Digitally Signed and Permission Restricted PDF Files: a Case Study on Digital Forensics

Patricio Domingues
ESTG - Polytechnic Institute of Leiria
Instituto de Telecomunicações
Centro de Investigação em Informática e Comunicações
Leiria, Portugal
patricio.domingues@ipleiria.pt

Miguel Frade
ESTG - Polytechnic Institute of Leiria
Centro de Investigação em Informática e Comunicações
Leiria, Portugal
miguel.frade@ipleiria.pt

ABSTRACT

The PDF format is the de-facto standard for many types of documents. Often a forensic digital investigation is faced with a significant volume of PDF files. It is thus important to filter PDF files, giving priority to files that have an high probability to carry important and meaningful data. In this paper, we focus on identifying potential important PDF files, selecting i) digitally signed files and ii) files that have special owner restrictions set, such as interdiction to assemble/separate pages. For this purpose, we present the python-based `digiSign|protected`PDF module for the open source Autopsy forensic software. When run over a digital forensic data source, the module creates two lists: one holding the digitally signed files and, another one with files that have special restrictions in their usage. To study the occurrence of digitally signed and of permission-protected PDF and their importance for digital forensics, we analyzed a Windows 10 forensic image, finding that 2.81% of the PDF files were digitally signed and 3.75% were permission-protected. The study shows that digitally signed PDF files can harbor meaningful data for a digital forensic investigation.

CCS CONCEPTS

• **Applied computing** → **System forensics**;

KEYWORDS

PDF files; Digital forensics; Digital signatures; permission-protected PDF files

ACM Reference Format:
Patricio Domingues and Miguel Frade. 2018. Digitally Signed and Permission Restricted PDF Files: a Case Study on Digital Forensics. In *ARES 2018: International Conference on Availability, Reliability and Security, August 27–30, 2018, Hamburg, Germany*. ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3230833.3232811

ARES 2018, August 27–30, 2018, Hamburg, Germany

ACM ISBN 978-1-4503-6448-5/18/08…$15.00
https://doi.org/10.1145/3230833.3232811

1 INTRODUCTION

In the last decades, the world has clearly gone digital and online. Many traditional services have migrated to digital, while new ones, purely digital such as social networks and video streaming services, have emerged [37]. In part, this is made possible due to easy access to Internet, with high bandwidth networks and ubiquitous affordable devices such as smartphones, netbooks and laptops helping users to move to online services. Also contributing to this trend, is the fact that companies and state services are keen to push customers to digital channels, since this usually reduces costs and speed up operations. Consumers are also attracted to digital/online services since they often can be used anywhere, and anytime, avoiding physical travels and waiting queues.

This digitization trend is stressing digital forensic examinations. Indeed, many investigations require digital forensics due to the existence of multiple digital devices such as mobile phones, smartphones, tablets and laptops that might hold valuable data for the investigation. On top of that, mobile devices allow users to access and create large amount of data. In fact, the steady increase on storage capacity and bandwidth in regular devices, as well as, the dropping prices of storage and connectivity, originate large volumes of data. Quick & Choo [26] state that the large volume of data of current cases has three main causes: *i*) an increase in the number of seized devices; *ii*) rise in the number of cases where digital devices need to be analyzed; *iii*) increase in the size of data on each individual item. All of this contributes to increase the workload for digital forensic investigations, and consequently lengthen backlogs [13, 26].

Data found in digital devices are not all equally important for a digital forensic examination. Many files have no relevant data for investigators, while other files, often a minor subset, hold information that might be relevant to the investigation. This work focuses on Portable Document Format (henceforth, PDF) files that have one or two of the following properties: *i*) are digitally signed and *ii*) are published with special restrictions in their usage, such as forbidding pages disassembly/assembly. The rationale behind our approach to favorite digitally signed and permission-restricted PDF files lies in the fact that they have a higher probability of holding meaningful data than regular PDF files in some investigations. This is especially true for digitally signed PDF files, since digitally signing a document engages the responsibility of the signer/ees, as it holds, in some countries [10], the same legal value of a physical hand signature. Additionally, a digitally signed PDF file embeds, at least, the digital certificate of the signer/ees, which provides for some identification, a timestamp and other data. For instance, the

existence of multiple digitally signed documents addressed to the same recipient is a strong indication that the recipient might be the owner of the digital device under analysis. This the case, for example, of digital invoices from utilities (electricity, water, etc.), since many of the companies providing these services are digitally signing the invoices that they provide to their customers. Similarly, producing a PDF document sets with restricting permissions often attest the importance that the publisher attributes to the content.

To the best of our knowledge, this is the first time that digitally signed and/or permission-restricted PDF files are studied expressly for digital forensics. We believe that the main contributions of this paper are as follows:

- Highlighting the importance that digitally signed PDF files can have in digital forensics, namely since they provide for identity, timestamps, and data that are considered important enough to deserve a digital signature.
- Development of the `digiSigned|Protected`PDF module for the digital forensic software Autopsy. The module analyzes a forensic image and creates two lists of PDF files: one holding the PDF files that are digitally signed and another one listing the PDF files that have restricted permissions. The module is available under an MIT open source license.
- Case study with the analysis of the PDF files existing on a given Windows 10 personal computer, quantifying the occurrences of PDF files that are either digitally signed or permission-restricted.

The remainder of this paper is organized as follows: Section 2 briefly introduces digital signatures. Section 3 reviews the PDF format, focusing on *i*) digitally signed files and on *ii*) permission restricted files. Section 4 reviews related work, while Section 5 presents the developed `digiSigned|Protected`PDF module for the Autopsy software. Section 6 discusses the study of PDF files found on a given Windows 10 PC. Finally, Section 7 concludes the paper and presents future work.

2 DIGITAL SIGNATURES

Due to its uniqueness, hand made signatures have always been a respected authentication mechanism. A physical signature uniquely identifies the signer and endorses his/her agreement of the signed document. Even if forgeries are possible and are often attempted, physical signatures are considered a valid mean of authentication. For important acts, signing of documents usually requires an appropriate signature protocol, like, for instance, having accredited officials to witness and register the signatures, acting as a notary service. Therefore, in the physical world, huge importance is given to hand signed documents.

In the digital world, the equivalent of a physical signature is the so called *digital signature*. A digital signature is the result of a set of mathematically-based and cryptographic procedures which unequivocally links the signer and the signed document. This provides for authentication, non-repudiation and integrity [27]. The signer is authenticated since she/he is the only one that can sign with her/his digital identity. The act of signature often requires a physical token held by the signer – e.g., a smartcard ID or a identifying token – and a given knowledge like, for instance, the pin code of the smart card ID needed to digitally sign an electronic document. Like physical signatures, digital signatures provide for non-repudiation, impeding the signer from unilaterally canceling a signed document. Digital signatures also insure the integrity of the signed document in such a way that changes to a signed document are properly reported, at least when the signature is verified. This way, if properly applied, digital signatures are, at least, as trustful as physical ones.

It is important to point out that digital signatures are a class of electronic signatures. Indeed, digital signatures rely on sound cryptography, namely hashing algorithms and public-key infrastructure (PKI), while the designation *electronic signature* has a broader meaning and might have weaker on non legal value. Examples of electronic signatures include the name of the sender of an email typed at the end of a message, or a digitized image of the physical signature overlaid on a document [5]. Digital signatures are actually a restricted subset of the electronic signatures universe. In this work, we solely consider digital signatures in the strict interpretation of the designation, i.e., legally binding signatures.

Digital signatures were first established by Rivest et al. in their seminal work [27]. The adoption of digital signatures for documents varies widely across the globe. For instance, in the Europe Union (EU), where digital signatures have the same legal value of notary supervised ink signatures since 1999, as defined in the eSignature Directive (1999/93/EC), adoption rate is low and varies considerably from country to country. For individuals, this can be a consequence of the widely different approaches taken by each country regarding the administrative document, if any, used for identification by their citizens. While some countries such as Ireland and the UK do not require their citizens to have ID card, others like Estonia [1] and Portugal [14] have mandatory electronic ID smartcards, which can also be used to digitally sign documents. For this purpose, the card bearer needs to enable digital signatures [8]. Data indicates that as in 2014, as much as 45% of Portuguese ID card holder have the digital signature option activated, and therefore can digitally sign documents [14]. As reported by Anthes [1], Estonia estimates a cost saving of around 2% of GDP ($500 million a year) due to the wide usage of digital signatures. To tackle the low adoption of digital signatures, the EU has since July 2016 replaced the *eSignature* directive with the eIDAS regulation with the goal of allowing EU citizens to use digital services anywhere in the EU, by standardizing trust services such as digital signatures, certificates and preservation of digital ID and documents [10]. Companies and institutions that preferentially interact with their customers through online channels – web, emails,etc. – have adopted digital signatures, namely for producing legally binding invoices.

2.1 Anatomy of digital signatures

The typical usage of a digital signature has three main stages: 1) Generation of the keys; 2) Signing; 3) Verification. The 1^{st} stage corresponds to the creation of public/private pair of keys and of the corresponding public certificate. The 2^{nd} stage is the signing act, where the signer(s) uses the private key to sign the document. Stage 2 is performed every time a document is digitally signed. Finally, stage 3 allows for the validation of the signature. This stage occurs whenever someone needs to verify the signature.

Digital signatures relies on asymmetric cryptography and on digest hashing functions. Asymmetric cryptography involves two keys: a private key and a public key [32]. The private key can only be known by the owner, otherwise security is lost. On the contrary, the public key can be released to the whole world. The two keys form a pair, with the public key being derived from the private key. The keys have a complimentary role: when one key is used to encrypt data, only the other key can decrypt them. For instance, data encrypted with the private key can solely be decrypted by resorting to the public key, and vice-versa. For digital signatures, the signer uses his/her private key to encrypt the file to sign. This way, anyone with the public key can attempt to decrypt the file. If the decryption is successful, it means that the file was indeed signed by the private key owner. In practice, it is not the whole file which gets encrypted, but solely an hash digest of the content of the file. Indeed, it is pointless to encrypt the whole file, as it only would waste computational resources for the encrypt/decrypt operations, without providing any security, since anyone with the public key could decrypt the file. Worse, any reading operation of the signed file would require a costly decrypt operation, as asymmetric cryptography is computationally demanding. By resorting to an hash digest function, the encrypt/decrypt operations are much faster, since they operate over a small and fixed-size hash value and not on the whole file. To validate the signature, the validating software acts as follows: *i*) it computes the hash value of the signed file; *ii*) it uses the public key to decrypts the hash value computed by the signer; *iii*) it compares the values obtained in steps *i*) and *ii*). It both values match, the signature is valid, else, it is flagged as invalid.

Three of the most well known digital signature algorithms are RSA (Rivest, Shamir, & Adleman) [32], DSA (Digital Signature Algorithm) [32] and ECDSA (Elliptic Curve Digital Signature Algorithm) [21]. Examples of hashing functions are Message Digest 5 (MD5), Secure HAsh 1 (SHA1), and SHA-2 hashing functions (SHA224, SHA256, SHA384 and SHA512). Due to weaknesses that can lead to collisions, both MD5 [35] and SHA1 [34] are no longer considered secure. The current recommended hash algorithm is SHA2-256 [2].

2.2 Public-Key infrastructure

A final issue remains: how to associate a public key with an individual or organization in a way that it can be trusted, minimizing the risks of stolen identities? This is achieved through the so called public-key infrastructure (PKI) [9, 38]. First, a digital certificate needs to be created. It contains the name and possibly other identification items of the signer, his/her public keys. However, for the certificate to be valid and thus be accepted, it needs to be signed by a certificate authority (CA). The CA needs to be a reputed entity, endorsed by a legal or government entity, with a proper protocol to assert the identity of anyone who requests a certificate. In practice, there can be several levels of CA that act as follows: a top level CA issues a signed certificate that authorizes an intermediate entity to act as an intermediate CA. The intermediate CA is responsible for effectively checking the identity of the individual/organization who requests a signed certificate. This way, a chain of trust is created: the individual/organization/site's certificate is signed by the intermediate CA, which in turn, has its own certificate signed by an higher level CA. To allow for a proper validation of a digital signature, all certificates that comprise the chain of trusts of the certificate that represents the signer needs to be available whenever a request is made to validate the signature. For this purpose, the chain of certificates can be embedded in the signed file, depending on whether the file format and associated software support this feature or not. A public key certificate has an expiration date, after which the certificate is no longer considered valid.

The X.509 standard regulates the format of public key certificates that are used in public-key cryptographic infrastructure [9], not only for digital signatures, but also for encrypting communication channels, like for instance for HTTPS and SSH. A digital certificate encompasses several fields of data. Among other elements, a certificate holds the name and/or identification of the bearer, the start date and end date of validity, the company/organization which has emitted the certificate, i.e., the CA, the digital signature of the CA on the certificate. Note that the emitting company/organization is vouching for the authenticity of the information that exists in the certificate. It does so by signing the certificate.

For various reasons, it might be required to declare a certificate as no longer valid. For instance, the private key has been breached or lost. For this purpose, a certificate can be revoked. This can be achieved through so called *certificate revocation lists* (CRL), which, as the name implies, list the certificates that have been revoked. Note that the CRL lists only contain certificates that have not yet reached their expiration date. Certificates that follow the X.509 standard announce the link where the corresponding CRL can be obtained [9]. An alternative to CRL is the Online Certificate Status Protocol (OCSP) [31]. OSCP is a client-server protocol, where the client can request the validation of a X.509 certificate to a given OCSP server. The certificate under scrutiny is identified by its serial number. The OCSP server should respond with one of three possible value: *good*, *revoked* and *unknown*. For a OCSP response to be considered definitive, it needs to be digitally signed by an entity that is trusted by the OCSP client. Further details of OCSP are given in [31].

3 THE PORTABLE DOCUMENT FORMAT

The first version, labeled as 1.0 of the Portable Document Format definition was published in 1992 [20]. In 1993, Adobe released the Acrobat Reader software, priced at $US 50 per user. However, the PDF format only gained traction with version 1.1, when the Adobe Reader software was made available at no cost. In 2008, version 1.7 of the format was standardized, labeled as ISO 32000-1 [17]. More recently, Version 2 of the standard was published in July 2017 under the reference ISO 32000-2 [18].

A critical feature to the success of PDF has been its rendering fidelity, where a given document maintains its appearance regardless of the underlying hardware and display devices. To ensure compactness and thus facilitate exchange and storage of files, PDF relies on several compression algorithms. For instance, for embedded images, the JPEG compression algorithm is used, while text is compressed with LZW-based algorithm [18].

Over the years, a large ecosystem has grown to support the needs of PDF users. For instance, a large number of tools exists for visualization, printing and to perform special operations such

Table 1: Available Permissions in Owner Password Mode

Permission	Description
Annotate	Add or modify comments
Assemble	Assemble the document
Copy	Copy text and images
Extract	Extract text and images
Fill forms	Fill in interactive forms
Modify	Modify the document
Print	Print
Sign	Digitally sign the document

as extraction of pages, rotation, and file merging. Often, software directly supports the creation of PDF output. For software that do not explicit produce PDF output, it is often possible to use a PDF printer driver to still produce PDF output. In some cases, it is the device itself that renders its output in PDF format, as it happens with scanning machines. Therefore, it is without surprise that the PDF format is ubiquitous and universally used, regardless of the hardware platform, the operating system and software. All of this has contributed for the PDF format to become the de-facto standard for document exchange and publishing[7].

3.1 Structure of a PDF file

According to the standard, a PDF document is a data structure comprised of a small set of basic types of objects, such as numbers, strings, names, arrays, dictionaries and streams. The basic object types are used to represent the components of the PDF document, namely, pages, fonts, annotations and so forth [17]. As noted by Castiglione et al. [7], a PDF has the following main components: *i*) a header with the PDF 4-byte magic sequence, followed by the version identifier; *ii*) one or more *body* sections holding the collection of objects that form the document and effectively describes how the document should be rendered; *iii*) one or more tables that cross reference the objects of the document; *iv*) one or more so called *trailers* that locate the cross reference tables within the file. Finally, a PDF file should end with the sequence `%%EOF`. The 4-byte magic number is the hexadecimal sequence `25 50 44 46` corresponding to the ASCII string `%PDF`. Following the 4-byte sequence, is the version of the PDF file, again expressed in ASCII, obeying to the format *dash, major version, dot, minor version*. For example, for a 1.4 version PDF file, the whole sequence is `%PDF-1.4`.

3.2 Permissions and restrictions of PDF files

To preserve confidentiality, the PDF standard allows for the encryption of a PDF document. For this purpose, the creator supplies a so-called *user* or *open* password, which is used to encrypt the data of the file. The encryption algorithm depends on the PDF version. For instance, the standard mandates that a 2.0 PDF file can only be encrypted with the AES256 algorithm or superior [18]. To open an encrypted PDF file, the reader needs to supply the proper credentials.

Besides the document encryption mode, the PDF standard also supports the creation of a PDF file under an *owner* or *permission* password mode. This mode does not prevent the opening and, hence, the visualization of the document. However, it allows the creator of the document to restrict certain operations on the document, thus effectively setting permissions on the document usage. Examples of restrictions that can be imposed in *owner* mode include printing, assembling/disassembling, filling forms, copying or extracting content, digitally signing and modifying the document. It is important to note that owner's set permissions need to be enforced by software to be effective. However, as pointed out in the standard, nothing precludes a PDF viewer software to ignore the owner-defined permissions, since the software has access to the whole PDF document [18]. This is the case for some PDF software. There is also software and online services that remove the owner password, effectively voiding any restriction set through this mechanism. Table 1 lists the permissions that can be configured through the owner password mode.

In this work, for the owner-defined permissions PDF files, in this work, we only tag as relevant PDF documents that have the *assembly* and/or the *modify* permission inhibited. The rationale is that a creator sets one or both of these permissions for documents that he/she might consider important. As we shall see later on, this behavior is the one adopted and coded in `digiSigned|Protected`PDF software module.

3.3 Digital Signatures in PDF

Initial support for digital signatures was added in PDF version 1.3, released in 1999, and has been updated to newer algorithms and wider keys [28]. Specifically, PDF version 1.7 supports the RSA and DSA [17] digital signature algorithms, while support for ECDSA has been formally included in version 2.0 of the standard [18]. For hashing purposes, the PDF format supports SHA1 (since version 1.3), SHA256 (since version 1.6), SHA384 and SHA512 (both since version 1.7) [28].

In a signed PDF file, as described in [28], the data of a digital signature is kept in a special object, a *signature dictionary*. The dictionary hosts several fields, such as the signature handler, the signature format, the type of signature, and the byte range. Note that the byte range identifies the blocks of the PDF files that were considered for the digital signature. The byte range is defined by four integer numbers, with each pair of numbers defining a block. Specifically, the first number of a pair defines the offset of the start of the block, while the second number of the pair defines the size of the block. The existence of two pairs allows for the inclusion of two blocks of bytes.

Others dictionaries involved in a digital signature include the field dictionary, the certificate dictionary and the seed value dictionary. The seed value dictionary holds several important attributes such as the digest method, the timestamp and a reference to the dictionary that holds the signer's certificate. The certificate dictionary contains the fields of the public certificate, such as the type and the subject. A detailed description of the structure of digital signatures in PDF file and of the process of signing a document, is given in [28].

4 RELATED WORK

Several techniques and methodologies for digital forensic *triage* have been proposed and tried with the goal to speed digital forensic

examinations, and thus reduce the backlogs of digital forensic examinations. Roussev et al. argues that actual tools for digital forensic examinations fail to focus on performance, namely latency, as a top-level concern [29]. The authors benchmark several open source tools focusing on their ability for usage in triage operations. For ExifTool, they measure a processing rate of 4.8 MB/s in a multicore machine and a ext4 Linux filesystem. Shawn and Browne argue that *triage* is poorly defined and poorly understood [33]. We believe that identifying PDF files that are digitally signed or have owner permissions restricted can be part of a *triage* protocol, namely in digital forensic examinations related with white collar crime.

Castiglione et al. [7] analyze PDF from a privacy and security point of view. In particular, they focus on content deleted by the creator of a PDF file, but which might still exist on the PDF if the saving is done in a incremental manner. They rightly argue that the apparently deleted content that is still recoverable can be useful in digital forensics. The authors analyzed 35 000 PDF files downloaded from internet, finding out that more than 10% of the documents have two or more cross-reference tables, meaning that there were edited within a PDF editor [7].

Garfinkel [12] proposes the Forensic Feature Extraction (FFE), which purpose is to extract information from emails. The author states that FFE can be used to identify the primary user of a computer. Digital signatures have a similar potential, since a signed document holds a public certificate, which if properly created, identifies the signer. Moreover, invoices digitally signed by companies may also help to identify the owner of the analyzed digital device.

As stated earlier, we are not aware of any other work that proposes the use of qualified digital signatures in the context of digital forensics. Pan and Chena report on signature verification for digital forensics[24], but they do so for ink signatures. Specifically, they resort to a Support Vector Machine (SVM) algorithm to distinguish between real and fake handwritten signatures. Digital signatures can easily be checked and thus are immune to traditional imitation forgeries. However, if the private key is compromised, it is trivial for fraudsters to digitally sign documents that will be indistinguishable from regular ones and hence considered as valid. Private keys can be compromised due to data breaches or due to deficient technology such as in the ROCA case [23]. In ROCA, an algorithmic flaw in a library allowed attacks on public RSA keys to recover the respective private keys. The flaw affected, among others, the electronic identity cards of Estonia, since the flawed library was running inside the smartcard used to generate the pair of private/public keys. This forced the revocation of the vulnerable public key certificates and the issuance of new ones. Other examples of implementation problems leading to weak cryptographic keys are given in [4, 36].

Jansma and Arrendondo [19] compare the strength and performance of the Elliptic Curve Cryptograhpy (ECC) and the RSA for digital signatures. They report that for signature verification, RSA scales better than ECC. However, regarding security, ECC provides the same level of security of RSA with smaller key length [19], and thus less space overhead. The growing dependency on mobile devices to access online services increases the need for digital identification and signatures. Liu et. al. [22] study the performance and energy consumption of two middle-range Android-based mobile devices when dealing with digital signatures. The authors assess both the signature generation and the verification process. They conclude that current middle-range devices can deal with signatures. Furthermore, they observe that signature verification is more expensive, time and energy-wise, than signature generation. Finally, Roy and Karforma analyze the scientific literature on digital signatures, producing an extensive survey [30] .

5 THE DIGISIGNED|PROTECTEDPDF SOFTWARE MODULE

5.1 Autopsy software

Autopsy is a desktop software for digital forensic analysis[1], available under an Apache 2.0 open source license. The software has several functionalities tailored for the forensic analysis of digital content. Autospy aggregates many functionalities from other software tools and libraries, merging them under a productive graphical user interface. A significant part of Autopsy sits on top of the Sleuth kit (STK), a powerful platform for processing digital forensic content. Apart from dealing with the main types of file systems – FAT, exFAT, NTFS, ext3, ext4, etc. – STK also supports the digital forensic data formats E01 and AFF. It also provides services specifics to forensic analysis, such as processing several types of metadata and identifying the operating systems existing in digital forensic contents. Other tools and libraries used by Autopsy are RegRipper [6] for the analysis of Windows OS registry and PhotoRec [15] for recovering content marked as deleted and to process unallocated space.

The functionality of Autopsy can be extended through specifically developed external modules. For this, Autopsy exposes an application programming interface (API) for the Java and Jython programming languages. Jython corresponds to the Python programming language executed within the context of a java virtual machine. The Autopsy software supports three types of modules: *i*) File ingest; *ii*) Data source ingest; *iii*) Report. A file ingest module defines a method which can be called for every file of the forensic image being analyzed. It can save the results as artifacts in Autopsy. A data source ingest module is roughly similar to file ingest, except that it deals with data source. Unlike file ingest, a data source ingest module cannot access carved files and files that are stored in zip archives. Finally, a report module creates a report, expressed through an HTML or/and Excel file, focusing one or several aspects of the analyzed data. An incomplete list of 3^{rd} party modules available for Autopsy is given in [3]. As we shall see in the next section, `digiSigned|Protected`PDF is a file ingest module.

5.2 The `digiSigned|Protected`PDF module

The `digiSigned|Protected`PDF [25] is an Autopsy module that identifies and list the PDF files that *i*) are digitally signed or that *ii*) have one or both of the *document assembly* and *document modify* owner mode permissions deactivated. The module is developed in Jython and uses the *file ingest* methodology provided by the Autopsy software. This methodology triggers an event for every PDF file found on the analyzed forensic image. The PDF file is then tested for the existence of digital signatures and the owner permissions are assessed. If the document is found to be digitally signed, it is flagged as such and added to the list of digitally signed documents.

[1] http://www.sleuthkit.org/autopsy/

Table 2: Most Relevant Status Code (verifier/`digiSigned|ProtectedPDF`)

Code	Description
0	Valid signature
10	No signature
20	No revocation information found
30	Timestamp token is invalid
40	No timestamp token (signature date/time is from the signer's computer)
50	Invalid signature according to OSCP
60	Certificate cannot be verified
61	Certificate expired
62	Certificate not yet valid
63	Certificate revoked
64	Certificate has unsupported critical extension
65	Invalid state. Possible circular certificate chain.
66	Certificate has an unspecified problem
70	Some content is not signed

Next, the PDF document is tested for owner permissions, and if one or both of the *document assembly* or *document modify* permissions are restricted, the file is added to the corresponding list of restricted owner permissions.

The module resorts to two external open source applications: `JSigner`[2] and `ExifTool` [16]. Next, we briefly describes each of these external tools.

5.2.1 JSigner. `JSigner` is a software suite, developed in JAVA, that can be used to digitally sign PDF documents, as well as, to detect and validate digitally signed documents. The detection and validation is done through the `verifier` command line application. The `JSigner` software is quite popular, averaging more than 1000 downloads per week. The `digiSigned|ProtectedPDF` module relies on the `verifier` application to detect digitally signed PDF documents and to assess the soundness of the signature(s). For instance, for digitally signed documents, along with the identification of properly signed documents, the `verifier` application can signalize the documents whose signature(s) has an invalid certificate (e.g., the certificate is not yet valid) or that have no timestamp token. The most relevant status codes that are returned by the `verifier` application and mapped by `digiSigned|ProtectedPDF` are shown in Table 2.

5.2.2 ExifTool. ExifTool is a command line software tool that deals with metadata of files [16]. ExifTool is developed by Phil Harvey, who makes the tool available under the same open source license as PERL. ExifTool can read the metadata of several file formats, and can also modify or erase some metadata. Although the tool has its main focus on metadata related to image files, namely EXIF metadata (EXchangeable Image file Format), hence its name, ExifTool has support for many other file formats. Regarding PDF files, ExifTool can process the common metadata – file size, MAC timestamps, etc. – and PDF-specific metadata. Specifically, ExifTool can report the PDF owner permissions using the label shown in Table 1. This functionality is used by the `digiSigned|ProtectedPDF` module. Although ExifTool is written in the PERL programming language, a Windows stand alone executable, created with the `Perl2Exe`, exists. This way, the tool does not require a PERL installation to run. Similarly, an installation package of ExifTool exists for macOS devices, while the PERL version can be run on Linux.

The fact that both external tools and Autopsy support the main desktop OS platforms – Windows, macOS and Linux – allows `digiSigned|Protected`PDF to be used on any of the main desktop platforms supported by Autopsy.

[2] http://jsignpdf.sourceforge.net

Table 3: Version of PDF files in PDF_{W10}

Version	Count	%
v1.1	67	0.64%
v1.2	125	1.19%
v1.3	874	8.34%
v1.4	5315	50.70%
v1.5	2692	25.68%
v1.6	803	7.66%
v1.7	607	5.79%
Total of PDF files	10 483	100%
Total of files	387 781	–

6 ANALYSIS OF PDF FILES: CASE STUDY

In this section, we analyze the prevalence of PDF files in a Windows 10 machine, categorizing the PDF version of the files, the percentage of digitally signed PDF documents and the owner-restricted permissions set on the files. We identify the dataset of PDF files as PDF_{w10}.

6.1 Digitally Signed Files

The files were collected from a desktop PC and represent 221 GiB of occupied storage. Data regarding the number and versions of PDF files are shown in Table 3. Specifically, PDF_{W10} has a total of 387 781 files, of which 10 483 are PDF files, that is, 2.70% . Table 3 shows the frequency of PDF PDF_{W10}'s files grouped by version.

Table 4 presents statistics regarding digitally signed PDF files of the dataset PDF_{W10}. A slight bit more of 2.80% (294 out of 10 483) of the PDF files are digitally signed. Out of these 294 digitally signed files, 163 present some kind of issue relative to the digital signature. Specifically, 18 have an invalid timestamp token, 95 used the signer's computer timestamp (and not an external timestamp server), 7 have invalid OCSP data, 41 cannot be verified, while one file has an unsupported extension. Finally, a file is flagged as having unsigned content, because it was slightly altered – an annotation was added – after having been signed. This is another interesting feature of digitally signed PDF: the ability to rollback a PDF file to obtain the version that was effectively signed. It is important to note, that as much as 55.44% of the analyzed PDF signed files have some kind of issue with the digital signature. The most prevalent issue is the non-usage of a timestamp server, relying instead on the signer's computer clock.

For a better knowledge of what kind of PDF_{W10} documents are digitally signed, we grouped the signed PDF files accordingly to their origins into the following four categories: *i*) Corporations

Table 4: Signed PDF files in PDF_{W10}

Status	PDF_{W10}	(%)
Valid signature (20)	131	1.25%
Invalid timestamp token (30)	18	0.17%
Timestamp from signer (40)	95	0.91%
Invalid OSCP (50)	7	0.07%
Cannot verify certificate (60)	41	0.39%
Unsupported extensions (64)	1	0.01%
Some unsigned content (70)	1	0.01%
Total	294	2.81%

Table 5: High level grouping of signed PDF files

Type	Count	(%)
Corporations entities	16	5.44 %
Individuals	28	9.52%
Invoices	172	58.50%
Public entities	78	26.53%
Total	**294**	**100%**

entities; *ii*) Individuals; *iii*) Invoices; *iv*) Public entities. The category *corporation entities* groups the documents signed by a corporation/company, while *Individuals* identifies PDF files signed by individuals. The *invoices* category groups digitally signed invoices. Finally, the *Public entities* category gathers the digitally signed PDF files related to public services. Table 5 counts the number of PDF files per category, and presents the percentage considering the set of 294 signed PDF files. The dominant category is *invoices*, with 172 PDF documents (58.50%). The invoices are mainly from three service utilities: electricity, water and internet services. There is also one invoice of an airline ticket. Interestingly, a PDF containing a boarding pass from an airline company was also found to be digitally signed (although with no signer ID), while a boarding pass from another airline company was not signed. In the analyzed files, the PDF files labeled as corporations entities are bids and proposals regarding acquisition of equipment. The 28 PDF files categorized as *individuals* are from four different individuals that digitally signed them with their own Portuguese ID card. Surprisingly, every public-key certificate embedded within PDF files signed with the Portuguese ID card include, in addition of the citizen ID and the full name, the date of birth of the card owner in the *subject directory attributes* field. Finally, the 78 PDF files from public entities are documents signed by representative of public institutions, containing information regarding administrative data of institutions. For examples, several of the signed documents are minutes of official meetings.

6.2 Permission-restricted Files

Results regarding owner permissions for the analyzed PDF files are shown in Table 6. Of the 10 483 analyzed PDF files, 497 have the *assembly* and/or *modify* permissions set to off. Almost all of these files, 347 files (88.30%) have both permissions disabled, while 18 files (4.58%) have the *assemble* permission off, and 28 files (7.12%) have the *modify* permission inhibited.

Table 6: PDF files with owner permissions sets

	Count	(%)
AssembleOFF_ModifyOFF	347	88.30%
AssembleOFF_ModifyON	18	4.58%
AssembleON_ModifyOFF	28	7.12%
Total files	393	100.00%

Table 7: Categories of PDF files with both Assemble and Modify permissions OFF

	Count	(%)
Official documents	198	57.06%
Technical documents	58	16.71%
Invoices	45	12.97%
Bank operations	25	7.20%
Miscellaneous	21	6.05%
Total	347	100.00%

For a better perception of the types of PDF files that are simultaneously set with the *assembly* and *modify* permissions off, we performed a review on the files, grouping them into four distinct categories. The categories and the count of PDF files per category are listed in Table 7. Of the 347 documents with both the assembly and modify permissions off, 198 (57.06%) are public official documents, mostly from the publicly available official journal of the Portuguese government. Around 16.7% of the PDF files are technical documents (catalogs of products, datasheets, user's manuals, etc.), 45 (12.97%) are invoices, and 25 (7.20%) are related to bank operations (payments, money transfers, etc.). Finally, 21 (6.05%) documents do not belong to any category in particular, since they are too diverse to fit in a category, and thus are classified as miscellaneous.

6.3 High Level Analysis

To assess the potential value of digitally signed and/or owner permissions restricted PDF files, we performed an analysis of the information, with forensic value, that could be gathered by solely examining these PDF files in the PDF_{W10} set of files. Through the digitally signed documents, we were able to determine the identity of the owner of the files and to trace back the address of his previous and current (at the time of acquisition of PDF_{W10}) residence. This information was obtained from the digitally signed invoices of the service utilities. We also found out that the owner of the files made at least one air travel on a given airline. This last information was gathered from the boarding pass PDF file that was also digitally signed. The PDF files with restricted owner permissions were less valuable. Nonetheless, the invoices from the internet service utility were detected, allowing to track the address change and to accurately time it, while files holding data from online bank operations revealed details about the monthly rent paid for the location of the first residence of the owner of the files.

7 CONCLUSION

In this work, we describe the `digiSigned|Protected`PDF module for the Autopsy open source software. The module detects, in digital

forensic data sources, PDF files that are digitally signed or/and set with assemble and modify owner restrictions. Based on the results provided by the analysis of a Windows 10 dataset, we studied the relevance of digital signatures and owner permissions of PDF files in the context of digital forensic examinations. In the analyzed case, the detection of some digitally signed PDF files allowed to rapidly gather some valuable information regarding the ownership of the files and some of the personal details, such as home address and job, as well as, to provide a timeline of events, since signed documents have a timestamp, whose accuracy can, in certain cases, be validated. Relatively to PDF files with owner restricted permissions, they also provided some information, but were less informative, from a digital forensic examination perspective, than digitally signed PDF. As the number of operations and transactions done online, namely through mobile platform, continues to increase, it is expectable that situations requiring digitally signed documents will continue to augment. Therefore, the importance of digitally signed documents will certainly increase, opening new opportunities for digital forensic examinations.

As future work, we aim to produce a version of the module that can be run outside of Autopsy. The goal is to allow for the use of the functionalities of `digiSigned|ProtectedPDF` in the triage stage [33]. We also aim to add the possibility for the module to output results in the Digital Forensic XML (DFXML) format [11]. Finally, we aim to study the prevalence of digital signatures and owner restricted permissions in content found on mobile platforms.

ACKNOWLEDGMENTS

This work was partially supported by FCT and Instituto de Telecomunicações under project UID/EEA/50008/2013 and by DEI-ESTG-Polytechnic Institute of Leiria.

REFERENCES

[1] Gary Anthes. 2015. Estonia: a model for e-government. *Commun. ACM* 58, 6 (2015), 18–20.
[2] AusCert. [n. d.]. Deprecation of SHA1 Certificates. Website (access on 2018-05-10). ([n. d.]). https://cs.auscert.org.au/news/deprecation-of-sha1-certificates.
[3] Autopsy. 2018. Autopsy - 3^{rd} party modules. Website (access on 2018-05-10). (2018). https://wiki.sleuthkit.org/index.php?title=Autopsy_3rd_Party_Modules.
[4] Daniel J Bernstein, Yun-An Chang, Chen-Mou Cheng, Li-Ping Chou, Nadia Heninger, Tanja Lange, and Nicko Van Someren. 2013. Factoring RSA keys from certified smart cards: Coppersmith in the wild. In *International Conference on the Theory and Application of Cryptology and Information Security*. Springer.
[5] Stephen E Blythe. 2005. Digital signature law of the United Nations, European Union, United Kingdom and United States: Promotion of growth in E-commerce with enhanced security. *Richmond Journal of Law & Technology* 11, 2 (2005), 6.
[6] Harlan Carvey. 2011. *Windows registry forensics: Advanced digital forensic analysis of the windows registry*. Elsevier.
[7] Aniello Castiglione, Alfredo De Santis, and Claudio Soriente. 2010. Security and privacy issues in the Portable Document Format. *Journal of Systems and Software* 83, 10 (2010), 1813–1822.
[8] Isabel Cerqueira, Vítor J Sá, and Sérgio Tenreiro de Magalhães. 2012. Study of the Perception on the Portuguese Citizen Card and Electronic Signature. In *Global Security, Safety and Sustainability & e-Democracy*. Springer, 164–170.
[9] David Cooper. 2008. *Internet X. 509 public key infrastructure certificate and certificate revocation list (CRL) profile.* RFC 5280. RFC Editor. 1–151 pages. https://www.rfc-editor.org/rfc/rfc5280.txt
[10] CMKC Cuijpers and Jessica Schroers. 2014. eIDAS as guideline for the development of a pan European eID framework in FutureID. (2014).
[11] Simson Garfinkel. 2012. Digital forensics XML and the DFXML toolset. *Digital Investigation* 8, 3-4 (2012), 161–174.
[12] Simson L Garfinkel. 2006. Forensic feature extraction and cross-drive analysis. *Digital Investigation* 3 (2006), 71–81.
[13] Simson L Garfinkel. 2010. Digital forensics research: The next 10 years. *Digital Investigation* 7 (2010), S64–S73.
[14] Gemalto. 2017. 10 years of Portuguese ID card. Website (access on 2018-04-30). (2017). https://www.gemalto.com/govt/customer-cases/portugal-id.
[15] Christophe Grenier. 2011. Photorec. *URL http://www.cgsecurity.org/wiki/PhotoRec* (2011).
[16] Phil Harvey. 2013. ExifTool: Read, write and edit meta information. *Software package available at http://www.sno.phy.queensu.ca/ phil/exiftool* (2013).
[17] ISO ISO. 2008. 32000-1: 2008, Document Management–Portable Document Format–Part 1: PDF 1.7. *International Organization for Standardization, Geneva, Switzerland* (2008).
[18] ISO ISO. 2017. 32000-2: 2017, Document Management–Portable Document Format–Part 2: PDF 2.0. *International Organization for Standardization, Geneva, Switzerland* (2017).
[19] Nicholas Jansma and Brandon Arrendondo. 2004. *Performance Comparison of Elliptic Curve and RSA Digital Signatures.* Technical Report. University of Michigan, Ann Arbor, MI, USA.
[20] Duff Johnson, Larry Masinter, Dejan Markovic, Matthew Hardy, and Martin Bailey. 2017. *RFC8118: The application/pdf Media Type.* RFC 8118. RFC Editor. 1–12 pages. https://www.rfc-editor.org/rfc/rfc8118.txt
[21] Don Johnson, Alfred Menezes, and Scott Vanstone. 2001. The elliptic curve digital signature algorithm (ECDSA). *International journal of information security* 1, 1 (2001), 36–63.
[22] DYW Liu, GZ Xue, Y Xie, XP Luo, and MH Au. 2016. Performance of Digital Signature Schemes on Mobile Devices. In *Mobile Security and Privacy*. Elsevier.
[23] Matus Nemec, Marek Sys, Petr Svenda, Dusan Klinec, and Vashek Matyas. 2017. The Return of Coppersmith's Attack: Practical Factorization of Widely Used RSA Moduli. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1631–1648.
[24] Weiwei Pan and Guolong Chen. 2016. A method of off-line signature verification for digital forensics. In *Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), 2016 12^{th} International Conference on*. IEEE, 488–493.
[25] PatricioDomingues. 2018. PatricioDomingues/digiSignedOrProtectedPDF: Version 1.0. (April 2018). https://doi.org/10.5281/zenodo.1218481
[26] Darren Quick and Kim-Kwang Raymond Choo. 2014. Impacts of increasing volume of digital forensic data: A survey and future research challenges. *Digital Investigation* 11, 4 (2014), 273–294.
[27] Ronald L Rivest, Adi Shamir, and Leonard Adleman. 1978. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* 21, 2 (1978).
[28] Ben Rogers. 2009. Digital Signatures in a PDF. Website (access on 2018-04-30). (2009). https://www.adobe.com/devnet-docs/acrobatetk/tools/DigSig/Acrobat_DigitalSignatures_in_PDF.pdf.
[29] Vassil Roussev, Candice Quates, and Robert Martell. 2013. Real-time digital forensics and triage. *Digital Investigation* 10, 2 (2013), 158–167.
[30] Abhishek Roy and Sunil Karforma. 2012. A Survey on digital signatures and its applications. *Journal of Computer and Information Technology* 3, 1 (2012), 45–69.
[31] S Santesson, M Myers, R Ankney, A Malpani, S Galperin, and C Adams. 2013. *X. 509 internet public key infrastructure online certificate status protocol - OCSP (rfc 6960).* RFC 6960. RFC Editor. 1–41 pages.
[32] Bruce Schneier. 2007. *Applied cryptography: protocols, algorithms, and source code in C*. John Wiley & Sons.
[33] Adrian Shaw and Alan Browne. 2013. A practical and robust approach to coping with large volumes of data submitted for digital forensic examination. *Digital Investigation* 10, 2 (2013), 116–128.
[34] Marc Stevens, Elie Bursztein, Pierre Karpman, Ange Albertini, and Yarik Markov. 2017. The first collision for full SHA-1. In *Annual International Cryptology Conference*. Springer, 570–596.
[35] Marc Stevens, Arjen K Lenstra, and Benne De Weger. 2012. Chosen-prefix collisions for MD5 and applications. *International Journal of Applied Cryptography* 2, 4 (2012), 322–359.
[36] Petr Švenda, Matúš Nemec, Peter Sekan, Rudolf Kvašňovskỳ, David Formánek, David Komárek, and Vashek Matyáš. 2016. The Million-Key Question–Investigating the Origins of RSA Public Keys. In *Proceedings of the 25th USENIX Security Symposium*. USENIX Organization, 893–910.
[37] Hamed Taherdoost, Shamsul Sahibuddin, and Neda Jalaliyoon. 2013. E-services usage evaluation; applications' level of co-creation and digitalization. (2013).
[38] Peter Yee. 2013. Updates to the internet X. 509 public key infrastructure certificate and Certificate Revocation List (CRL) profile. (2013).