忘了这本书是用什么东西调试的程序了，这里推荐用OllyDbg。

最后请完成以下自测题。可以自由选择编程语言，并随意使用搜索引擎。

## 综合自测题

用ZwCreateFile函数模拟一个CreateFileA。

挂钩自己进程的CreateFileW，让所有访问扩展名是.arc文件的操作中断（通过int3断点）。（挂钩相关请搜索inline hook）

通过SEH自己写一个简单的反调试。

选择一个Blowfish的库，调用它加密一个文件。（Blowfish是一种加密算法，网上有现成的实现可以使用）

尝试阅读Crass的代码以及asmodean的代码。

学会Python。（如果你按部就班学到这里的话，这个要求真的就只是一道测试题一样的程度了）

# 新的开始

到这里，可以说编程是刚刚入门，逆向可以开始起步了。