Network Working Group
Internet-Draft
Expires: August 23, 2004

E. Taft
J. Pravetz
S. Zilles
L. Masinter
Adobe Systems
February 23, 2004

The application/pdf Media Type
draft-zilles-pdf-03.txt

Status of this Memo

Copyright Notice

Abstract

PDF, the 'Portable Document Format', is a general document representation language that has been in use for document exchange on the Internet since 1993. This document provides an overview of the PDF format, explains the mechanisms for digital signatures and encryption within PDF files, and updates the media type registration of 'application/pdf'.

Table of Contents

1. Introduction

This document is intended to provide updated information on the registration of the MIME Media Type "application/pdf", with particular focus on the features that help mitigate security concerns. This document refers to features documented in the PDF References versions 1 [1], 1.3 [2], 1.4 [3] and 1.5 [4], as updated by errata [5].

PDF is used widely in the Internet Community. Since PDF was introduced in 1993, it has grown to be a widely-used format for capturing and exchanging formatted documents electronically, across the Web, via e-mail, and, for that matter, virtually every other document exchange mechanism.

PDF represents formatted documents. These documents may be structured or simple. They may contain text, images, graphics and other multimedia content, such as video and audio. There is support for annotations, metadata, hypertext links, and bookmarks.

PDF supports encryption and digital signatures in the document. The encryption capability is also combined with access control information in a way that is intended to manage the uses that a recipient can make of a document.

PDF usage is specified in other international standards. ISO 15930-1:2001 PDF/X [16] has been adopted as the exchange standard for electronic documents within the Prepress community. PDF/X is a profile of PDF that references the PDF Reference, Third edition [2] as the source specification.

Another profile of PDF, known as PDF/A [17], is being developed for use as an international standard as an electronic document file format for long-term preservation. Following the work on PDF/X, the activity is joint work between NPES (The Association for Suppliers of Printing, Publishing and Converting Technologies) and AIIM International (the Association for Information and Image Management, International). AIIM is the secretariat for ISO/TC 171 SC2, Document Imaging Applications.

PDF usage is widespread enough for 'application/pdf' to be used in other IETF specifications. RFC2346 [15] describes how to better structure PDF files for international exchange of documents where different paper sizes are used; HTTP byte range retrieval is illustrated using application/pdf (RFC2616 [14], Section 19.2); RFC3297 [13] illustrates how PDF can be sent to a recipient that identifies his ability to accept the PDF using content negotiation.

2. History

PDF was originally envisioned as a way to communicate and view printed information electronically reliably across a wide variety of machine configurations, operating systems and communication networks.

PDF relies on the same imaging model as the PostScript page description language to render complex text, images and graphics in a device and resolution-independent manner, bringing this feature to the screen as well as the printer. To improve performance for interactive viewing, PDF defines a more structured format than that used by most PostScript language programs. PDF also includes objects, such as hypertext links and annotations, that are not part of the page itself but are useful for building collections of related documents and for reviewing and commenting on documents.

The application/pdf media type was first registered in 1993 by Paul Lindner for use by the gopher protocol; the registration was subsequently updated in 1994 by Steve Zilles.

3. Fragment identifiers

The handling of fragment identifiers [6] is currently defined in Adobe Technical Note 5428 [7]. This section summarizes that material.

A fragment identifier consists of one or more PDF-open parameters in a single URL, separated by the ampersand (&) or pound (#) character. Each parameter implies an action to be performed and the value to be used for that action. Actions are processed and executed from left to right as they appear in the character string that makes up the fragment identifier.

The PDF-open parameters allow the specification of a particular page or named destination to open. Named destinations are similar to the "anchors" used in HTML or the IDs used in XML. Once the target is specified, the view of the page in which it occurs can be specified, either by specifying the position of a viewing rectangle and its scale or size coordinates or by specifying a view relative to the viewing window in which the chosen page is to be presented.

The list of PDF-open parameters and the action they imply is:

nameddest=<name>
   Open to a specified named destination (which includes a view).

page=<pagenum>
   Open the specified (physical) page.

zoom=<scale>,<left>,<top>
   Set the <scale> and scrolling factors. <left> and <top> are measured from the top left corner of the page independent of the size of the page. The pair <left> and <top> are optional but

both must appear if present.

view=<keyword>,<position>
Set the view to show some specified portion of the page or its bounding box; keywords are defined by Table 8.2 of the PDF Reference, version 1.5. The <position> value is required for some of the keywords and not allowed for others.
viewrect=<left>,<top>,<wd>,<ht>
As with the zoom parameter, set the scale and scrolling factors, but using an explicit width and height instead of a scale percentage.
highlight=<lt>,<rt>,<top>,<btm>
Highlight a rectangle on the chosen page where <lt>, <rt>, <top> and <btm> are the coordinates of the sides of the rectangle measured from the top left corner of the page.

All specified actions are executed in order; later actions will override the effects of previous action; for this reason, page actions should appear before zoom actions. Commands are not case sensitive (except for the value of a named destination).

4. Encryption

PDF files allow access to be controlled using encryption and permission settings. The keys to decrypt document data, and permission settings for a document, are provided by encryption handlers. An 'Encryption Dictionary' is provided in the document trailer to enable encryption handlers to store document-specific information. Different encryption handlers can provide for different sets of permissions. The PDF encoding rules for password and public key encryption handlers is specified in the PDF Reference.

A person that is able to 'access' a document is said to be able to open and view the document. Access is possible when a person can provide the key with which to decrypt the document. The key is protected and provided by the encryption handler. Encryption handlers will normally require some sort of authentication before a person can access the document decryption key.

Encryption of PDF files is normally applied to all string and stream data in the document, and only to string and stream data. By encrypting only data portions of the PDF file, random access to PDF file contents is maintained. The data is normally encrypted using 40 to 128-bit RC4 [8] encryption algorithm. Use of decryption filters allows algorithms other than RC4 to be used.

The person that has access to a document will be given certain permissions for the document. A person that has full permissions, including permission to save a document without encryption, is said to be an 'owner'. A person that has restricted permissions is said to

be a 'user'. Example permissions include the ability to copy text and other content from the PDF file, the ability to fill in form field

data, and the ability to print the PDF file. Enforcement of permissions is the responsibility of the viewing application.

Password encryption allows the possibility of two different passwords to be used when providing access to the document. The 'author' password allows access to the document and full permissions, including the permission to save the document without encryption. The 'user' password allows access to the document, but access is restricted by a set of permissions.

Public key encryption of PDF files uses one or more PKCS#7 [9] objects to store information regarding recipients that are able to open a document. Each PKCS#7 object contains a list of recipients, a document decryption key, and permission settings that apply to all recipients listed for that PKCS#7 object. The document decryption key is protected with a triple-DES key that is encrypted once with the public key of each listed recipient.

5. Digital Signatures

A digital signature can be used to authenticate the identity of a user and the validity of a document's contents. PDF supports the association of a digital signature with a complete record that is needed to reproduce a visual representation of what a person saw when they signed the PDF file. PDF digital signatures allows for multiple signers to update and sign the same document; a subsequent user may then view the state of the document at each point when any individual signature was applied.

The full specification for PDF digital signatures is contained in the PDF Reference [4] section 8.7 and Appendix I; an overview is provided here.

PDF signature information is stored in a 'signature dictionary' data structure. A signature is created by computing a digest of the data stored in the document. To verify the signature, the digest is recomputed and compared with the one stored in the document. Differences in the digest values indicate that modifications have been made since the document was signed.

All bytes of the PDF file are covered by the signature digest, including the signature dictionary, but excluding the signature value itself. The range of bytes is defined and stored as the value of the ByteRange key in the signature dictionary. The ByteRange value is an array of integer pairs, where each pair includes a starting byte offset and length in bytes. There are two pairs, one describing the range of bytes preceeding the signature value, and the other describing the range of bytes that occur after the signature value.

PDF public key digital signature syntax is specified for PKCS#1 [11] and PKCS#7 [9] signatures. In both cases, all bytes of the PDF file are signed, with the exclusion of the PKCS#1 or PKCS#7, signature value, objects.

The signature dictionary contains additional attributes. The 'SubFilter' attribute describes the encoding of the signature value, and the 'Contents' attribute contains the signature value which is normally hex (base16) encoded. There are currently three recommended SubFilter types:

adbe.x509.rsa_sha1
: In this case the Contents key contains a DER-encoded PKCS#1 [11] binary data object representing the signature obtained as the RSA encryption of the byte range SHA-1 digest with the signer's private key. When using PKCS#1, the certificate chain of the signer is included with other signature information in the signed document.

adbe.pkcs7.sha1
: In this case the value of Contents is a DER-encoded PKCS#7 binary data object containing the signature. The SHA1 digest of the byte range is encapsulated in the PKCS#7 signed-data field with ContentInfo of type "data".

adbe.pkcs7.detached
: In this case the value of Contents is a DER-encoded PKCS#7 binary data object containing the signature. No data is encapsulated in the PKCS#7 signed-data field.

If the type of signature is 'adbe.x509.rsa_sha1', the signature dictionary includes a key named 'Cert', which contains at least the signer's X.509 public-key certificate represented as a binary string. The value could also be an array of strings where the first entry is the signer's certificate and the following entries are one or more issuer certifications from the signer's trust chain.

If the type of signature is 'adbe.pkcs7.sha1' or 'adbe.pkcs7.detached', the 'Cert' key is not used and the certificate must be put in the PKCS#7 object stored in the 'Contents' key. The minimum required certificate to include in the PKCS#7 object is the signer's X.509 signing certificate. It may optionally contain also one or more issuer certifications from the signer's trust chain.

Multiple signatures are supported using the incremental save capabilities of PDF. When changes to a file are made and a new signature is applied to the document, the changes are appended after the last byte of the previously existing document and then the new signature digest is of all bytes of the new file. In this manner changes can be made to a document and new signatures added to a

document without invalidating earlier signatures that have been applied to the PDF file. Any change to a document is detected because all bytes of the PDF file are digested.

The state of a signed document, when an earlier signature of a multiple signature document was applied, can be viewed by extracting the earlier set of bytes of the file and opening them in a PDF viewing application. This process is called 'rollback' and allows viewing of the exact state of the document when it was signed.

PDF syntax allows for 'author' and 'user' signatures. Under normal circumstances the first signature of a document is considered an author signature and all other signatures are considered user signatures. Authors can specify what changes are to be allowed to the PDF file before the author's signature is presented as invalid. Example changes include the ability to fill in form field data, the ability to add comments to a document, the ability to make no changes, and the ability to make any changes. Changes are detected by opening the existing document and the author's version of the document and performing a complete object compare of the two documents. Change detection is not a substitute for the legal value of document rollback.

6. PDF implementations

There are a number of widely available, independently implemented, interoperable implementations of PDF for a wide variety of platforms and systems. Because PDF is a publicly available specification, hundreds of companies and organizations make PDF creation, viewing, and manipulation tools. For examples, see descriptions or tools lists from Adobe [20], Apple [21], Ghostscript [22], Planet PDF [18] and PDFzone.com [19].

7. Security considerations

An "application/pdf" resource contains information to be parsed and processed by the recipient's PDF system. Because PDF is both a representation of formatted documents and a container system for the resources need to reproduce or view said documents, it is possible that a PDF file has embedded resources not described in the PDF Reference.

Although it is not a defined feature of PDF, a PDF processor could extract these resources and store them on the recipients system. Furthermore, PDF processor may accept and execute "plug-in" modules accessible to the recipient. These may also access material in the PDF file or on the recipients system. Therefore, care in establishing the source, security and reliability of such plug-ins is recommended. Message-sending software should not make use of arbitrary plug-ins without prior agreement on their presence at the intended recipients. Message-receiving and -displaying software should make sure that any non-standard plug-ins are secure and do not present a security

threat.

PDF may contain "scripts" to customize the displaying and processing of PDF files. These scripts are expressed in a version of JavaScript [10] based on JavaScript version 1.5 of ISO-16262 (formerly known as ECMAScript). These scripts have access to an API that is similar to the "plug-in" API. They are intended for execution by the PDF processor. User agents executing such scripts or programs must be extremely careful to insure that untrusted software is executed in a protected environment.

In addition, JavaScript code might modify the appearance of a PDF document. For this reason, validation of digital signatures should take this into account.

In general, any information stored outside of the direct control of the user -- including referenced application software or plug-ins and embedded files, scripts or other material not covered in the PDF reference -- can be a source of insecurity, by either obvious or subtle means. For example, a script can modify the content of a document prior to its being displayed. Thus, the security of any PDF document may be dependent on the resources referenced by that document.

As noted above, PDF provides mechanism for helping insure the integrity of a PDF file, Encryption (Section 4), and to be able to digitally sign (Section 5) a PDF file. The latter capability allows a recipient to decide if he is willing to trust the file.

Where there is concern that tampering with the PDF file might be a problem it is recommended that the encryption and digital signature features be used to protect and authoritate the PDF.

In addition, PDF processors may have mechanisms that track the source of scripts or plug-ins and will execute only those scripts or plug-ins that meet the processors requirements for trustworthiness of the sources.

8. IANA considerations

This document updates the registration of 'application/pdf', a media type registration as defined in Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures [23]:

MIME media type name: application
MIME subtype name: pdf

Required parameters: none
Optional parameter: none
Encoding considerations:
PDF files frequently contain binary data, and thus must be encoded in non-binary contexts.
Security considerations:
See Section 7 of this document.
Interoperability considerations:
See Section 6 of this document.
Published specification:
Adobe Systems Incorporated, "PDF Reference, Fourth Edition", Version 1.5, August 2003, <http://partners.adobe.com/asn/tech/pdf/specifications.jsp>, as amended by errata <http://partners.adobe.com/asn/acrobat/sdk/public/docs/errata.txt>.
Applications which use this media type:
See Section 6 of this document.
Additional information:
Magic number(s): All PDF files start with the characters '%PDF-' using the PDF version number, e.g., '%PDF-1.4'. These characters are in US-ASCII encoding.
File extension(s): .pdf
Macintosh File Type Code(s): "PDF "
For further information:
Adobe Developer Support <dev-support@adobe.com>
Adobe Systems Incorporated
345 Park Ave
San Jose, CA 95110
http://www.adobe.com/support/main.html
Intended usage: COMMON
Author/Change controller:
Adobe Developer Support <dev-support@adobe.com>
Adobe Systems Incorporated
345 Park Ave
San Jose, CA 95110
http://www.adobe.com/support/main.html

References

[1] Adobe Systems Incorporated, "Portable Document Format Reference Manual", Version 1.0, ISBN: 0-201-62628-4, Addison-Wesley, New York NY, 1993.

[2] Adobe Systems Incorporated, "PDF Reference, Second Edition", Version 1.3, ISBN: 0-201-61588-6, Addison-Wesley, New York NY, 2000.

[3] Adobe Systems Incorporated, "PDF Reference, Third Edition",

Version 1.4, ISBN: 0-201-75839-3, Addison-Wesley, New York NY, November 2001.

[4] Adobe Systems Incorporated, "PDF Reference, Fourth Edition", Version 1.5, August 2003, <http://partners.adobe.com/asn/tech/pdf/specifications.jsp>.

[5] Adobe Systems Incorporated, "Errata for PDF Reference, Fourth Edition", December 2003, <http://partners.adobe.com/asn/acrobat/sdk/public/docs/errata.txt>.

[6] Berners-Lee, T., Fielding, R. and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998.

[7] Adobe Systems Incorporated, "PDF Open Parameters", Technical Note 5428, May 2003, <http://partners.adobe.com/asn/acrobat/sdk/public/docs/PDFOpenParams.pdf>.

[8] Rivest, R., "RC4 - an unpublished, trade secret encryption algorithm", November 1993, <http://www.rsasecurity.com/rsalabs/faq/3-6-3.html>.

[9] Kaliski, B., "PKCS #7: Cryptographic Message Syntax Version 1.5", RFC 2315, March 1998.

[10] Adobe Systems Incorporated, "Acrobat JavaScript Scripting Reference", Technical Note 5431, September 2003, <http://partners.adobe.com/asn/acrobat/sdk/public/docs/AcroJS.pdf>.

[11] Jonsson, J. and B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1", RFC 3447, February 2003.

Informative References

[12] Adobe Systems, "Adobe Patent Clarification Notice", 2003, <http://partners.adobe.com/asn/developer/legalnotices.jsp>.

[13] Klyne, G., Iwazaki, R. and D. Crocker, "Content Negotiation for Messaging Services based on Email", RFC 3297, July 2002.

[14] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P. and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[15] Palme, J., "Making Postscript and PDF International", RFC 2346, May 1998.

[16] International Standards Organization, "Graphic technology -- Prepress digital data exchange -- Use of PDF -- Part 1:

Complete exchange using CMYK data (PDF/X-1 and PDF/X-1a)", ISO 15930-1:2001, November 2002.

[17] Association for Information and Image Management, "PDF-Archive Committee home page", December 2003, <http://www.aiim.org/pdf_a/>.

[18] Planet PDF, "Planet PDF Tools List", December 2003, <http://www.planetpdf.com/>.

[19] InternetBiz.net, "PDF software from the PDF zone toolbox", December 2003, <http://www.pdfzone.com/toolbox/>.

[20] Adobe Systems Incorporated, "Adobe products page", December 2003, <http://www.adobe.com/products/>.

[21] Apple Computer, Inc., "Apple Mac OS X Features - Preview", December 2003, <http://www.apple.com/macosx/features/preview/>.

[22] Artifex Software, Inc, "Ghostscript", December 2003, <http://www.ghostscript.com/>.

[23] Freed, N., Klensin, J. and J. Postel, "Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures", RFC 2048, November 1996.

Authors' Addresses

Edward A. Taft
Adobe Systems
345 Park Ave
San Jose, CA 95110
US

EMail: taft@adobe.com

James D. Pravetz
Adobe Systems
345 Park Ave
San Jose, CA 95110
US

EMail: jpravetz@adobe.com

Stephen Zilles
Adobe Systems
345 Park Ave
San Jose, CA 95110
US

Phone: +1 408 536 7692
EMail: szilles@adobe.com

Larry Masinter
Adobe Systems
345 Park Ave
San Jose, CA 95110
US

Phone: +1 408 536 3024
EMail: LMM@acm.org
URI: http://larry.masinter.net

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.