STUDIENARBEIT

SPEZIELLE GEBIETE
DER TEXT- UND BILDVERARBEITUNG –
VERTIEFUNG POSTSCRIPT/PDF

SICHERHEIT VON PDF-DATEIEN

Print and Media Technologies
FB E | BU Wuppertal

Alexander Jacob | 2005

INHALTSVERZEICHNIS

1. EINLEITUNG

Die Nutzung von PDF-Dateien hat in den letzten Jahren enorm zugenommen und dies nicht nur in der Druckvorstufe. Der Office-Bereich hat PDF als zuverlässiges Mittel zur Archivierung von Dateien oder zum Austausch von Informationen entdeckt, vor allem wegen der Plattformunabhängigkeit von PDF.
In allen Anwendungsbereichen spielt nicht nur die Zuverlässigkeit von PDF als Datenformat eine Rolle, sondern auch dessen Möglichkeit, einen Zugriff durch Unbefugte zu verhindern oder für einzelne Nutzer Zugriffsbeschränkungen einzurichten.
Besonders in der Druckvorstufe kann es sich als großer Vorteil erweisen, wenn bestimmte Personenkreise unterschiedliche Rechte für eine Datei haben (Kontrolle, Proofen, Ändern, etc.), damit die Integrität der Daten sichergestellt ist und Änderungen nur durch befugte Personen vorgenommen werden können.

2. SCHUTZ VON PDF-DATEIEN

Die erwünschte Eigenschaft ist es, dass jeder, der einen PDF-Viewer besitzt, die Datei einsehen kann und nicht die zur Erstellung benutzte Software besitzen muss.
Leider ist dabei einer PDF-Datei der Schutz vor Unbefugten nicht in die Wiege gelegt worden, sondern es kann ein vom Benutzer angelegter Schutz über ein Passwort integriert werden.
Um Änderungen der Datei oder ein Öffnen von Unbefugten auszuschließen, lässt sich in Acrobat sowohl ein Owner-Passwort als auch ein User-Passwort vergeben. Das Owner-Passwort ist sozusagen das Master-Passwort. Es wird vom Ersteller angelegt und mit ihm besitzt man nicht nur Zugriff auf die Datei, sondern kann auch Änderungen in allen Bereichen durchführen. Der Ersteller kann ebenfalls ein User-Passwort einrichten, mit dem Nutzungsbeschränkungen verbunden sein können.
Diese Beschränkungen können sich auf folgende Bereiche beziehen:

- Drucken
- Kopieren und Ändern des Inhaltes bzw. des Dokumentes
- Auswählen von Text bzw. Grafik
- Hinzufügen bzw. Ändern von Anmerkungen und Formularfeldern

Auf Wunsch kann der PDF-Viewer angewiesen, werden jegliche Änderungsmöglichkeiten zu deaktivieren bzw. zu unterdrücken oder einzuschränken.
Ohne Einschränkungen lässt sich eine PDF-Datei problemlos in hoher Auflösung ausdrucken, editieren und Inhalte wie z.B. Bilder und Grafiken über das Snapshot-Tool extrahieren.
Beschränkungen und Berechtigungen können mit „Drucken erlaubt: Niedrige Auflösung (150 dpi)“ oder „Einfügen, Löschen und Drehen von Seiten erlaubt“ angelegt werden, um Beispiele zu nennen.

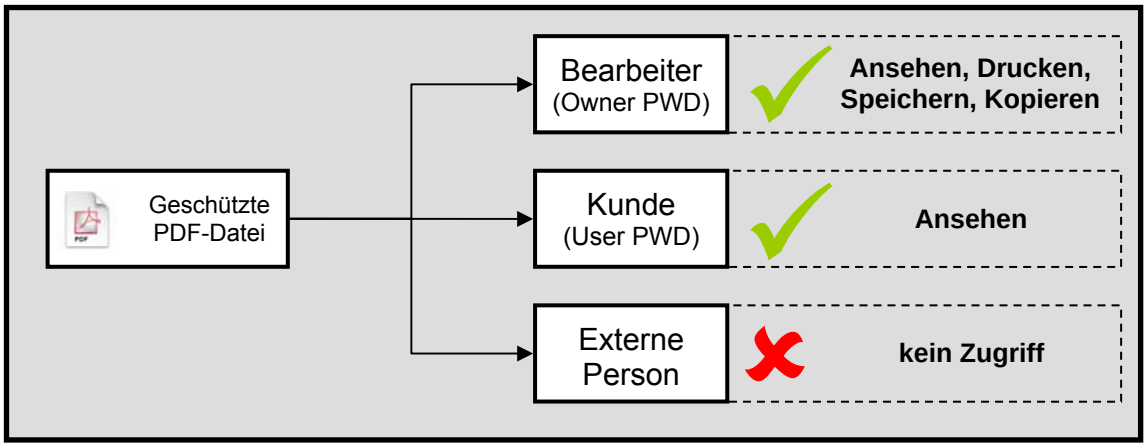

Abb. 1 Beispiel für Nutzungsbeschränkungen über Owner/User Passwort

Ein Kunde kann dadurch z.B. über ein User-Passwort die PDF-Datei von seinem Auftrag ansehen, aber sonst keine Änderungen durchführen. Über ein Owner-Passwort kann dies aber der Bearbeiter der Datei. Externe Personen, die kein Passwort besitzen, können (ohne Hilfsmittel) diese Datei gar nicht öffnen.

3. VERSCHLÜSSELUNGSMETHODEN/-ALGORITHMEN UND DEREN ANWENDUNG IN ACROBAT

3.1 Schlüssel/Key und Verschlüsselung

Passwörter werden nach der Eingabe in Acrobat über eine so genannte Hash-Funktion in Schlüssel, immer gleich lange Zeichenketten, verwandelt. Adobe bezeichnet diesen Schlüssel als „Encryption Key“. Die Länge des Schlüssels wird in Bit angegeben. Wenn man daher über eine 128 Bit-Verschlüsselung spricht, die ab Acrobat 5.0 Standard ist, sollte man sich immer im Klaren darüber sein, dass es sich dabei um eine Verschlüsselung mit einer Schlüssellänge von 128 Bit handelt.

$$128\ \text{Bit} = 2^{128} \approx 3.4\text{x}10^{38}\ \text{Zeichen} = 340282366920938463463374607431768211100!$$

Eine Faustregel besagt, dass, wenn man den Schlüssel um ein Bit verlängert, sich die Stärke des Verschlüsselungsverfahrens gegen Angriffe mittels Ausprobieren verdoppelt. Das Verschlüsseln mit einer möglichst großen Bit-Zahl erscheint daher mehr als sinnvoll.

Eine Verschlüsselung bedeutet immer viel Mathematik, die im Hintergrund abläuft. Generell kann gesagt werden, dass mit einem Schlüssel ein Klartext in einen Chiffriertext umgewandelt wird, denn im Schlüssel stecken (meist zwei) Variablen, ohne die man das Dokument nicht ver- und entschlüsseln kann. Mit der einen wird der Klartext „umgerechnet“ und mit der anderen wieder „zurückgerechnet“.

3.2 Symmetrische Verschlüsselung

Die symmetrische Verschlüsselung ist die älteste Form der Verschlüsselung. Jeder von uns hat sicherlich in seiner Kindheit einmal Geheimbotschaften, die über einen bestimmten Schlüssel zu entschlüsseln waren, geschrieben (z.B. jedem Buchstaben im Alphabet wird eine Zahl zugewiesen und damit Wörter „kodiert“). Bei der symmetrischen Verschlüsselung wird der gleiche Schlüssel sowohl zum Verschlüsseln als auch zum Entschlüsseln benutzt, wie es die nachfolgende Abbildung zeigt.

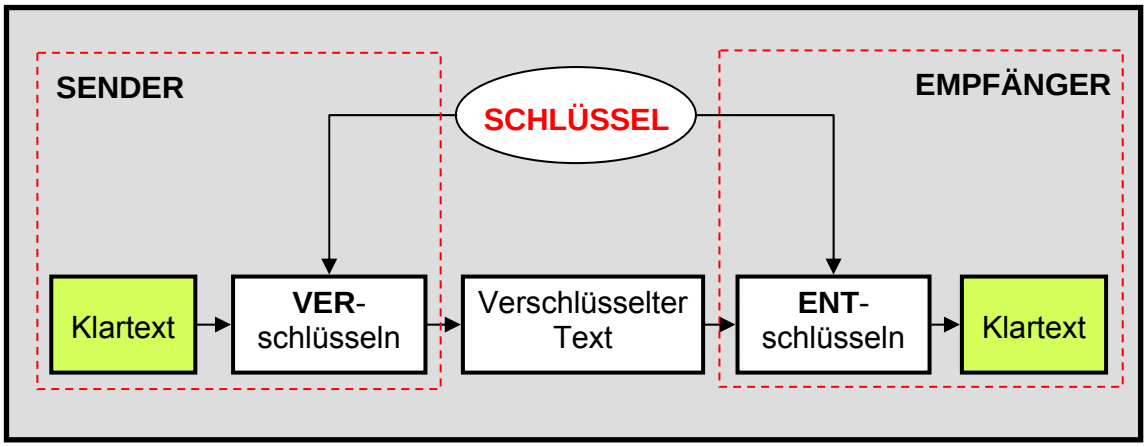

Abb. 2 Symmetrische Verschlüsselung

Bei der Schlüsselgenerierung werden zwei etwa gleich lange Primzahlen (*p* und *q*) erzeugt. Diese werden multipliziert und man erhält das Produkt *N*. Anschließend wird auf *N* die *Euler-Funktion* angewandt ($\varphi(N) = (p-1) \cdot (q-1)$) und zwei weitere Zahlen (*e* und *d*), die möglichst teilerfremd zu $\varphi(N)$ sind, erzeugt. Die generierten Variablen werden im Schlüssel gespeichert, *e* und *N* werden zum Verschlüsseln benutzt und *d* und $\varphi(N)$ zum Entschlüsseln. Diese Kombinationen sind durch die verwendete Mathematik begründet, denn die „Rückrechnung“ kann nicht über dieselbe Variable stattfinden.

SCHLÜSSEL	***p*** und ***q***	***N***	***φ(N)***	***e***	***d***

Abb. 3 Symmetrischer Schlüssel

Diese Verschlüsselungsmethode hat einen großen Nachteil, denn der Schlüssel muss beiden Parteien bekannt sein. Besonders bei einer räumlichen Trennung muss dem anderen Partner der Schlüssel mitgeteilt werden, teilweise geschieht das über unsichere Wege, wie z.B. E-Mail.
Um diese Schwachstelle zu umgehen wurde die asymmetrische Verschlüsselung entwickelt.

3.3 Asymmetrische Verschlüsselung

Bei der asymmetrischen Verschlüsselung erzeugt jede Partei ein Schlüsselpaar aus Public und Private Key. Der Public Key kann in der Öffentlichkeit verbreitet werden, der Private Key muss jedoch geheim gehalten werden. Um jemanden z.B. eine verschlüsselte Nachricht zu schicken, benutzt man zum Verschlüsseln dessen Public Key. Zum Entschlüsseln benutzt der Empfänger dann seinen Private Key und nur er kann damit aus der Nachricht wieder einen Klartext erzeugen.

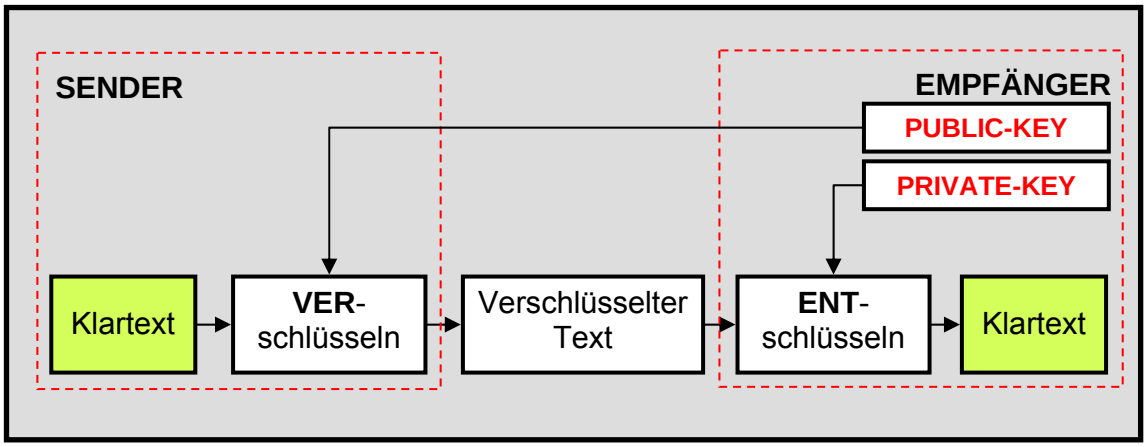

Abb. 4 Asymmetrische Verschlüsselung

Ziemlich verwirrend erscheint es, wieso der Empfänger mit seinem Schlüssel die Nachricht entschlüsseln kann.
Dazu muss man sich die zu Grunde liegende Mathematik etwas näher anschauen.
Wie bei der symmetrischen Verschlüsselung werden wieder zwei Primzahlen (*p* und *q*) erzeugt, daraus das Produkt *N*, die Euler-Funktion angewendet und zwei weitere Variablen *e* und *d* erzeugt. Allerdings werden dann bei der Schlüsselpaargenerierung die Variablen *e* und *N* im Public Key und *d, p, q (also auch N)* und *φ(N)* im Private Key gespeichert. Mit den Variablen *e* und *N* wird verschlüsselt und mit *d* und *φ(N)* entschlüsselt.
Der Vorteil liegt auf der Hand. Jeder kann dem Empfänger eine verschlüsselte Nachricht schicken, aber nur dieser kann die Nachricht entschlüsseln.

PUBLIC KEY	***p* und *q***	***N***	***e***

PRIVATE KEY	***p* und *q***	***(N)***	***φ(N)***	***d***

Abb. 5 Public- und Private-Key

3.3.1 Asymmetrische Verschlüsselung in Acrobat: Self-Sign Sicherheit

Adobe bedient sich dieser Eigenschaft der asymmetrischen Verschlüsselung und hat die Möglichkeit die Public Key Struktur (PKI) in Acrobat über das Self-Sign-Plugin einzubinden. Dies gilt allerdings nur für das Acrobat-Vollprodukt und wird vom Adobe Reader nicht unterstützt.
Wird eine PDF-Datei für einen Job erzeugt hat der Ersteller die Möglichkeit verschiedene Nutzer mit identischen bzw. unterschiedlichen Nutzungsbeschränkungen für diese Datei anzulegen. Die Public Keys der Nutzer sind als Zertifikate hinterlegt, die dann in der PDF-Datei hinterlegt werden.
Der schon angesprochene Encryption Key, der auch zum Verschlüsseln der Daten benutzt wird, wird mit den Nutzungsbeschränkungen (Permissions) und dem Public Key des Users zu einem Encrypt Key vereint, der innerhalb der PDF-Datei hinterlegt wird.
Auf diese Weise können etliche User mit unterschiedlichen Berechtigungen angelegt werden, für jeden Nutzer muss dann aber auch ein Encrypt Key erzeugt werden.

Möchte einer der User die Datei öffnen, wird mit seinem Private Key der Encrypt Key wieder in Encryption Key und Permissions zerlegt und Acrobat dadurch angewiesen Nutzungsbeschränkungen zu erzwingen und mit dem Encryption Key die Daten zu entschlüsseln.

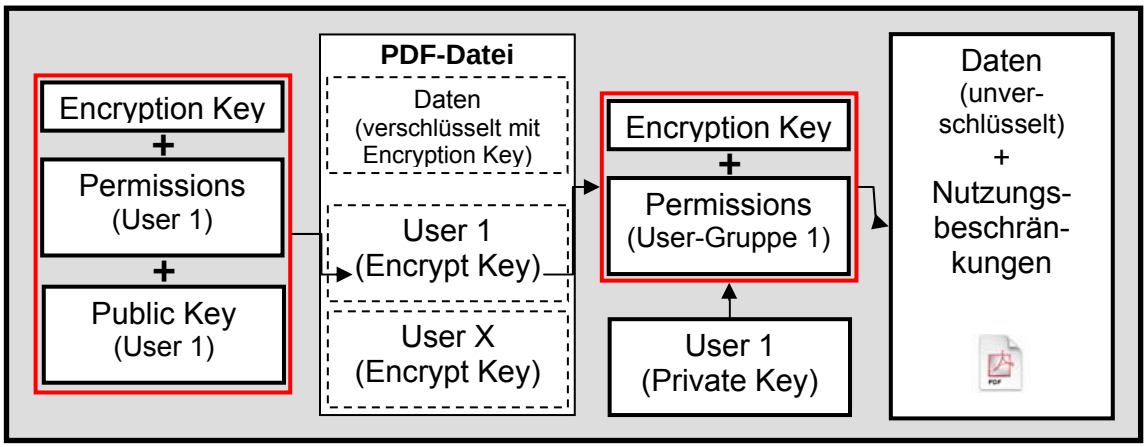

Abb. 6 Einsatz von asymmetrischer Verschlüsselung in Acrobat: Self-Sign Sicherheit

3.4 Hash-Funktion (MD5) als Prüfsummenalgorithmus

Wie schon erwähnt werden eingegebene Passwörter über eine Hash-Funktion in eine Zeichenkette verwandelt (Encryption Key). Acrobat benutzt dazu den MD5 (Message Digest 5) Hash-Algorithmus. Es handelt sich hierbei um die mathematische Erzeugung einer Prüfsumme. Des Weiteren wird die Hash-Funktion in Acrobat auch als Kontrollmechanismus benutzt, um die Integrität von Daten zu überprüfen und um so sicherzustellen, dass sie nicht verfälscht oder abgeändert werden. Die Besonderheit hierbei ist, dass der Ablauf des Algorithmus nur in eine Richtung möglich ist und man keine Chance hat aus einem Hash-Wert Rückschluss auf die Eingabe zu ziehen.

Die grundlegende Funktionsweise zeigt die folgende Abbildung.

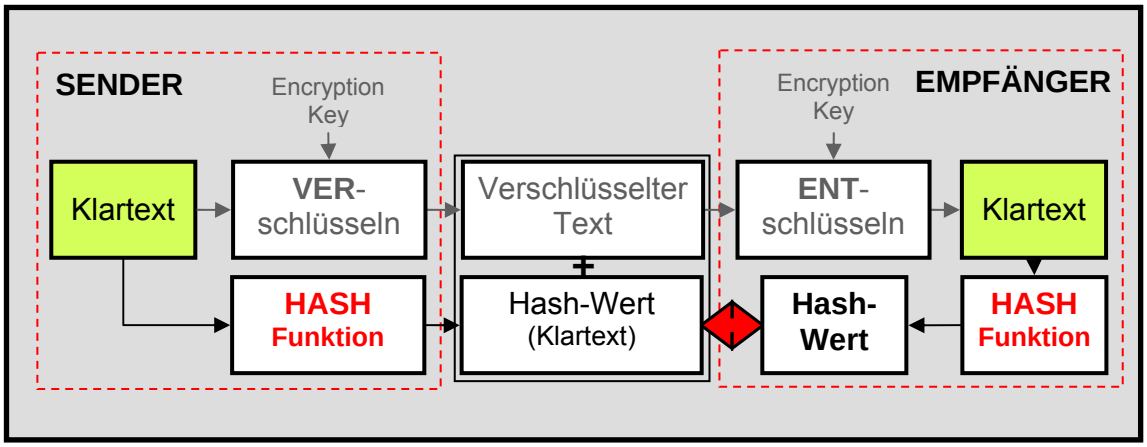

Abb. 7 Einsatz der Hash-Funktion als Prüfsumme

Aus einem Klartext wird ein Hash-Wert erzeugt, der einzigartig ist. Wird bei einem Manipulationsversuch auch nur ein Zeichen im Klartext verändert, ist der Hash-Wert nicht mehr derselbe. Dieser Wert wird als Klartext zusammen mit dem verschlüsselten Text gespeichert. Möchte jemand den Text entschlüsseln, benutzt er den aus seinem Passwort/Private Key erzeugten Encryption Key. Anschließend wird aus dem Klartext wieder ein Hash-Wert erzeugt und mit dem mitgelieferten Wert verglichen. Stimmen sie überein, kann man sicher sein, dass der Text nicht verändert wurde.

Der MD5-Hash-Algorithmus erzeugt immer einen Wert mit 128 Bit Länge, egal wie lang seine Eingabe auch ist. Der Input wird in 512-Bit-Blöcke aufgeteilt. Passt die Nachricht nicht in dieses Blockschema, wird sie so aufgefüllt, dass ihre Länge durch 512 teilbar ist. Aus der Länge des Input wird eine 64-Bit-Integerzahl erzeugt. Diese wird an das Ende des letzten 512-Bit-Block gesetzt. Anschließend werden eine 1 und so viele Nullen, wie nötig sind, eingefügt bis ein Block erzeugt wurde, der in das gewünschte Blockschema passt.
Jeder Block wird dann wieder in 16 32-Bit-Blöcke unterteilt. Diese werden über die Variablen A, B, C und D miteinander verknüpft. Anschließend wird diese Prozedur mit den 512-Bit-Blöcken wiederholt, bis die vier Variablen eine Länge von jeweils 32 Bit besitzen. Diese aneinandergereiht ergeben den Hash-Wert mit einer Länge von 128 Bit.

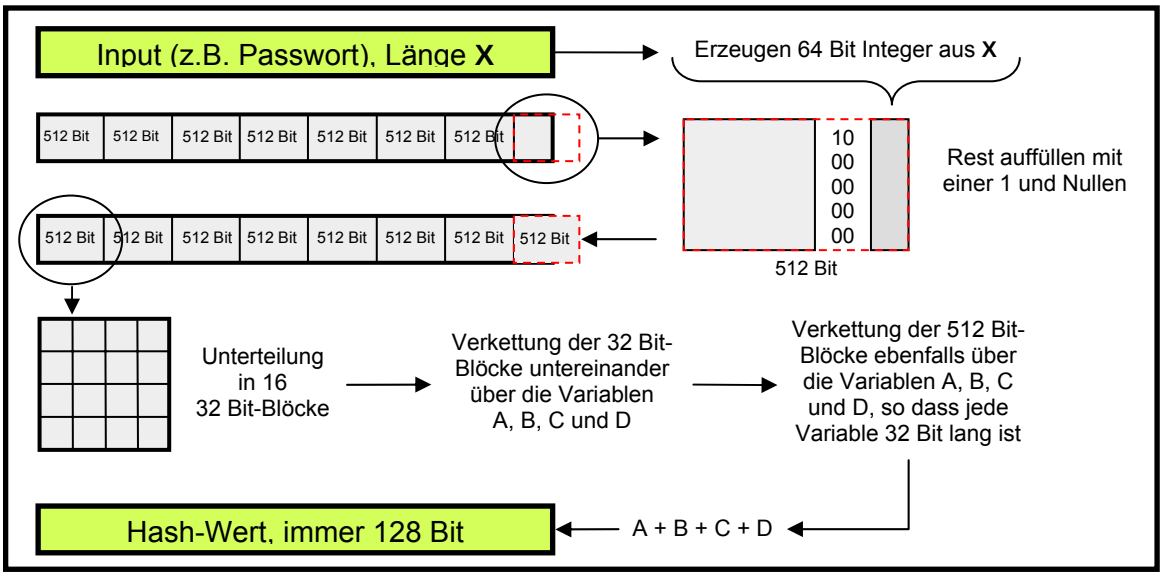

Abb. 8 Funktionsweise der MD5-Hash-Funktion

Bei der Umwandlung zu Hexadezimal werden die Bits in Byte umgerechnet (8 Bit = 1 Byte): 128 Bit : 8 = 16 Byte 1 Byte wird in Hexadezimal durch 2 Stellen ausgedrückt und somit der Hash-Wert als 32-stellige Hexadezimalzahl ausgedrückt.

Die Erzeugung eines Hash-Wertes zeigt folgendes Beispiel:
Eingabe von: *PDFst@r2005!* erzeugt den
Hash-Wert: *8d4d753354e3d51aa43a2af82b2e6283*

Eingabe von: *PDFst**a**r2005!* erzeugt den
Hash-Wert: *541c44a16895c4be2fb43073533df57f*

Man sieht, dass durch das Ändern eines Zeichens (@ in a) ein vollständig anderer Hash-Wert entsteht und nicht nur z.B. der betroffene Bereich der Hash-Summe verändert wird. Das Verfahren stellt somit ein sehr sicheres Prüfmittel für die Integrität von Daten dar.

3.4.1 Einsatz von MD5 in Acrobat

In Acrobat lassen sich Passwörter mit einer maximalen Länge von 32 Zeichen eingeben. Bei der Umwandlung zu Hexadezimal werden 8 Bit bzw. 1 Byte pro Zeichen verwendet. Man erhält somit eine Zeichenkette von genau 32 Byte (32 Byte x 8 Bit/Byte = 256 Bit). Wird ein Passwort eingegeben, dass weniger als 32 Zeichen hat, füllt Acrobat automatisch die fehlenden Zeichen selbständig aus einem hinterlegten Pool auf (28 BF 4E 5E 4E 75 8A 41 64 00 4E 56 FF FA 01 08 2E 2E 00 B6 D0 68 3E 80 2F 0C A9 FE 64 53 69 7A → 32 Hexadezimalzahlen = 32 Byte). Wird kein Passwort vergeben, liefert das den Wert 0 und es werden alle Zeichen aus dem Pool verwendet. Bevor diese Daten aber die Eingabe der Hash-Funktion bilden, fließen noch einige Parameter, wie z.B. das Owner Password (falls vorhanden), Nutzungsbeschränkungen (falls vorhanden) und die ID aus dem document trailer der PDF-Datei, mit ein.
Aus diesen Daten wird der Encryption Key erzeugt.
Zusätzlich nutzt Acrobat die Hash-Funktion auch als Prüfsummencheck, um die Integrität der Daten sicherzustellen und überprüfbar zu machen.

3.5 Verschlüsselung von Daten mittels RC4-Algorithmus

Der RC4-Verschlüsselungsalgorithmus dient als Grundlage der Verschlüsselung für die Daten einer PDF-Datei. Es handelt sich hierbei um einen so genannten Stromchiffrierer, benannt nach Ronald L. Rivest (RC4 → Ron's Cipher 4). Stromchiffrierer verschlüsseln Bit für Bit und erzeugen mit einem geheimen Schlüssel (in Acrobat: der Enrcyption Key) über eine Substitutionsbox einen Schlüsselstrom. Dieser wird dann mit dem Klartext, ebenfalls Bit für Bit, durch die XOR-Funktion (e**X**clusive-**OR** bzw. exklusiv-ODER-Verknüpfung) verknüpft.

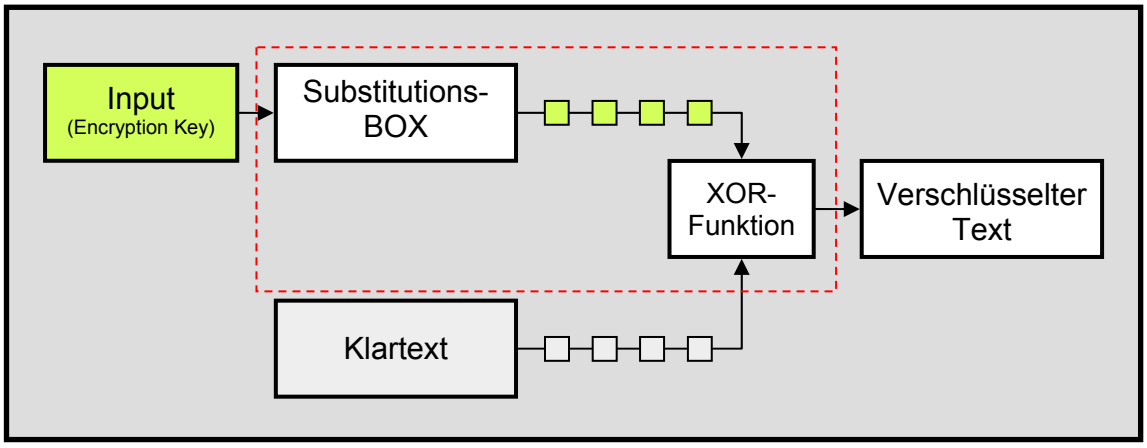

Abb. 9 Prinzip des RC4/ AES Verschlüsselungsalgorithmus

Eine XOR-Funktion vergleicht das eingehende Bit vom Klartext mit dem aus der Substitutionsbox, das auch zum „Verschlüsseln" benutzt wird. Hierbei können zwei Zustände, 0 und 1, erzeugt werden. Ein Ergebnis ist nur genau dann 1, wenn nur einer der beiden zu verknüpfenden Werte 1 ist; sind beide gleich, ist das Ergebnis 0.

XOR-Verknüpfung zweier Bits:
0 XOR 0 = 0
0 XOR 1 = 1
1 XOR 0 = 1
1 XOR 1 = 0

Wegen der Verschlüsselung auf Bit-Ebene ist der RC4-Algorithmus sehr schnell und findet deshalb auch Anwendung in Echtzeit-Systemen.

3.5.1 Einsatz von RC4 in Acrobat

RC4 ist sehr schnell, aber dennoch würde die Verschlüsselung der kompletten Datei zu lange dauern, deshalb wird in Acrobat nur der Inhalt von Strings und Streams, nicht aber die Objektverwaltung verschlüsselt.
Somit ist sichergestellt, dass ein schneller Zugriff auf die unterschiedlichsten Teile der Datei erfolgen kann ohne die gesamte Datei entschlüsseln zu müssen.
Dokumenteninformationen und Lesezeichen werden ebenfalls verschlüsselt. In die Kodierung gehen die Objekt- und Generationsnummer, sowie die Berechtigungen, die dafür vorliegen, mit ein.

3.6 Verankerung der Verschlüsselung im Quellcode einer PDF-Datei

Ohne eine PDF-Datei zu öffnen lässt sich schon anhand des Quellcodes erkennen, ob eine Verschlüsselung vorliegt oder nicht.
Findet sich der Eintrag „/Encrypt“ im Document Trailer, liegt eine Verschlüsselung vor.

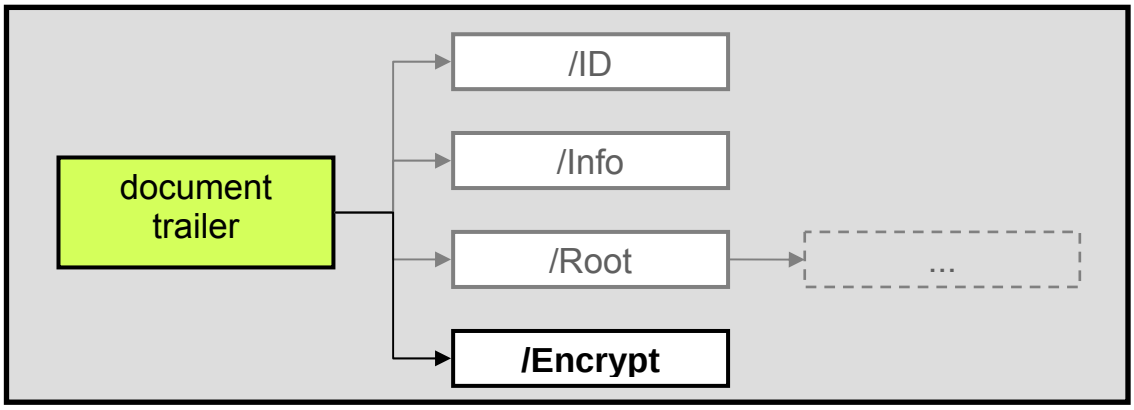

Abb. 10 Verankerung der Verschlüsselung im Quellcode

Wird eine passwortgeschützte PDF-Datei aufgerufen, übernimmt der Security Handler die Aufgabe der Passwortabfrage, die Umrechnung des Passwortes über eine Hash-Funktion und das Vergleichen mit dem gespeicherten Wert.
Acrobats Security Handler ist standardmäßig eingebettet und aktiv, es lassen sich aber auch Security Handler von anderen Anbietern auswählen, wenn diese installiert sind.
Jegliche Kennwörter, Nutzungsbeschränkungen und zusätzliche Informationen für die Verschlüsselung sind kodiert als Hash-Werte im, zum Security Handler gehörigen, Encryption Dictionary näher beschrieben.

Auf die wichtigsten Einträge soll etwas detaillierter eingegangen und werden im Verlauf an einem Beispiel erläutert weiteren.
Die Werte „Filter“ und „SubFilter“ geben an, welcher Ver- und Entschlüsselungsmechanismus (Security Handler) benutzt wird. Ab PDF 1.3 dienen diese Angaben auch zur Unterstützung der Public Key Struktur. Mit „R“ (Revision) kann zusätzlich noch die Revision des Security Handlers angegeben werden. Die Art der Anwendung des RC4-Algorithmus wird über „V“ (Version, Zahlen 0 bis 4) definiert. Die Standardeinstellung von Acrobat 6.0 ist V=3 und steht für eine 40 bis 128 Bit-Verschlüsselung (vor Version 5.0 nur 40 Bit) mit dem Encryption Key, anhängig von der PDF-Version. Optional wird über die Integerzahl „Length“ die Länge des Encryption Key in Bit angegeben.
„O“ und „U“ repräsentieren Owner- und User-Passwort und sind als deren Hash-Wert hinterlegt. Falls Nutzungsbeschränkungen vorliegen, werden diese als 32-stellige Binärzahl ausgedrückt und dann als Dezimalzahl mit „P“ (Permission) angegeben. Jede Stelle der Binärzahl hat eine bestimmte Bedeutung (Wert 0 = deaktiviert, 1 = aktiviert).
Ob die Metadaten auch verschlüsselt wurden, lässt sich über den Eintrag „EncryptMetadata true/false“ feststellen. Der Default-Wert ist „true“. Ist V=4, dann fehlt dieser Eintrag und die Verschlüsselung der Metadaten wird über den RC4-Algorithmus gesteuert.

Beispiel: Quellcode einer PDF-Datei

```
trailer                 % Trailer dictionary
<<
    /Size 95            % Anzahl der Objekte in der Datei
    /Root 93 0 R        % Der Dokumentenbaum hat die object ID (93,0)
    /Encrypt 94 0 R     % Das Encryption Dictionary hat die object ID (94,0)
>>

94 0 obj                % Encryption dictionary
<<
    /Filter /Standard   % Standard Security Handler wird benutzt
    /R 3                % Revision 3 des Security Handlers
    /V 4                % verwendete Anwendung des RC4-Algorithmus
    /Length 128         % Länge des Encryption Key, hier: 128 Bit
    /O (xxx...xxx)      % Hashed Owner Password (32 bytes)
    /U (xxx...xxx)      % Hashed User Password (32 bytes)
    /P 65472            % Permissions als Dezimalzahl
                        % kein EncryptMetadata, weil V=4
>>
endobj
```

3.7 Funktionsprinzip und Aufgabe des Security Handlers

Wie schon erwähnt ist der Security Handler für die Abläufe der Passwortabfrage etc. zuständig.
Ein eingegebenes Passwort wird gehashed und der so entstandene Encryption Key wird an den RC4-Algorithmus weitergereicht. Dort wird der Key mit dem Content der PDF-Datei verknüpft und man erhält die verschlüsselte Datei. Gleichzeitig wird aus dem Content über die Hash-Funktion auch noch ein Hash-Wert erstellt, mit dem später die Integrität der Daten überprüfen wird.

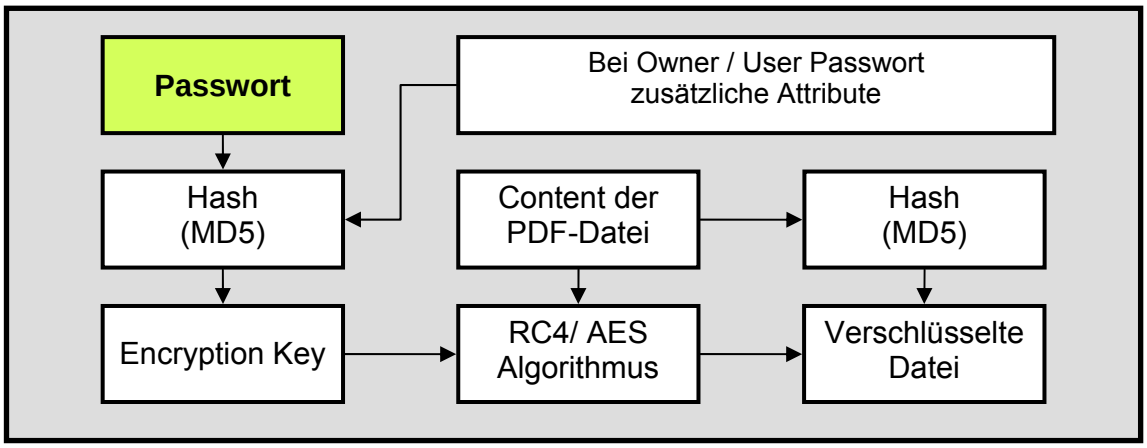

Abb. 11 Verschlüsselung mit dem Standard Security Handler

Die Prozedur lässt sich auch in gewisser Weise wieder umkehren. Wird ein Passwort beim Öffnen eines PDF-Dokuments abgefragt und eingegeben, wird dieses gehashed und somit der Encryption Key erzeugt. Der Key und der verschlüsselte Content werden dann an den RC4-Algorithmus gereicht und dieser entschlüsselt den Inhalt.
Aus dem erzeugten Inhalt wird auch hier ein Hash-Wert errechnet und mit dem gelieferten Wert aus der verschlüsselten Datei verglichen, um zu überprüfen, ob die Daten ggf. verfälscht wurden.

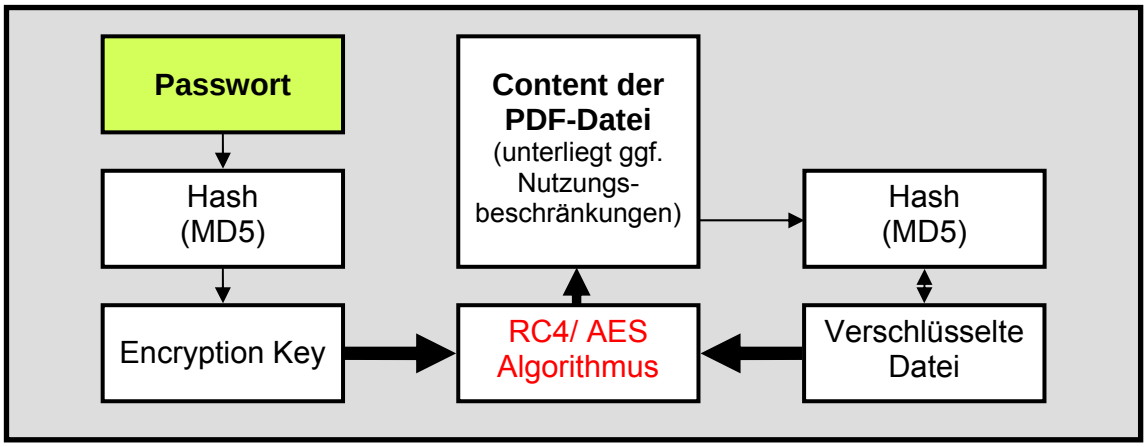

Abb. 12 Entschlüsselung mit dem Standard Security Handler

4. QUALITÄT DER ACROBAT-VERSCHLÜSSELUNG

4.1 Stärken und Schwachstellen

Nachdem die verwendeten Sicherheitsmechanismen etwas näher untersucht wurden, ist es auch möglich, eine Aussage über die Qualität der verwendeten Verschlüsselung zu treffen. Viele Faktoren beeinflussen die Stärke gegen den Zugriff Unbefugter.
Generell liegt die größte Schwachstelle jeder Verschlüsselung in der Qualität des verwendeten Passwortes. Als Faustregel gilt, dass mindestens 8 Zeichen verwendet werden sollten, die möglichst aus einem Mix aus Ziffern, Buchstaben und Sonderzeichen bestehen sollten.
Der Ausdruck *PDFst@2004!* wurde verwendet, um zu zeigen, wie eine Hash-Funktion einen Wert erzeugt, stellt aber gleichzeitig auch ein Beispiel für ein sinnvoll gewähltes Passwort dar.
Eben so wichtig ist aber auch die Länge des Schlüssels, den Acrobat verwendet. Denn es hilft kein noch so gutes Passwort, wenn nur mit 16Bit verschlüsselt wird. Erst ab Acrobat 5.0 wurde die Verschlüsselung mit 128Bit als Standard eingeführt und hat die 40Bit Verschlüsselung abgelöst. Eine Abwärtskompatibilität der Acrobat-Versionen untereinander ist leider nicht automatisch gewährleistet. Ein Acrobat-Nutzer der Version 4.0 kann kein verschlüsseltes Dokument öffnen, dass mit dem Standardeinstellung der Version 5.0 verschlüsselt wurde. Dies geht nur, wenn vom Ersteller bei den Sicherheitseinstellungen eine ältere Version ausgewählt wird.

Wurden beim Speichern der PDF-Datei Nutzungsbeschränkungen vergeben und die Datei mit einem Passwort versehen, regelt, wie gezeigt, der Security Handler die Passwortabfrage beim Öffnen. Das gilt sowohl für Acrobat selbst als auch für den Reader. Wird kein richtiges Passwort eingegeben, ist der Zugriff komplett blockiert. Wird aber das richtige User-Passwort eingegeben wird die Datei komplett entschlüsselt und der Security Handler weist Acrobat an die vom Ersteller gesperrten Funktionen zu blockieren. Die lässt sich aber recht leicht umgehen, z.B. ignoriert Ghostscript (u.a. ein PDF-Conversion-Tool und Viewer) beim Öffnen einfach die Anweisungen zur Nutzungsbeschränkung.

Adobe hat ab Acrobat 4.0 eine einfache, aber effektive Codeanweisung implementiert, die ein erneutes Distillieren einer geschützten Datei verhindern soll. So lässt sich zwar eine PostScript-Datei vom geschützten Original erzeugen, aber beim Distillieren erscheint eine Fehlermeldung. Dieser Schutzmechanismus ist so intelligent eingebaut, dass jede Änderung unweigerlich einen Fehler beim Distillieren erzeugen würde.

4.2 Advanced PDF Password Recovery (APDFPR)

Jede Art von Verbot ruft in einigen Menschen den Wunsch hervor, genau dieses zu umgehen. Und genau da schafft Adobe durch die Veröffentlichung ihres Reference Manuals noch eine kleine Starthilfe. Wieso sollte es nicht möglich sein Passwörter zu knacken bzw. den Mechanismus auszuhebeln, der dahinter steckt, wenn eine Anleitung vorhanden ist.
Die russische Softwarefirma Elcomsoft entwickelte ihr Programm „Advanced PDF Password Recovery“ (APDFPR), mit dessen Hilfe es möglich ist Passwörter von PDF-Dateien zu knacken und Nutzungsbeschränkungen außer Kraft zu setzen.

4.2.1 Funktion

Das Programm bedient sich dazu üblicher Methoden, um Passwörter herauszufinden. Wie schon bereits gehört lässt sich ein Hash-Algorithmus nicht zurückrechnen, d.h. ein Angriff kann nur durch Ausprobieren erfolgen.
Mit der so genannten Brute-Force-Methode werden Passwörter generiert, wobei alle möglichen (druckbaren) Zeichen verwendet werden. Diese werden dann auf die gleiche Weise gehashed, die Acrobat nutzt und verglichen, indem sie dem Security Handler vorgesetzt werden. Diese Methode ist allerdings sehr rechenintensiv und ist nur effektiv für kurze bzw. einfach gewählte Kennwörter (ohne Sonderzeichen, nur Ziffern bzw. Buchstaben etc.).
Eine Verfeinerung davon ist die Dictionary Attack, bei der ganze Wörterbücher verwendet werden, um einfache Passwörter sehr schnell herauszufinden.

4.2.2 Anwendungsgebiet/Beschränkungen

Tests, die mit APDFPR durchgeführt wurden, haben gezeigt, dass sich eine 40Bit-Verschlüsselung problemlos knacken lässt. Allerdings ist hierbei auch die Qualität des Passwortes entscheidend. Eine 128Bit-Verschlüsselung ist derzeit nicht zu knacken, wenn das Passwort entsprechend „gut“ gewählt wurde.
Das beste Beispiel dafür zeigt sich bei einem Test mit dem Programm selbst und einer verschlüsselten PDF-Datei (128 Bit) auf einem Pentium III/900 MHz.
Bei 4 Zeichen rechnet der PC 22 Minuten, bei 5 Zeichen schon 9 Tage, bei 6 Zeichen 42 Tage und bei weiterer Erhöhung um 1 Zeichen bereits schon mehrere Jahre.
Auch das Zusammenschalten von mehreren Computern verringert die Dauer zum Finden des richtigen Passwortes, wenn dieses sicher gewählt wurde, nicht auf eine akzeptable Zeitspanne.

APDFPR kann nur die Standardsicherheit von Acrobat verarbeiten, Security Handler von Drittanbietern können nicht umgangen werden, da hier die Mechanismen geheim gehalten werden. Allerdings sind auch die Möglichkeiten den Standard Security Handler zu umgehen stark eingeschränkt. PDF-Dateien der Version 1.4 und 1.5 mit einer 128 Bit Verschlüsselung lassen sich generell (noch) nicht über die Brute-Force-Attack-Methoden knacken. Einige Dateien (ab PDF 1.5), die mit Acrobat 6.0 erstellt worden sind, können ebenfalls nicht verarbeitet werden.

5. FAZIT

Viele gute Sicherheitsmechanismen schaffen in ihrer Summe leider noch keine absolute Sicherheit. Die verwendeten Algorithmen in Acrobat sind als sicher anzusehen, allerdings sollte man sich der Schwachstellen, die verbleiben, bewusst sein. Diese entstehen sehr häufig schon beim Nutzer. Wie bereits erwähnt sollte man deshalb ein möglichst „gutes“ Passwort wählen (Mix aus Zahlen, Buchstaben und Sonderzeichen), das eine Länge von mindestens 8 Zeichen besitzt. Dies ist vor allem wichtig gegen Angriffe mit Programmen, die, wie beispielsweise APDFPR, mit der Brute-Force-Methode arbeiten, um den Encryption Key ausfindig zu machen. Auf diese Weise kann unter gewissen Umständen die Verschlüsselung einer PDF-Datei ausgehebelt (mit APDFPR nur bei Einsatz des Standard Security Handlers) werden. Wichtig in diesem Zusammenhang ist es auch, darauf zu achten, dass das Dokument mit einer Schlüssellänge von 128 Bit kodiert wird. Eine RC4-Verschlüsselung mit 128 Bit ist seit Acrobat 5.0 Standard und sollte Nutzern einer neueren Acrobat-Version daher keine Probleme bereiten, wobei jedoch auf eine eventuellen Abwärtskompatibilität zu achten ist.
Auch sollte man sich über die möglichen Komplikationen bei der Festlegung von Nutzungsbeschränkungen im Klaren sein. Die Objektverwaltung der PDF-Datei bleibt unverschlüsselt, es sind nur die Strings und Streams verschlüsselt, die beim Öffnen entschlüsselt werden. Das bedeutet, dass beim Öffnen und nach Eingabe eines Passwortes/Public-Keys die komplette PDF-Datei entschlüsselt vorliegt und nur der PDF-Viewer angewiesen wird, bestimmte Funktionen zu deaktivieren oder zu unterdrücken. Mit dem Acrobat Reader funktioniert dies tadellos, mit Software anderer Hersteller könnten Probleme bei der Einhaltung der Nutzungsbeschränkungen auftreten.

Für größere Firmen eignet sich eine Public-Key-Infrastruktur besonders gut. Mit dem Self-Sign-PlugIn lassen sich dadurch viele Nutzer einfach und sicher verwalten. Leider funktioniert dies nur mit dem Acrobat Vollprodukt und ist sicherlich dadurch auch eine Kostenfrage. Mit der LifeCycle-Produktfamilie bietet Acrobat neuerdings auch die Möglichkeit an, dass die Verwaltung der User-Zertifikate und die Verschlüsselung zentral auf einem Server stattfinden können.

Die Verwendung einer Sicherheitslösung von anderen Anbietern (z.B. Entrust oder Authentica) gilt bislang jedoch als am sichersten. Da Adobe die Acrobat-Spezifikation offen gelegt hat, lassen sich diese PlugIns mühelos implementieren und bieten aufgrund der Geheimhaltung der verwendeten Mechanismen keine große Angriffsfläche für einen unbefugten Zugriff.

6. LITERATURVERZEICHNIS

PDF Reference, fifth Edition, Version 1.6, Adobe Systems Incorporated, November 2004

Die PostScript- & PDF-Bibel, Thomas Merz, Olaf Drümmer, PDFLib GmbH, 2002[2]

Wikipedia – Die freie Enzyklopedie, http://de.wikipedia.org

Elcomsoft Co.Ltd., Informationsmaterial über Advanced PDF Password Recovery, zusätzlich: http://www.elcomsoft.com