IOActive Security Advisory

Title	`doc.export*` Methods Allow Arbitrary File Creation
Severity	High
Date Discovered	July 13, 2009

Affected Products

Adobe Reader 9.1.2; earlier versions are likely vulnerable as well.

Description

Several JavaScript methods of the Document Object do not honor the Privileged Context and Safe Path settings. IOActive was able to execute certain privileged JavaScript methods that can be used to create arbitrary files and folders on a targeted file system.

Technical Details

According to the Acrobat user documentation, certain methods are either not available or have security restrictions in a non-privileged context. Therefore, it should not be possible to call these methods from *page open* events. The following content is taken from the user documentation:

> *Some Acrobat JavaScript methods, marked by S in the third column of the quick bar, have security restrictions. These methods can be executed only in a privileged context, which includes console, batch, menu, and application initialization events. All other events (for example, page open and mouse-up events) are considered non-privileged.*

IOActive discovered that some methods do not adhere to this policy; the following methods are executable from the 'page open' event:

- doc.exportAsFDF
- doc.exportAsText
- doc.exportAsXFDF
- doc.exportDataObject
- doc.exportXFAData

These methods take a `cPath` parameter, which can be used to create a file or folder on the file system and should be considered a "safe path." For example, the documentation has this to say about the `cPath` parameter for the `doc.exportAsText` method:

NOTE:(SecurityS): The parameter cPath is must have a safe path (see "Safe Path" on page 34) and have a .txt extension. This method will throw a NotAllowedError (see Error Object) exception if these security conditions are not met, and the method will fail.

This is incorrect as the method allows any arbitrary file extension and any arbitrary path, which means that an attacker can use any arbitrary file extension and any arbitrary path. Attackers can leverage this vulnerability to create malicious files in targeted system folders.

Proof of Concept

The following code will write a file with the `.exe` extension in the Startup folder of the targeted user:

```
6 0 obj<</S/JavaScript/JS ( 0
function Init\(\) {
this.exportAsText(false, null, "/c/Documents and
Settings/Administrator/Start
Menu/Programs/Startup/malware_installer.exe");
} Init\(\); )>>endobj
1 0 obj<</Type/Catalog/Pages 2 0 R/OpenAction 6 0 R>>endobj
```