ハッカージャパントレーニング Elements

進め！リバースエンジニアリングの道

第1回 解析ツールの紹介

文●愛甲健二

リバースエンジニアリングとは

本来リバースエンジニアリングという言葉にはかなり広い意味合いが含まれており、ソフトウェアだけでなく、ハードウェアや場合によってはシステム全体の構造、内部仕様、構成技術などの把握を行うといった、対象物の隠された技術的仕様を解析、分析することを意味します。よって、例えば購入したパソコンを分解して、メモリやCPUの存在を確認したり、ハードディスクを取り出したり、メモリを増設することさえも広い意味ではリバースエンジニアリングと呼べるかもしれません。

ただ、残念ながら一般的なパソコンはすでに内部仕様などが周知されており、書店に置いてあるパソコン雑誌を読めば誰もがハードディスクを交換できますしメモリを増設できます。そういう意味では、リバースエンジニアリングを行う必要はなく、ただパソコン雑誌を購入して、記事に書かれてあるとおりに操作するだけでよいでしょう。

しかし、世の中には公開されていないが必要である情報が多々あります。例えばマルウェアの情報です。数ヵ月前に日本で大流行したマルウェアGumblarは、当然ですがソースコードが公開されていません。よって、Gumblarがどのファイルへアクセスしているのか、どのように感染するのか誰もわかりません。しかし、アンチウイルスベンダー各社はどうにかしてGumblarの被害を食い止めなければならないので、Gumblarの感染サイトや検体そのものを入手してその動作を解析します。

マルウェア解析はコンピューターセキュリティにおけるリバースエンジニアリングのもっとも一般的な例ですが、最初に書いたとおり、リバースエンジニアリングの本質は「対象物の隠された技術的仕様を解析する」ことです。その対象物はマルウェアだったり、競合他社製品だったり、脆弱性を持つアプリケーションだったりとさまざまですが、いずれに対しても、隠された技術的仕様を解析するという点で一致しており、この行為こそがリバースエンジニアリングと呼べます。

本連載について

かなり広い意味合いを持つリバースエンジニアリングという言葉ですが、コンピューターセキュリティにおいては、主にソフトウェアをアセンブラレベルで解析し、その動作概要を調べることという少し狭い範囲に置き換えられます。なので、意味としては「ソフトウェア解析」に近いと言えるでしょう。

そもそもハードウェア解析とソフトウェア解析では、それぞれ全く異なる基礎知識が必要であり、技術的視点においても別物と言えます。なので、業界によって意味の使い分けがなされており、コンピューターセキュリティ分野においては主にソフトウェア解析の意味として使われることが多いのです。もちろん多いというだけで、ハードウェア側のセキュリティの話をする際は当然ハードウェア解析という意味で使われます。

本連載ではハードウェアは扱いませんので、今後はソフトウェア解析という意味においてリバースエンジニアリングという言葉を使っていきます。また読者層として、プログラミングの経験はあるがソフトウェア解析の経験はない、つまりコンピューターについての基礎知識は持っているがリバースエンジニアリングは初めてだ、

という方を対象とします。1回4ページの連載なのであまり高度なものを扱うことはできませんが、最終的にはマルウェアを解析するくらいの記事を書ければと思います。というわけで、今後ともよろしくお願いいたします。

ツール紹介

リバースエンジニアリングという言葉の意味はわかったとして、では具体的に何をするのか？

何をすることでソフトウェアの動作を調べたり、脆弱性を発見できるのか？ という疑問が出てきます。その疑問に対する答えは簡単です。それはアセンブラ命令を読んでソフトウェアの動作を理解し、脆弱性のある部分を探し当てることです。そして、それらを行うためのツールが「逆アセンブラ」と「デバッガ」です。

逆アセンブラとは、CPUが解釈するマシン語を人間が解釈しやすいアセンブリ言語（ニーモニック）に変換するソフトウェアのことです。マシン語とアセンブリ言語は、他のプログラミング言語とは異なり基本的に命令コードが1対1で対応しているため、双方向の変換が容易です。

本当はマシン語からC言語あたりに変換してくれればいちばんよいのですが、逆コンパイルはまだあまり実用的ではないため、現状は逆アセンブラによって得られたアセンブリ言語を読むことがソフトウェア解析の一般的な方法となります。

次にデバッガですが、デバッガとはプログラムを実行しながら解析できるソフトウェアのことで、通常はソフトウェア開発者がバグの発見や修正を目的として使用します。セキュリティ分野においては、脆弱性診断やプログラムの挙動を調べる目的に使われることが多いのですが、本来は開発者向けのソフトウェアだと言えます。

ではさっそく、解析用途に使われる逆アセンブラとデバッガを紹介します。

○IDA Pro 4.9

IDA Pro（図1）は世界でもっとも使われている逆アセンブラなのですが、かなり高価なソフトウェアで、スタンダード版ですら10万円以上する代物です。しかし簡易版であるバージョン4.9がWebサイトにて無償で提供されており、高度な使い方をしないかぎりは簡易版でもほとんど問題がありません。最初のうちは無償の4.9を使っておき、本格的にソフトウェア解析について学びたいと感じたら、思い切って製品版を購入するのもよいでしょう。

インストールについても難しくありません。基本的にはダウンロードしたファイルidafree49.exeを実行し、表示されるメッセージに従ってNEXTボタンを押していくだけでインストールが完了します。

○OllyDbg

OllyDbg（図2）は有名なWindows用のデバッガの1つで、他のデバッガと比べて扱いやすいユーザーインターフェイスが特徴です。通常のデバッガは使い方が特殊であったり、難しいコマンドを覚えなければ扱えませんが、OllyDbgはそのような問題を極力改善して作

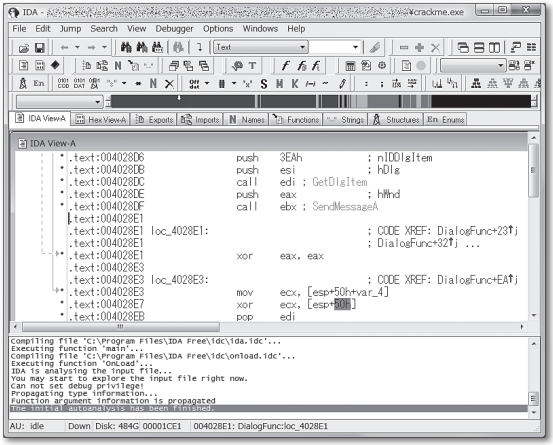

図1 IDA Pro 4.9 http://www.hex-rays.com/idapro/idadownfreeware.htm

られたデバッガです。またそこそこ人気もあり、これまでにも多くの解析者が使ってきた経緯から解説サイトや文献が多いというのも利点の1つです。

最近、最新版としてOllyDbg 2.00が正式にリリースされましたが、安定性や日本語化パッチの有無などの問題により、本記事ではOllyDbg 1.10を使います。ただ、日本語化パッチ以外の部分ではさほど違いはないように思えるので、このあたりは好みで選んでも構いません。

○crackme.exe

本来ならツール紹介の次は、これらのツールの使い方の解説などをやるべきかもしれませんが、今回は第1回目ですし、デバッガの使い方の解説なんて面白くないものに誌面を割くより、実際にプログラムを解析しましょう。実践に勝る練習方法はありません。

今回対象とするプログラムはcrackme.exeです（付録DVD-ROMに収録）。crackme.exeは実行するとメッセージボックスにおかしなエラーを表示して終了します（図3）。

さて、時間制限により出力されたこのエラーメッセージを回避して、無事crackme.exeを起動するにはどうすればよいでしょうか？

■OllyDbg上で実行

crackme.exeを実行すると時間に関するエラーが出力され、プログラムが終了します。1192年の5月15日にしか起動しないという何ともおかしなエラーですが、残念ながらWindowsの設定時間を1192年に戻すことはできません。しかし、crackme.exeの中で動いているのはただのマシン語なので、本当に時間を戻さなくともマシン語を書き換えることで制限を突破できます。

まずはcrackme.exeをOllyDbg上で実行します。OllyDbgを管理者権限で実行し、メニューから「ファイル」→「開く」をクリックします（図4）。そしてcrackme.exeを選択してください。次に同じくメニューから「解析」→「実行」と選択すると、crackme.exeがOllyDbg上で実行されます（図5）。ちなみにファイルの読み込みは<F3>を、プログラムの実行は<F9>を押すことで同様に行えます。

OllyDbgが起動した状態でcrackme.exeが立ち上がれば成功です。 おそらくcrackme.exeが実行された後なので、エラーのメッセージボックスが表示されているでしょう。では、ここでメッセージボックスのOKボタンを押さずに、OllyDbgのメニューから「解析」→「一時停止」を選択してください（図6）。<F12>を押すだけでも構いません。 これでcrackme.exeからOllyDbgへ処理が移りました。

OllyDbgに処理が移っている状態なので、crackme.exeは完全に停止しています。 ここでOllyDbgのメニューから「解析」→「ユーザーコードまで実行」を選択、もしくは<Alt>＋<F9>を押してください（図7）。これで再びcrackme.exeに制御が移ります。

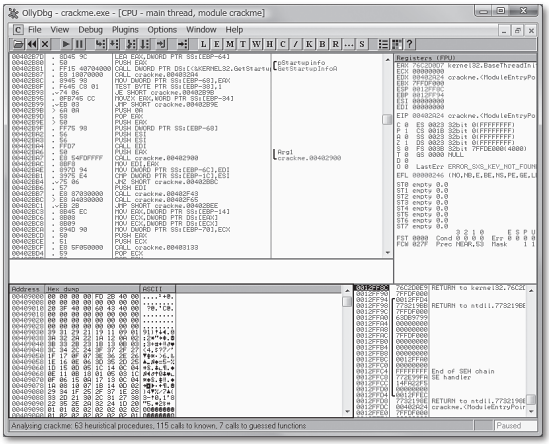

図2　OllyDbg 1.10　http://www.ollydbg.de/
（日本語化パッチ　http://hp.vector.co.jp/authors/VA028184/）

OllyDbgでcrackme.exeを解析

図3　crackme.exeを実行。1192年の5月15日にしか起動しないというエラーが出る。「OK」をクリックしてcrackme.exeを終了し、OllyDbgを起動する

図4　OllyDbgメニューの「ファイル」→「開く」でcrackme.exeを開く

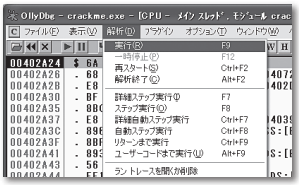

図5　「解析」→「実行」でOllyDbg上でcrackme.exeを実行

図6　「解析」→「一時停止」でcrackme.exeからOllyDbgへ制御を移す

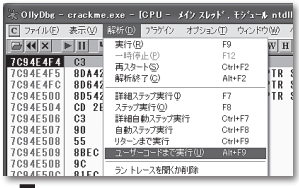

図7　「解析」→「ユーザーコードまで実行」を選択し再びcrackme.exeに制御を戻す

図8　00402998の命令コードでMessageBoxA関数が呼び出されている

そして最後にcrackme.exe側のメッセージボックスのOKボタンを押してください。すると、OllyDbg上の00402998の命令コードで処理が停止します(図8)。

時間制限回避のためには？

OllyDbgを使うことで、エラーのメッセージボックスを表示するためにMessageBoxA関数を呼び出している箇所(00402998)を特定しました。エラーメッセージが出ているということは、すでに現在時刻が1192年の5月15日ではないことが判明しているわけなので、おそらく現在時刻との判断部分は、MessageBoxA関数が呼び出される箇所よりも上の命令コードに存在するはずです。

次回は、現在時刻の取得部分、そして時間を評価し処理を分岐させている部分を特定し、時間制限を回避するバイナリパッチを作ります。