オブジェクトサイニングについての考察

6 E − 5

兒島 尚（東京工業大学情報理工学研究科）
hisashi@cs.titech.ac.jp
丸山 宏（日本アイ・ビー・エム東京基礎研究所及び東京工業大学情報理工学研究科）
maruyama@jp.ibm.com, maruyama@cs.titech.ac.jp

1. はじめに

現在、ネットワークを介して配布する署名付きオブジェクトが一般化しており、Netscape のコード署名 Java アプレット[5]、Microsoft の Authenticode を利用した ActiveX[4] などがある。これらは、第三者に署名者が意図しない使われ方をするのを避けるため、「できるだけ汎用性、再利用性のない」ようプログラムしなければならない。しかし、プログラマーには「できるだけ汎用性、再利用性の高い」ものを作るというプログラミングプラクティスが染み付いており、容易にその習性を変えることはできない。実際我々は Microsoft、Netscape のそれぞれについて、不注意による重大なセキュリティ違反を発見した[2] [3]。これは、「従来のプログラミングプラクティスに依存したセキュリティモデル」に限界があると我々は考える。今のところ、再利用性のないコードを書くか、砂箱モデル[1]以外によい解決策はない。

2. オブジェクトサイニング（コード署名）

オブジェクトサイニング（以下、コード署名）とは、公開鍵暗号方式を利用して、コードの出所とコードが改変されていないことを保証する。これにより問題が起こった時の責任の所在が明確になる。（ここでは公開鍵暗号方式そのものは基本的に安全であると仮定する）

3. Netscape のモデル

Netscape の Java コード署名モデルでは Java1.0 の砂箱モデルを拡張したものである。Java1.0 においては「実行できるかできないか」であったが、Netscape では Privilege という概念を導入し、ファイルの読み書き、ネットワーク接続などに異なる Privilege を設けた。プログラムがそのような動作をしたい場合、まずシステムに Privilege を要求し、システムはプログラムの署名をもとに、許可を与えたり、もしくはユーザーに許可を求めたりする。Privilege が取得できないと、プログラムはその動作が実行できない。Microsoft もこれに似た Java コード署名モデルを実装している。

3.1. Netscape の失敗

我々は JarPackager という、Netscape が署名したアプレットを公開 Web サイト上で発見した。これは署名付きの Java アプレット・アプリケーションを作るもので、Netscape により署名されているので、どんな Privilege でもユーザーに尋ねることなく取得できる。そしてその中に、

```
public void setInputFile(File file)
public InputStream getInputStream()
```

という public メソッドがあった。setInputFile でファイル名を指定し、getInputStream でその InputStream を取得することができる。問題はこれらがどちらも public であり、さらにメソッド内で Privilege を取得していたことであった。つまり署名されていないプログラムからこれらのメソッドを呼び出すことが可能なので、あらゆるファイルを読み出すことができるのである。Netscape は署名を更新し、JarPackager を抹消することで解決した。

4. Microsoft のモデル

Microsoft の ActiveX はネイティブコードなので、Java のようなアクセス制限は基本的にはなく、Initializable と Scriptable という2つのマークによって機能制限を行っている。ActiveX には外部から参照可能なプロパティとメソッドが定義されており、Initializable とマークされていれば、初期化時の色やテキストなどのプロパティの変更が可能となる。Scriptable とマークされていると、HTML 上のスクリプトからコントロールのメソッドやプロパティを参照できる。ここで注意すべきなのは、2つのマークとも署名者がコントロール作成時に実装するものであり、ユーザーにはまったく意識されないことである。Scriptable とマークされている場合、誰でもそのコントロールを操作することができるため、どんな操作をされても安全であるということを良く確認した上で、署名者はマークしなければならない。

4.1. Microsoft の失敗

Microsoft は二つの ActiveX コントロール A、B を作った。両方ともに Intializable、Scriptable であり、Microsft の Verisign Software Publisher Certificate により署名されている。

Discussion about Object Signing, Hisashi Kojima, Tokyo Institute of Technology, and Hiroshi Maruyama, IBM Research, Tokyo Research Laboratory and Tokyo Institute of Technology

ファイルのダウンロード

ActiveX コントロール A は Internet Explorer 4.0 の追加コンポーネントを Web 上からダウンロードしてインストールするためのコントロールである。これには SetCIFFile というメソッドがあり、これは与えられたファイル名のファイルをダウンロードしてきて、その中にあるはずの CIF ファイルというインストール用の設定ファイルを読み出すことが目的であった。しかし、このメソッドはいかなるファイルでも、とりあえず一時ディレクトリに保存してしまうことがわかった。そのディレクトリは 通常 C:¥windows¥msdownld.tmp¥ASE000.tmp であり、Internet Explorer 4.0 の起動中は有効であった。よって任意のファイルをユーザーのマシンにダウンロードし、その場所もほぼ確実に推測することができる。

プログラムの実行

ActiveX コントロール B は、Windows レジストリの HKEY_LOCAL_MACHINE 以下にあるものを、プログラムへのパスとして解釈して実行する能力があった。これは本来 NetWits という msn のサイトで、ユーザーにインストールされているあるゲームをブラウザから実行させるために作られたものだった。しかし、標準の Windows95 、WindowsNT4.0 の環境にある RunDll32.exe と URL.DLL を使うと、任意のローカルファイルを実行することができることがわかった。

ゆえにこの2つを組み合わせることにより、任意のファイルをダウンロードし、そして実行できる。Microsoft はこの問題を、ActiveX コントロール B を抹消することにより解消した。

5. 議論

Netscape と Microsoft のコード署名の一番の問題点は、ユーザーから信頼済みのプログラムが信頼されていないプログラムに利用されることである。それを防ぐために Netscape も Microsoft もコードを書く際には細心の注意を払うようにしつこく述べている。先の Microsoft の例では、例えば ActiveX コントロール A がダウンロードできる URL をハードコートして、Microsoft のサイトからでないとダウンロードできないようにすれば回避できた。しかしそれは「できるだけ再利用性、汎用性のない」プログラムを書くということになる。これは従来のプログラミングプラクティスである、「できるだけ再利用性、汎用性の高い」プログラムを書くということに反している。多くのプログラマーにはこの習性が染み付いており、容易に変えられるものではない。提案者の Netscape や Microsoft でさえもこのことを守れなかったのである。これはつまり、このようなコンポーネント単位のセキュリティモデルそのものに限界がある、と我々は考える。Microsoft の例での ActiveX コントロール A と B は、本来の使用目的としては互いに全く関連が無かったはずである。複数のコンポーネントの組み合わせによって、はじめて問題が露呈する。これはもはやコンポーネント個々の開発者の手におえる問題ではない。

アプリケーションやシステムが複雑化するにつれ、ソフトウェア開発で一気に全体を構築するような方法は難しくなり、コンポーネントごとに分割し、それらを組み合わせるという方法が発達してきた。その最たるものが「オブジェクト指向プログラミング」であり、これにより「コードの再利用性、汎用性」が高くなり、大幅に開発コストが減少された。しかしセキュリティに関しては、Netscape や Microsoft の失敗を見るように、コンポーネントレベルのセキュリティモデルの積み重ねで全体のセキュリティシステムを構築することは難しいと言える。

今のところ、できるだけ再利用性の無いコードを書くよう努力するか、砂箱モデルしか良い解決策はない。コンポーネントの相関関係をモデリングする、新たな理論が必要であろう。

参考文献

[1] "The Java Language Environment White Paper," http://www.javasoft.com/docs/white/langenv/, JavaSoft, 6/1996.

[2] Maruyama, H, Tamura, K., and Kojima, H., "A Security Hole in Netscape Communicator 4," TRL Research Report TR-0220, IBM Research, Tokyo Research Laboratory, 10/3/1997.

[3] Maruyama, H. and Kojima, H., "A Security Hole in Microsoft Internet Explorer 4.0," TRL Research Report TR-0231, IBM Research, Tokyo Research Laboratory, 12/22/1997.

[4] "Developer's Guide to Deploying ActiveX Executables on the Internet," http://www.microsoft.com/security/guide/devguide.htm, Microsoft Corporation, 4/2/1997.

[5] "Netscape Object Signing -- Establishing Trust for Downloaded Software," http://developer.netscape.com/library/documentation/signedobj/trust/owp.htm, Netscape Communications Corporation, 7/2/1997.

[6] Garfinkel, S. and Spafford, G., "Web Security & Commerce", O'Reilly & Associates., 6/1997.