Available online at www.sciencedirect.com

SciVerse ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

A taxonomy and survey of attacks on digital signatures

*Jorge L. Hernandez-Ardieta**, *Ana I. Gonzalez-Tablas*, *Jose M. de Fuentes*, *Benjamin Ramos*

Computer Science and Engineering Department, University Carlos III of Madrid, Avda. de la Universidad 30, 28911 Leganes, Madrid, Spain

ARTICLE INFO

Article history:
Received 29 December 2011
Received in revised form
9 October 2012
Accepted 19 November 2012

Keywords:
Security
Taxonomy
Digital signature
Attacks
Non-repudiation
Public key infrastructure

ABSTRACT

Non-repudiation is a desired property of current electronic transactions, by which a further repudiation of the commitments made by any involved party is prevented. Digital signatures are recognized by current standards and legislation as non-repudiation evidence that can be used to protect the parties involved in a transaction against the other's false denial about the occurrence of a certain event. However, the reliability of a digital signature should determine its capability to be used as valid evidence. The inevitability of vulnerabilities in technology and the non-negligible probability of an occurrence of security threats would make non-repudiation of evidence difficult to achieve. We consider that it is of the utmost importance to develop appropriate tools and methods to assist in designing and implementing secure systems in a way that reliable digital signatures can be produced. In this paper, a comprehensive taxonomy of attacks on digital signatures is presented, covering both the signature generation and verification phases. The taxonomy will enable a rigorous and systematic analysis of the causes that may subvert the signature reliability, allowing the identification of countermeasures of general applicability. In addition, an intensive survey of attacks classified under our taxonomy is given.

1. Acronyms

Next Table 1 compiles the definition of the acronyms used along the paper.

2. Introduction and motivation

Traditional sensitive operations, like banking transactions, purchase processes or contract agreements, need to tie down the involved parties respecting the commitments made, avoiding a further repudiation of the responsibilities taken. Depending on the context, the commitment is made in one way or another, though handwritten signatures have been possibly the most common mechanism ever used. With the shift to digital communications, the same properties that are found in real world transactions are expected from electronic ones as well. Non-repudiation is thus a desired property of current electronic transactions, like those carried out in Internet banking, e-commerce or, in general, any electronic data interchange scenario.

ISO/IEC 13888-1 (2009) defines a general model for non-repudiation mechanisms providing evidence based on cryptographic check values generated using symmetric or asymmetric cryptographic techniques. Evidence is generated, collected, maintained, made available and verified by non-repudiation services in order to resolve disputes about the occurrence of a certain event, protecting the parties involved in a transaction against the other's false denial about such an event.

* *Corresponding author.*
E-mail addresses: jlhernan@inf.uc3m.es (J.L. Hernandez-Ardieta), aigonzal@inf.uc3m.es (A.I. Gonzalez-Tablas), jfuentes@inf.uc3m.es (J.M. de Fuentes), benja1@inf.uc3m.es (B. Ramos).
0167-4048/$ – see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cose.2012.11.009

Table 1 – List of acronyms used for the entities within the signature creation and verification environments.

Acronym	Name of the entity	Acronym	Name of the entity
CSP	Cryptographic Service Provider	SCE	Signature Creation Environment
DTBS	Data To Be Signed	SCS	Signature Creation System
DTBSR	Data To Be Signed Representation	SD	Signer's Document
DTBV	Data To Be Verified	SSCDev	Secure Signature Creation Device
SAD	Signer's Authentication Data	SVA	Signature Verification Application
SCA	Signature Creation Application	SVE	Signature Verification Environment
SCD	Signature-Creation Data	SVS	Signature Verification System
SCDev	Signature Creation Device	SWKey	Software Keystore

Non-repudiation services are built based on non-repudiation mechanisms that provide protocols for the exchange of non-repudiation tokens specific to the service. A non-repudiation token includes the evidence itself and, optionally, additional data. Within the ISO model (ISO/IEC 13888-3, 2009), a digital signature is a non-repudiation token that is exchanged during a protocol and which can be used subsequently, by disputing parties or by an adjudicator, to arbitrate in disputes.

When the life cycle of evidence is properly assured, it becomes valid in accordance with the non-repudiation policy in force. In this case, evidence is considered as proof, meaning that it serves to prove the existence of something. Valid evidence or proof avoids a later repudiation of the commitments made in the transaction by the involved parties. On the contrary, evidence which generation, transfer, maintenance or verification is not reliable cannot contribute to the establishment of proof about an event or action, rendering it useless.

A digital signature (Rivest et al., 1978) based on public key cryptography (Diffie and Hellman, 1976) fulfills the properties that non-repudiation evidence should fulfill (Zhou and Gollmann, 1997). In particular, the origin and integrity of evidence must be verifiable by a third party, and the validity of the evidence must be undeniable.

The electronic signature, as a conceptual term, has become a key element in the information society. Several national and international legislations recognize the legal effectiveness of electronic signatures and their admissibility as evidence in legal proceedings. Under current legislation, the signatory is legally bound respecting the commitments made in the signed document once their knowledge and approval of the content of the document are consciously represented by their electronic signature. The electronic signature acts as an instrument of evidence regarding the authenticity of the electronic document in the same way as the handwritten signature does regarding the paper-based document. In addition, current legislation specifically grants electronic signatures an important role for promoting e-commerce under secure conditions (European Directive 1999/9, 1999; Federal Trade Commission, 2000; Department of Justice and Government of Canada, 2008; United Nations, 2001).

The aforementioned legislation on electronic signature is technology-neutral, as the technical or procedural requirements for generating and verifying electronic signatures are not specified (Broderick et al., 2001). They establish generic requirements that must be fulfilled by the implementing technology, either present or future (European Directive 1999/9, 1999; Department of Justice and Government of Canada, 2008; United Nations, 2001). An electronic signature which conforms to these requirements (functional equivalence) will have legal effect, no matter its nature or technical background. This model grants, to market forces, the power to decide what constitutes an electronic signature.

However, based on the current state-of-technology, only cryptographic digital signatures satisfy the requirements for certain types of electronic signatures, such as the advanced or qualified signatures under the European Directive 1999/9 (1999). In this sense, some legislations explicitly state that a digital signature supported by a Public Key Infrastructure (PKI) is one of the potential underlying technologies. For instance, the European Directive refers to digital certificates and signature creation devices, and the UNCITRAL Model Law even establishes PKI and digital signatures as an example of implementing technologies for generating compliant signatures.

It should be noted that, in spite of the above, the reliability of a digital signature still determines its capability to be used as valid evidence. In general, and in case of disagreement respecting the authorship of a certain signed document, the security of the means used to produce the digital signature and evaluated at Court by an expert's report (when required) will determine whether the signature is accepted as valid evidence or not. Depending on the type of signature and the legislation in place, the onus of proof may be reversed, moving the burden of proof to the alleged signatory instead of the verifier (McCullagh and Caelli, 2000). In any case, it is possible to repudiate the authorship of certain document by proving, on the balance of probabilities or beyond reasonable doubt, that the corresponding digital signature is not as reliable as it should be for its application as evidence.

In this sense, the reliability depends on the trustworthiness of the whole life cycle of the signature, including the generation, transfer, verification and storage phases. Any vulnerability in it would undermine the reliability of the digital signature, making its applicability as non-repudiation evidence difficult to achieve.

Unfortunately, technology is subject to vulnerabilities, always with the existence of risk – higher or lower, but never void – of an occurrence of security threats. This situation produces undesirable consequences, that we summarize here:

- Non-repudiation evidence based on digital signatures becomes useless as there will always be a chance to prove the existence of a vulnerability in the evidence life-cycle.

- Non-repudiation evidence based on digital signatures is unfairly enforced if any of the stages within its life-cycle is compromised but it cannot be proved by the affected party.

Therefore, two unfair rulings could be made by the Court:

- The alleged signatory must take on the commitments made in a document signed by a malicious external entity without their consent. Under some legislations (e.g. European Directive 1999/9, 1999) and certain types of signatures (e.g. qualified signatures), it should also be mentioned that, although the alleged signatory proved that the private key was compromised, it is very likely that he will have to take on the responsibilities as he would be charged with negligence for not keeping the private key in a proper manner (Rivero and Pons, 2006).
- The alleged signatory is capable of proving the existence of a vulnerability in the process, avoiding the signature exercising its authentication function, that is, identifying the apparent signatory as the subscriber of the document. As a result, digital signatures would be repudiable, losing their property of non-repudiation evidence, and thus, their usefulness as intended by the Law and the current standards.

The reliability of a signature as evidence in a legal proceeding will highly depend on the capability to find and prove the existence of a vulnerability in the process. As a result, it would be advisable determining, in a rigorous manner, to what extent current technology provides an acceptable level of trustworthiness to produce reliable non-repudiation evidence. As it can be deduced from current non-repudiation standards and legislation, it is generally assumed that existing technology for digital signatures, when certain requirements are met, provides the sufficient guarantees to use digital signatures as non-repudiation evidence. However, to date, no rigorous mechanism addressing the security problem of digital signatures technology in a holistic manner has been devised.

We claim that it is of the utmost importance to develop appropriate tools and methods to assist in designing and implementing secure systems in a way that reliable digital signatures can be produced.

A taxonomy is a system or scheme to systematically classify an area of knowledge. By applying a systematic and rigorous analysis, such knowledge is classified in accordance with a limited and well defined set of categories. From a general viewpoint, the benefits of a taxonomy are multiple. A taxonomy permits splitting a complex phenomenon into more understandable pieces of information. As a result, a taxonomy makes further studies possible, providing a common agreed base, and identifying the parts of the phenomenon that are less known. Using the classification of the taxonomy, one is more capable of explaining observed phenomena. In that sense, a taxonomy of attacks on digital signatures would enable a rigorous and systematic analysis of the causes that might subvert the signature reliability, allowing the identification of countermeasures of general applicability. To the best of our knowledge, such taxonomy has not been proposed so far.

In this paper, we present a comprehensive taxonomy and survey of attacks on digital signatures. We focus on practical attacks (malicious faults) on the signature generation and verification environments, as the generation and verification operations are the most sensitive and profitable ones for an attacker. Non-malicious faults are also quite common and may take place more frequently than malicious ones in real settings. However, we decided to address malicious faults in the first instance due to the fact that these can entail more serious consequences. Notwithstanding, the method of classification provided along with the taxonomy, and the design of the taxonomy itself, permit to add new categories in a manner the taxonomy can be extended and enriched as the state of the art evolves. In our opinion, a taxonomy that structures this area of knowledge and that makes possible further extensions by the research community is a significant contribution to the field.

3. Overview of the proposal and paper organization

The taxonomy proposed in this paper is specifically designed to classify attacks on digital signature (Rivest et al., 1978) applications, and, in particular, practical attacks focused on the signature creation and verification stages. We consider that digital signatures are created using asymmetric cryptography (Diffie and Hellman, 1976), and that a Public Key Infrastructure (PKI) (RFC 5280, 2008) is used to bind the public key and the identity of the signer by means of a digital certificate issued by a certification authority. We also assume that the underlying services for the certificate validation are provided by a PKI.

A phenomenon (an attack in our case) is always observed within and in relation to an environment. The behavior and even the occurrence of a phenomenon are subject to the variations that may be produced in the conditions of such an environment. As a result, delimiting the nature and acceptable conditions of the environment is paramount to devise a consistent taxonomy. In this sense, Section 6 defines the model of the signature creation and signature verification environments that may suffer attacks classifiable under our taxonomy. In addition, establishing any particularity or restriction on the potential attack that may be carried out on the aforementioned environments would also facilitate to design a more precise taxonomy. For that purpose, the attacker profile is also given in Section 6.

Section 5 explains the methodology used to derive the taxonomy, while Section 7 presents the taxonomy itself. The taxonomy is based on dimensions, where a dimension is a property that permits the classification of an event to take a more holistic view of such an event (Hansman and Hunt, 2005). Each event is described according to several properties, which are used to classify the event from a different perspective, all of them complementary as a whole. In the literature, the authors have chosen different dimensions according to the goal of the taxonomy and the approach followed for the classification. In our case, three dimensions are defined:

- **Attacker's goal**, which covers the goal of the attack.
- **Method of attack**, which corresponds to the method of attack executed by the attacker to achieve the goal classified in the previous dimension.

- **Target of the attack**, which identifies the target(s) of the attack. The multiple instances of this dimension permit to know every element, software, hardware or human, that is affected during the attack.

As a taxonomy intends to permit the classification of observed phenomena, a method that guides a user when a new element has to be classified should also be provided. The method for the classification of attacks according to our taxonomy is detailed in Section 8.

The results of an intensive survey and classification of 117 attacks found in the literature are presented in Section 9. This survey intends not only to demonstrate the completeness of the taxonomy but also to review the most relevant attacks on digital signatures.

In addition, any taxonomy should always be evaluated according to a general and well-known set of requirements for taxonomies. In this sense, the evaluation of our taxonomy is given in Section 10.

Next Section 4 reviews previous work, covering taxonomies of attacks on digital signatures, while our conclusions are given at the end of the paper in Section 11.

4. Related work

Many taxonomies aimed at classifying attacks and vulnerabilities in computer systems have been proposed so far. Among them, several authors have used the dimension approach to provide a holistic view of vulnerabilities or attacks. Lindqvist and Johsson (1997) introduced the concept of dimension in their early paper to classify computer security intrusions. Landwehr et al. (1994) also used characteristics to classify computer program security flaws. Bishop (1995) used several axes to classify vulnerabilities in a manner that was more useful for intrusion detection mechanisms than traditional mechanisms based on pattern recognition. Howard and Longstaff (1998) designed a process-driven taxonomy where multiple factors (attackers, tool, vulnerability, action, target, unauthorized result, objectives) were used for classifying security incidents.

Hansman and Hunt (2005) provided a specific dimension-based taxonomy able to cope with blended attacks, and which was intended to be the first taxonomy that gave a holistic approach to classify attacks, taking into account all parts of the attack. Their taxonomy uses four dimensions. The first one covers the attack vector and behavior. The second dimension focuses on the attack targets. The third dimension deals with the specific vulnerability that allows the attack to be carried out. The last one classifies attacks having payloads or effects beyond themselves.

In spite of the large number of taxonomies proposed so far, to date only a few taxonomies have concentrated on attacks on digital signatures. The expected specificity of a taxonomy makes proposals aimed at a different area of knowledge of little profit to deal with the non-repudiation problem. Next, we briefly review the most relevant taxonomies specifically focused on attacks and vulnerabilities that may affect the reliability of digital signatures, but which fail to offer a complete view of the problem.

A taxonomy of attacks on XML signatures is given in Hill (2004). The goal of the taxonomy is to list specific attacks on signatures that follow the XML signature standard, providing information such as the attack surface and impact, examples and possible countermeasures. Though the taxonomy is very useful to prevent these types of attacks, it fails to provide a holistic view of attacks on digital signatures.

In Rae and Wildman (2003), proposed a matrix-based taxonomy for attacks on secure devices. Classification trees based on the access, goal and method dimensions are provided. An overview of some attacks related to each category is also given. The authors consider that the attacker is able to communicate with the device through the available interfaces and established protocols, as well as physically handle and manipulate the device.

Other researchers have studied attacks and vulnerabilities in smart cards that may contain cryptographic material, such as private keys for signing purposes (Girard and Giraud, 2003). A special attack on hardware cryptographic devices, called side-channel attack, has been extensively studied in the literature (ECRYPT, 2008). However due to the heterogeneity and dependencies that such attacks have on the internal algorithms, the attacked device and the applied technique, no complete taxonomy has been presented so far.

Kain (2003) outlined a taxonomy for dynamic content attacks on signed documents. The taxonomy contains three dimensions, named hidden parameters, fraudulent content and nature of change, that are split into several subcategories. As mentioned by the author, the taxonomy is not complete. It is exclusively focused on attacks that can vary the semantics of the signed document, and thus it does not cover many other types of attacks.

CEN derives the set of requirements for signature creation applications (CEN Workshop Agreement 14169, 2004) from the threats that can affect each element of the Signature Creation Environment (SCE) model. We consider that it is an exhaustive work that covers most of the attacks on the generation stage. However, the threats are not categorized except by the component being affected. In addition, CEN has not performed the same study on the Signature Verification Environment (SVE) model, leading to an incomplete categorization of threats on digital signatures.

5. Methodology used to create the taxonomy

In this Section we explain the methodology used to create and compose the taxonomy. The methodology was designed taking into account Kitchenham's (2004) guidelines regarding the procedures for performing systematic reviews.

In the first instance, a thorough review of the state of the art of relevant taxonomies was performed. The taxonomies covered included not only those specifically focused on digital signatures (see Section 4), but also relevant taxonomies of other areas of knowledge. This study period permitted us to make an informed decision regarding the overall structure and design of the taxonomy, taking into account the advantages and disadvantages of the different models proposed in the literature (trees, lists, dimensions, etc.) and the purposes of our work. In our case, and as discussed in Section 3, we

chose a dimension-based approach as a way to provide a wider coverage of the attack properties.

Once the design criteria was selected, the dimensions and categories of the taxonomy had to be created. An interesting property of a taxonomy is that it is used to classify a particular area of knowledge, but it is also created after (or during) the retrieval and analysis of that knowledge, or a representative part of it, and once its properties have been extrapolated to a limited and well defined set of categories. In other words, the categories of the taxonomy used to classify the phenomena are created from the information retrieved from the phenomena themselves.

In our case, we surveyed and analyzed a first group of attacks that were used to extract properties shared amongst attacks on digital signatures. After this analysis, we stipulated the three dimensions that, in our opinion, permit to gain an understanding of the fundamental principles of an attack: the attacker's goal, the method of attack and the target(s) of the attack. Afterward, we expanded the search until a representative number of phenomena (attacks) from which deriving the categories of taxonomy was reached.

All attacks surveyed and analyzed were chosen according to their importance and relevance by using prestigious scientific electronic databases and specific search criteria and keywords ([attack AND digital signature] OR [attack AND electronic signature], and published at any date). Primary sources (i.e. attacks shown in the first positions of search results ordered by cites) were used to access secondary sources of relevance (i.e. papers that contain references to primary sources, or papers that were referenced from a primary source). The depth of search was extended following up the cites and references found in secondary sources.

In all cases, we only considered works published in relevant conferences, journals or security-relevant sources. Inclusion criteria considered not only relevant papers that dealt with attacks on signatures but also others that were of applicability (e.g. attack techniques that could be used to compromise signature-related data but where the authors focused on a different topic). Exclusion criteria used to detect false positives included a number of steps by which the title, abstract and finally the content of the paper were reviewed to detect those papers shown in the search results but where their applicability was out of the scope of our research. The data extraction strategy used in the methodology corresponds to steps one to four of the method of classification explained in Section 8, along with the extraction of the data needed to compose the complete reference to each attack (author, title, journal/conference, year of publication, etc.).

If the taxonomy design and method of classification are appropriate, the taxonomy evolves in a manner that it fulfills the following characteristic: the more phenomena are classified, the less new categories are created. Therefore, the taxonomy remains stable after a critical mass of phenomena has been classified, and thus its set of categories can be considered a reliable representation of that particular area of knowledge. Fig. 1 depicts the expected evolution a taxonomy should follow in terms of number of new categories created according to the number of phenomena classified.

It is very difficult, if not impossible, to analyze and classify all phenomena of any area of knowledge, particularly in the security field. We cannot predict attacks that have yet to be invented either. If we want to classify a new phenomenon that is, for instance, a new type of attack that uses a novel mechanism not previously documented, then it is likely that the taxonomy design did not cover it in the first instance. Consequently, the classification of a new phenomenon, with different characteristics than those found in already classified phenomena, may need changes in the categories or subcategories of the taxonomy. In that sense, a taxonomy and the method of classification must be designed to permit further refinements. The taxonomy has to be considered as something live, that may change along the time. However, the better the design of the taxonomy and the method of classification, the less structural changes in the taxonomy will need to be applied.

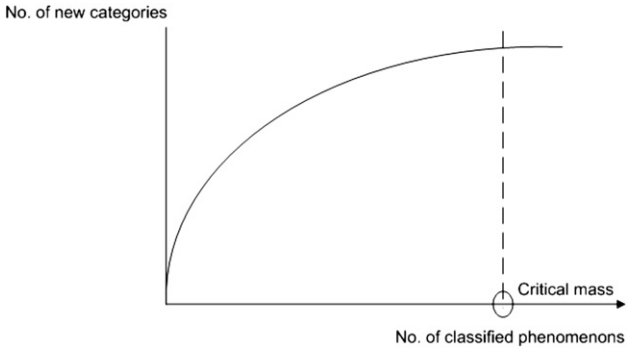

Fig. 1 – Expected evolution of a taxonomy.

In our particular case, we have ascertained that the evolution of our taxonomy has behaved as expected. During the last classifications of the total 117 attacks classified, only two new subcategories had to be created. This behavior supports the statement that the taxonomy has reached a stable, reliable and representative set of categories in respect to the current state of the art, and thus that the number of attacks classified represent a significant number that can be considered as a critical mass for the taxonomy consistency. This statement is also supported by the methodology followed to identify relevant attacks, and by which the probability of a biased approach driven by the researcher expectations is reduced.

Section 9 presents the results obtained after the analysis and classification of 117 attacks. It should be noted that there are some categories of dimension two (method) and three (target) for which no attack was found. These categories were not created by the direct analysis of the attacks surveyed, but were included by the authors based on the next reasons:

- For completeness purposes. The experience and results obtained after the survey show that the elements of the signature environment with a direct relationship with the signature creation or verification processes are more likely to be attacked. In this sense, and although no attack was found for certain elements, the list of target categories has been expanded to cover some elements that may be subject to attacks (e.g. *D3-CAT3.2.3: Memory*, as this element may manage the private key during a signature creation process,

and thus an attacker would be able to retrieve this sensible data using a cold boot attack, for example).

- Due to analogy between attacks. We observed that some attacks that used certain techniques may put into practice alternate methods of attack. Consequently, it would be interesting to capture all these methods in order to broaden the coverage of the taxonomy. For example, Loughry and Umphress (2002) propose a new information leakage technique called optical emanation. The authors present experimental results where plain text can be leaked due to a flawed design of the encryption module, and formulate a scenario where the encryption keys could be leaked as well. Using our taxonomy this attack method has been classified as *D2-CAT4.1.2.1: Observation*, and by which the attacker may observe the plain text introduced by the user (in our case, the signature authentication data). However, the same technique, i.e. optical observation, could be used to compromise the private key when used in a computer with particular hardware. Therefore, we created a category named *D2-CAT5.2.5: Optical observation* under first level category *D2-CAT5: Compromise of the signature creation data (SCD)* to reflect it.

6. Model of the signature environments

In this Section we present the models that describe the nature and composition of the signature creation and signature verification environments that may suffer an attack classifiable under our taxonomy. In addition, the profile of the attacker capable of compromising these environments is defined.

6.1. *Signature creation environment*

The model of the environment used by the signer to generate a digital signature corresponds to the one provided by CEN (CEN Workshop Agreement 14170, 2004), with some refinements further explained. Therefore, we will consider digital signatures generated by end users, who own and control the signing key and interact with the signing capabilities offered by the environment. The physical environment where the signing process takes place can be either under the signer's control and possession (e.g. personal or corporate computer, mobile phone, personal digital assistant, etc.) or operated by a service provider not necessarily related to or under the control of the signer (e.g. a public place like a point of sale or a bank), but in any case accessible by him. It should be mentioned that there is no direct interface or communication channel between the signer and the signing key and the information to be signed. The signer must rely on the technological elements of the environment to produce a signature. In any case, it is the purpose of the environment to provide the signer with the means to securely and consciously create digital signatures on their own behalf and on the intended information.

As mentioned by CEN, the model does not intend to specify the nature or distribution of the components. These aspects can only become more concrete in the context of a particular set of technologies that apply to the signature creation system.

In the CEN model, the Signature Creation Environment (SCE) is the physical, geographical and computational environment of the Signature Creation System (SCS), including the signer and the existent policies. The SCS consists of the software and hardware needed to generate digital signatures.

The Signature Creation Application (SCA) is the application within the SCS that creates digital signatures, excluding the Signature Creation Device (SCDev). The signer can interact with the SCA directly or through other applications (e.g. user applications). The SCDev is defined by the European Directive on electronic signatures (European Directive 1999/9, 1999) as *configured software or hardware used to implement the signature-creation data* (SCD), being the SCD *unique data, such as codes or private cryptographic keys, which are used by the signatory to create an electronic signature*. Therefore, the SCDev can be either software or hardware.

An SCDev, either hardware or software, that meets the requirements laid down in Annex III of the European Directive 1999/9 (1999) is called a Secure Signature Creation Device (SSCDev). CEN CWA 14169 (2004) defines the security requirements for SSCDev in accordance with the Annex III, and following the technology-neutral principle claimed by the European Directive.

However, and contrary to this, CEN CWA 14170 (2004) actually restricts the attribution of SCDev and SSCDev to hardware devices only. We decided to follow the general approach given by the European Directive and CEN CWA 14169, also considering software devices as devices that implement the SCD. The reason also stems from the possibility that a digital signature not generated with a hardware cryptographic device can be considered as evidence in legal proceedings as well. Furthermore, ISO model on non-repudiation does not specify the device to use for the signature computation.

In addition to the software (S)SCDev, we also consider three additional components in the model, not found explicitly in CEN CWA 14170 model: the device driver, the cryptographic service provider (CSP) and the Software Keystores (SWKey). These elements are commonly found in an SCS, independently of its nature, and will permit us to discover relevant attack categories useful for the taxonomy. The CSP is a software layer that operates on top of the operating system and that allows the SCA to transparently access and use the signature-creation data, SCD. SWKey are protected data structures that store the SCD of the user(s), but do not implement signing capabilities. These keystores are managed by specific software, such as SCA or Web browsers. Access to the SCD stored in SSCDev, SCDev and SWKey is protected by means of the Signer's Authentication Data (SAD), which are the data (e.g. PIN, password or biometric data) used to authenticate the signer and required to allow the use of the SCD.

The Data To Be Signed (DTBS) is defined in CEN Workshop Agreement 14170 (2004) as the complete electronic data to be signed. It covers the Signer's Document (SD) and, optionally, the signature attributes, which enrich the semantic of the document. The SD can be a local document, Web content, a document imported from another environment or any other type of information. Signature attributes are signed together with the SD and may include the data content type (it expresses the encoding format of the SD), the signature policy

reference or the commitment type made in the act of signing. Data To Be Signed Representation (DTBSR) is also defined in CEN Workshop Agreement 14170 (2004) as the data sent by the SCA to the (S)SCDev for signing. DTBSR generally corresponds to the cryptographic hash of the DTBS.

6.2. *Signature verification environment*

CEN models the Signature Verification Environment (SVE) in CWA 14171 (2004). The model intends to outline a general guideline on signature verification procedures in order to achieve the recommendations for secure signature verification given in Annex IV of the European Directive on electronic signatures (European Directive 1999/9, 1999). To summarize, the signature verification system is intended to permit the verifier to securely and unambiguously verify digital signatures and associated information.

CEN defines the verifier as *the entity which verifies the electronic signature*, and establishes that *it may be a single entity or multiple entities*. But contrary to CEN CWA 14170, the verifier is not only restricted to end users. Although the European Directive 1999/9 (1999) explicitly refers to the verifier as the person to whom the data used for verifying the signature and the verification result are displayed, CEN considers three different models: a natural person using their workstation and accompanying software to request verification of a received signature, a computer program using an automated procedure, for which the term "display" would cover a broader meaning, and a third-party to which the verification could be sub-contracted.

In this paper we adhere to the vision given by the European Directive, and thus, assume that the verifier is a human user that physically visualizes the signed data and any other information that must be verified during the signature verification process. However, we comply with a multi-party verification process as long as exists a participation of an end user. For example, the end user would be typically involved in the initial and subsequent verification (CEN Workshop Agreement 14171, 2004) in order to visualize and verify the signed data and signer's identity, while the validation information to be captured and archived during such stages may be leveraged to third parties. It should be noted that when referring to Signature Verification System (SVS), we mean a system that may implement the initial verification, the subsequent verification, or both, by means of a Signature Verification Application (SVA). While the initial verification stage could be partially performed by the signer, the subsequent verification stage is always performed by the verifier.

Along the rest of the paper we will use Data To Be Verified (DTBV) term to refer to the information that was signed and has to be verified against the signature. This information corresponds to both the signed document and the signed attributes, that is, the information contained in the DTBS during the signature generation.

6.3. *Attacker profile*

We consider two properties to profile the attackers: the attack potential and the capability to access or approach the target of the attack.

The attack potential is defined as the perceived likelihood of success should an attack be launched, expressed in terms of the attacker's ability (i.e. expertise and resources) and motivation (RFC 4949, 2007). We consider an attacker with enough expertise, resources and motivation to execute any feasible potential attack.

Respecting access capabilities, we consider attackers that can carry out both internal and external attacks.

In an internal attack, the attacker operates inside the security perimeter of the environment, and can be either (i) a malware that has infected an IT element of the system, (ii) a physical person that directly interacts with the environment, handles the hardware (e.g. the (Secure) Signature Creation Device ((S)SCDev)) or even communicates with the end user, or (iii) the end user itself (i.e. a malicious signer). Regarding attacks that handle the hardware (ii), we do not consider attackers that perform invasive tampering attacks on the hardware (e.g. micro probing techniques), and that physically harm it. The advantage of non-invasive attacks is that the equipment used in the attack can usually be disguised as a normal device (e.g. smart card reader), and thus the owner of the compromised hardware might not notice that the secret keys have been stolen. Therefore it is unlikely that the validity of the compromised keys will be revoked before they are abused (Kmmerling and Kuhn, 1999).

On the other hand, in an external attack, the attacker operates outside the security perimeter of the signature environment, possibly through the network.

As can be seen in Section 6.1, the signer is included by CEN in the Signature Creation Environment (SCE) model, as suggested by safety engineering best practices (Rushby, 1994). However, CEN does not include the entity that represents the verifier in the Signature Verification Environment (SVE) model. We also consider that it is not important for the taxonomy, as the verifier does not possess any secret, and thus the attacker cannot obtain any benefit from him.

7. The taxonomy of attacks on digital signatures

This Section describes each dimension of the taxonomy. Each category is assigned an identifier that consists of the number of dimension it belongs to (D) and a category or subcategory number (CAT) according to the hierarchical order established.

7.1. *Dimension one: attacker's goal*

The goal of an attack will consist of achieving one of the next service failures in the Signature Creation System (SCS) or the Signature Verification System (SVS):

(1) The SCS does not protect the signer from an unintended or unauthorized use of the signature creation data.
(2) The SCS does not protect the signer from signing a document different than the intended one.
(3) The SVS does not protect the verifier from performing an ambiguous signature verification.

In particular, there are six categories in this dimension, with no further refinement:

D1-CAT1: Deceive the signer to sign a document different to the intended one

The attacker does not directly use the signature creation data (SCD) but wants to deceive the signer to unconsciously sign a document that is of benefit to the attacker, against the signer's interests, or both. This category corresponds to service failure (2).

D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)

The attacker's objective is to use the Signature Creation Data (SCD) on behalf of the user, but without their consent and knowledge. For that purpose, the attacker will need to either obtain the SCD or have access to the signing function at will. This category corresponds to service failure (1).

D1-CAT3: Replace signed information

The attacker's objective is to directly replace part of or the whole signed information for their own benefit, the signer's detriment or both, and once the signature has been computed. This category corresponds to service failure (2), in the sense that the final signed document does not correspond to the original one.

D1-CAT4: Attribute the signed document to a user different to the actual signer

The attacker's objective is that a document signed by a certain signer is verified as signed by a different entity. Thereby, the attacker could provoke a wrong document's authorship attribution. For instance, the attacker may seek that a document signed by another one is verified as signed by himself (e.g. the document's content is beneficial). The attacker may also seek that a document not signed by a certain user is verified as signed by the user (e.g. the document's content is detrimental to the user). This category corresponds to service failure (3).

D1-CAT5: Make the Data To Be Verified (DTBV) show chosen content

The attacker's objective is that the signed document and/or signed attributes are shown to the verifier either with a content where the appearance may vary (polymorphic) or with a content different to what was actually signed or was intended to be signed. For instance, if the attacker is the signer, he may seek to make some content that was not signed in the beginning is verified as such (e.g. for their own benefit). If the attacker is an external malicious entity, he may seek to attribute to the signer some content not signed or intended to be signed by the signer (e.g. to damage the signer's interests). This category corresponds to service failure (3).

There is an exception when the attacker seeks to show a different content with regard to the identity of the signer. In this case, goal *D1-CAT4: Attribute the signed document to a user different to the actual signer* prevails, and thus the attack should be classified accordingly.

D1-CAT6: Make the signature validity verification conclude with an opposite result

The attacker's objective is to make a signature validity verification raise a result different than the correct one. The validity of the signature depends not only on the signature itself but also on the certificate validity. This goal covers both when a valid signature is verified as invalid, and when an invalid signature is verified as valid. For example, if the attacker is the signer, he may seek that a signature signed by himself is verified as invalid (e.g. to repudiate the commitment made in the signed document), while if the attacker is an external malicious entity, he may seek to make a valid signature generated by a certain user be verified as invalid (e.g. to damage the signer's interests). In the opposite direction, the attacker may seek to make a signature generated over a fraudulently modified document be verified as valid. This category corresponds to service failure (3).

7.2. *Dimension two: method of attack*

The methods that can be used by the attacker to achieve the identified goal are specified in this dimension. Seven categories have been devised at the first level, which are further refined into subsequent subcategories:

D2-CAT1: Environment manipulation

This category includes the methods aimed at manipulating the environment of the Signature Creation System (SCS)/ Signature Verification System (SVS) to have an effect on the signature creation process or the signed information once the signature has been computed.

D2-CAT2: Modification prior to signature computation

This category contains the attack methods that take part before the signature computation, and where the goal is the modification of the information to be signed, either directly (modification of such data) or indirectly (fraudulent data are included by reference from the data to be signed).

D2-CAT2.1: Document modification. This subcategory considers modifications in the document to be signed.

D2-CAT2.1.1: Dynamic content inclusion. This subcategory implies the inclusion of dynamic content into the document to be signed. These methods aim at maintaining the document's integrity while varying its semantic.

D2-CAT2.1.1.1: Hidden code. The attacker inserts special tags or fields in the document to be signed. This hidden code will be translated into certain value depending on specific conditions controlled by the attacker.

D2-CAT2.1.1.2: Active code. The attacker inserts special code, like scripts or macros, in the document to be signed. This code is executed during the signed document opening or visualization, and can perform operations like changing the content being shown.

D2-CAT2.1.1.3: Linked content. The attacker inserts links in the document to be signed that point to external content not controlled by the signer. Once the signature is performed, the attacker can manipulate that external content at will.

D2-CAT2.1.2: Content modification. The attacker modifies the content of the document to be signed, but without including any sort of dynamic content (e.g. modification of the text of the document to be signed).

D2-CAT2.2: Attribute modification. This subcategory considers modifications performed in the attributes to be signed. The explanation given for subcategory *D2-CAT2.1: Document modification* applies, except that the object being modified is an attribute instead of the document.

D2-CAT2.2.1: Dynamic content inclusion.

D2-CAT2.2.1.1: Hidden code.

D2-CAT2.2.1.2: Active code.

D2-CAT2.2.1.3: Linked content.

D2-CAT2.2.2: Content modification.

D2-CAT2.3: DTBS modification. The attacker modifies the information that represents the data to be signed.

D2-CAT2.4: DTBSR modification. The attacker modifies the hash of the data to be signed. This would be the last data transformation step before the signature is computed.

D2-CAT3: Modification post signature computation

This category contains the methods that take part once the signature has been computed, and where the goal is the modification of the signed information, either signed directly (modification of the signed data) or indirectly (modification of data referenced from the signed data).

D2-CAT3.1: External content. The attacker modifies information referenced from the signed information (e.g. XSD, DTD). The difference between this method and *D2-CAT2.1.1.3: Linked content* or *D2-CAT2.2.1.3: Linked content* lies in that, in the former, the link to the external content is not included by the attacker, while in the latter, the link is explicitly inserted by the attacker.

D2-CAT3.2: Cryptanalysis. The attacker applies a cryptanalytic method to generate a document different to the signed one without breaking the signature validity.

D2-CAT3.2.1: Hash function. The attacker applies methods specifically focused on breaking the security of the hash function used in the signature computation. Assuming a hash function that generates an n-bit output, there are three possible attacks.

D2-CAT3.2.1.1: Collision attack. The attacker is able to find a pair of messages $M \neq M'$ where $hash(M) = hash(M')$ with a complexity lower than $O(2^{n/2})$ (e.g. The birthday attack).

D2-CAT3.2.1.2: Preimage attack. The attacker, given a hash value H, is able to find a message M′ where $H = hash(M')$ with a complexity lower that $O(2^n)$.

D2-CAT3.2.1.3: Second preimage attack. The attacker, given one message M, is able to find a second message, M′, $M' \neq M$ to satisfy $hash(M) = hash(M')$ with a complexity lower than $O(2^n)$.

D2-CAT3.3: Signature replacement. The attacker substitutes the original signature by a signature generated by himself on the same message, compromising the proof of origin.

D2-CAT4: Unauthorized invocation of the signing function

This category collects the methods that do not permit the attacker to know the Signature Creation Data (SCD) but to make use of it without the user's consent and knowledge.

D2-CAT4.1: Compromise of the Signer Authentication Data (SAD). This subcategory covers the methods that permit the attacker to retrieve the SAD.

D2-CAT4.1.1: Social engineering. The attacker manipulates or tricks the signer to reveal the SAD.

D2-CAT4.1.2: SAD interception. The attacker intercepts the SAD during the SCS operation.

D2-CAT4.1.2.1: Observation. The attacker observes the SAD while the signer enters it in the Signature Creation System (SCS) (i.e. shoulder surfing).

D2-CAT4.1.2.2: Interception in interprocess/entities communication. The attacker intercepts the SAD during its transmission between logical or physical processes or entities belonging to the Signature Creation System (SCS) (e.g. sniffing techniques, software keyloggers, hooks, …).

D2-CAT4.1.2.3: Endpoint compromise. By having compromised a process or entity belonging to the Signature Creation System (SCS), and that intervene during the communication of SAD inside the SCS, the attacker is able to intercept the SAD when used (e.g. hardware keyloggers).

D2-CAT4.1.3: Guessing. The attacker uses a probabilistic method, brute force or keyboard acoustic emanation techniques to guess the SAD.

D2-CAT4.2: Authentication Bypass. The attacker bypasses the authentication method. As a result, the attacker is able to invoke the signing function without even knowing the SAD.

D2-CAT5: Compromise of the Signature Creation Data (SCD)

This category includes the methods that permit the attacker to retrieve the Signature Creation Data (SCD). Attacks classified under this category are the most dangerous ones, since the attacker would be able to make use of the SCD at will, even in a different environment.

D2-CAT5.1: SCD interception. The attacker intercepts the SCD during the creation or issuance processes.

D2-CAT5.1.1: Interception in interprocess/entities communication. The attacker intercepts the SCD during its transmission between logical or physical processes or entities.

D2-CAT5.1.2: Endpoint compromise. By having compromised a process or entity involved in the SCD creation, issuance, management or operation within the Signature Creation Environment (SCE) boundaries, the attacker is able to retrieve the SCD.

D2-CAT5.2: Eavesdropping (side-channel). Side-channel attacks exploit the information leakage from physical characteristics of the hardware during the execution of the cryptographic algorithm. Thereby, the cryptographic key can be guessed, and thus compromised. The complexity or security of the mathematical algorithm does not matter because the fundamentals of side-channel attacks rely on the dependencies between the data processed (e.g. the private key) and/or the operation performed by the cryptographic device (e.g. smart card) and the physical behavior of the underlying hardware.

D2-CAT5.2.1: Timing Analysis. A Timing Analysis attack exploits timing measurements from vulnerable systems to find the entire secret keys.

D2-CAT5.2.2: Electromagnetic Analysis. An Electromagnetic Analysis attack exploits correlations between secret data and variations in power radiations emitted by tamper-resistant devices, like smart cards.

D2-CAT5.2.3: Power Analysis. A Power Analysis attack analyses the relationship between the power consumption of a cryptographic device and the handled data during cryptographic operations.

D2-CAT5.2.4: Microarchitectural Analysis. Microarchitectural Analysis (MA) studies the effects of common processor components and their functionalities on the security of software cryptosystems. MA attacks exploit the microarchitectural components of a processor to obtain the cryptographic keys. These attacks are purely based on software, and can compromise the security system despite the implemented security techniques, such as virtualization, sandboxing or memory protection.

D2-CAT5.2.5: Optical Observation. Optical emanations can leak sensitive information. If the information being processed corresponds to the SCD, the attacker may compromise it by simply observing the optical signal being produced.

D2-CAT5.2.6: Fault Injection. A Fault Injection attack injects faults into the device (e.g. by tampering with the supply voltage) and analyzes the erroneous results produced by the device to obtain sensitive information.

D2-CAT5.3: Unauthorized access to the SCDev. The attacker compromises the SCD by accessing the (Secure) Signature Creation Device ((S)SCDev) (or software keystore) where it is stored.

D2-CAT5.3.1: Compromise of the Signer Authentication Data (SAD). The attacker is able to retrieve the SCD once the SAD is known. This method requires the SCD to be exportable. This subcategory is further refined using the same subcategories as *D2-CAT4.1: Compromise of the signer authentication data (SAD)*.

D2-CAT5.3.2: Authentication Bypass. The attacker is able to access the SCD even without knowing the SAD. This method requires the SCD to be readable by an entity different than the Signature Creation Device (SCDev) or the software keystore.

D2-CAT5.4: Cryptanalysis. The attacker applies a cryptanalytic method to discover the SCD.

D2-CAT5.4.1: Asymmetric Algorithm. This subcategory collects attacks focused on compromising the private key used in an asymmetric algorithm. Depending on the algorithm, the set of possible attack methods varies.

D2-CAT6: Influence on certificate verification result

This category includes methods of attack that have an impact during the verification of the certificate associated to the signature being verified. Some methods can be used to make a verifier conclude that either an invalid certificate is valid or that a valid certificate is invalid.

D2-CAT6.1: Alteration of subscriber's revocation request. The attacker alters the request made by the subscriber (legitimate owner of the certificate and associated private key) to revoke the certificate. This method is aimed at preventing the revocation of such certificate.

D2-CAT6.1.1: DoS of revocation request. The attacker performs a denial of Service (DoS) attack by preventing the request from reaching the certification authority in charge of processing the revocation.

D2-CAT6.1.2: Modification of revocation request. The attacker modifies the information of the request that identifies the certificate which revocation is being requested.

D2-CAT6.2: Alteration of certificate status verification. The attacker alters the certificate status verification process, causing the verifier to conclude that an invalid certificate (i.e. revoked or suspended) is valid, or that a valid certificate is invalid.

D2-CAT6.2.1: Grace or cautionary period bypassing. This subcategory collects methods that allow the attacker to bypass or make the grace/cautionary period, as defined in CEN Workshop Agreement 14171 (2004), ineffective.

D2-CAT6.2.1.1: Delay in time-stamped signature sending. The attacker delays the time-stamped signature sending until the Certificate Revocation List (CRL) is updated. This method assumes that the legitimate owner of the certificate cannot detect the private key compromise before the attacker makes use of the signed document and the corresponding signature. To implement this method, the attacker must have compromised the private key, generated a signature on behalf of the user and time-stamped it on their own. As a result, when the verifier receives the signature, he will possess a CRL issued after the signing time (specified by the time-stamp) and thus will not wait for any further update, considering the signature as valid.

D2-CAT6.2.1.2: Delay in time-marked signature sending. This method of attack is similar to *D2-CAT6.2.1.1: Delay in time-stamped signature sending*, with the difference that a time-mark is used instead of a time-stamp.

D2-CAT6.2.1.3: Exploit delay in CA's revocation request processing. This method of attack exploits the inevitable time that a Certification Authority (CA) needs to update the Certificate Revocation List (CRL) since the revocation request is received and processed. Therefore, and assuming that the attacker can use the private key of the victim, the attacker is able to enforce a signed document even though the owner had requested the revocation of the associated certificate.

D2-CAT6.2.2: Modification of certificate status verification request. The attacker alters the certificate status request made by the verifier to prevent him from discovering the actual revocation status, or to query the status of a different revoked certificate.

D2-CAT6.2.2.1: Modification of OCSP request. The attacker modifies the field *serialNumber* of the Online Certificate Status Protocol (OCSP) request structure (RFC 2560, 1999). This method makes the verifier request the status of a certificate different than the targeted one. In case the OCSP request is to be signed by the requester, then the attacker should perform the modification before the signing or compromise the OCSP signing key for a further modification and signature calculation. Also, as the standard establishes that the response must include the certificate serial number (to ascertain that the response is given for the desired certificate), and for this attack to succeed, the attacker should launch a secondary attack of type *D2-CAT6.2.3: Modification of certificate status verification response*, modifying the certificate serial number of the response.

D2-CAT6.2.2.2: Modification of LDAP-based request. This method of attack is similar to *D2-CAT6.2.2.1: Modification of OCSP request*, but being applied over a Lightweight Directory Access Protocol (LDAP) request.

D2-CAT6.2.3: Modification of certificate status verification response. This subcategory represents methods that modify the certificate status response given by the authority (e.g. CA, OCSP responder, etc.). The modification should be performed in a manner that the verifier accepts the message as valid and authentic.

D2-CAT6.2.4: Alteration of time reference verification. The attacker modifies the token used by the verifier as the time reference. The methods represented herein imply that the modification is made in a manner that cannot be detected by the verifier.

D2-CAT6.2.4.1: Modification of time-stamp. The attacker modifies the time reference included in the signature time-stamp in order to prevent the verifier from detecting the actual revocation status of the certificate at the time when the signature was generated.

D2-CAT6.2.4.2: Modification of time mark. The attacker modifies the time reference included in the signature time-mark to prevent the verifier from detecting the actual revocation status of the certificate at the time when the signature was generated. For this attack to be executed, the attacker needs to intercept the information sent by the time-mark authority to the verifier.

D2-CAT6.2.5: Validation information reply. The attacker reuses validation information to prevent the verifier from detecting the actual certificate revocation status.

D2-CAT6.2.5.1: OCSP response reply. The attacker replies with an outdated Online Certificate Status Protocol (OCSP) response that contains the *certStatus* field (RFC 2560, 1999) set to a value of interest to the attacker (e.g. 'good', for a certificate that is currently revoked, or 'unknown'/'revoked' for a valid certificate). Due to the time verification requirements established in RFC 5280 (2008), the attacker should modify the current time of the verifier's machine for this attack to succeed.

D2-CAT6.2.6: Alteration of certificate status verification result. The attacker intercepts the routine that performs the status verification process and alters the result that indicates the status of the certificate.

D2-CAT6.3: Untrusted trust anchor/trust point addition. The attacker injects a new trust anchor or trust point to make a certificate owned by the attacker be considered as trusted by the verifier during the certification chain verification. The attacker poses as the victim by using a certificate containing in the subject Distinguished Name (DN) field the victim's DN.

D2-CAT6.4: Alteration of certificate integrity verification result. The attacker intercepts the routine that performs the certificate integrity verification process and alters the result.

D2-CAT6.5: Alteration of certificate validity period verification result. This subcategory considers methods where the attacker alters the verification of the certificate validity period. For instance, the attacker may intercept the routine that performs the validity period verification process and alter the result to make the certificate be regarded as valid. The attacker may also modify the current time of the verifier's machine.

D2-CAT7: Influence on signature verification result

This category includes methods of attack that affect the verification of the signature being verified. Some methods can be used to make a verifier conclude that either an invalid signature is valid or that a valid signature is invalid. Methods specifically focused on influencing the verification of the signing certificate are included in *D2-CAT6: Influence on certificate verification result* category.

D2-CAT7.1: Presentation manipulation. This subcategory collects methods that manipulate the way the Data To Be Verified (DTBV) are visualized by the verifier. This set of methods violates the What-Is-Presented-Is-What-Is-Signed (WIPIWIS) principle.

D2-CAT7.1.1: DTBV masquerading. The attacker alters the visualization of the Data To Be Verified (DTBV), being able to present a DTBV different to what has been actually signed. This subcategory represents methods that focus on the way the DTBV is shown to the verifier, but independently of the viewer being used (e.g. superimposing text on the signed document during its visualization).

D2-CAT7.1.1.1: Document masquerading. The attacker alters the visualization of the signed document.

D2-CAT7.1.1.2: Attribute masquerading. The attacker alters the visualization of one or more signed attributes.

D2-CAT7.1.2: Viewer manipulation. The attacker manipulates the viewer used to present the Data To Be Verified (DTBV). In this case, the methods lie in achieving a different visualized DTBV by targeting the viewer used.

D2-CAT7.1.2.1: Viewer substitution. The attacker substitutes the viewer with another one that presents the Data To Be Verified (DTBV) in a different manner.

D2-CAT7.1.2.2: Alteration of viewer's behavior. The attacker alters the behavior of the viewer to make it present the Data To Be Verified (DTBV) in a different manner.

D2-CAT7.1.3: Verification result masquerading. The attacker manipulates the signature verification result shown to the verifier. A valid signature may be presented as invalid, or an invalid signature as valid.

D2-CAT7.2: Policy substitution. The attacker replaces a policy used for the signature verification.

D2-CAT7.2.1: Electronic signature policy substitution. The attacker replaces the electronic signature policy referenced in the signature and that contains the clauses and requirements that establish the conditions under which the signature should be considered as valid.

D2-CAT7.2.2: Certificate policy substitution. The attacker replaces the certificate policy referenced in the certificate and that contains a named set of rules that indicates the applicability of a certificate to a particular community and/or class of application with common security requirements.

D2-CAT7.3: Alteration of verification process. This subcategory includes methods that affect the processes that implement the signature verification process, in a manner that the achieved result differs from what is expected.

D2-CAT7.3.1: Injection of signature-signed data pair. The attacker replaces the information during the verification process by injecting a different signed document-signature. It is assumed that the attacker possesses a document signed by the signer and the corresponding signature, but

different to the signed document and signature that is to be verified.

D2-CAT7.3.2: Alteration of cryptographic verification result. The attacker intercepts the routine that performs the cryptographic verification process and alters the result.

D2-CAT7.3.3: Alteration of final verification result. The attacker is able to alter the final result of the signature verification process, influencing the conclusion about the validity or invalidity of the signature.

Not every method of attack permits the attacker to achieve every attacker's goal established for the first dimension. Table 2 shows the relationship between the categories of the first dimension with the categories in the first level of the second dimension.

7.3. *Dimension three: target of the attack*

This dimension classifies the target(s) of the attack. An attack can target more than one element at the same time, resulting in multiple entries in this dimension. It does not mean that the mutual exclusion principle is violated (see Section 10). A non-mutually exclusive taxonomy would produce two different entries for the same target. In this case, several targets may need to be classified for the same attack.

Next, the categories and subcategories of this dimension are listed:

D3-CAT1: Cryptography
D3-CAT2: Software

Table 2 – Relationship between the dimension *Attacker's Goal* and dimension *Method of Attack*.

Goal	Method
D1-CAT1: Deceive the signer to sign a document different to the intended one	D2-CAT1: Environment manipulation D2-CAT2: Modification prior to signature computation
D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)	D2-CAT4: Unauthorized invocation of the signing function D2-CAT5: Compromise of the Signature Creation Data (SCD)
D1-CAT3: Replace signed information	D2-CAT3: Modification post signature computation
D1-CAT4: Attribute the signed document to a user different to the actual signer	D2-CAT6: Influence on certificate verification result D2-CAT7: Influence on signature verification result D2-CAT3: Modification post signature computation (Only D2-CAT3.3: Signature replacement)
D1-CAT5: Make the Data To Be Verified (DTBV) show chosen content	D2-CAT1: Environment manipulation D2-CAT7: Influence on signature verification result
D1-CAT6: Make the signature validity verification conclude with an opposite result	D2-CAT6: Influence on certificate verification result D2-CAT7: Influence on signature verification result

D3-CAT2.1: Application.
D3-CAT2.1.1: External application.
D3-CAT2.1.1.1: Kernel level application.
D3-CAT2.1.1.2: User level application.
D3-CAT2.1.2: Related application.
D3-CAT2.1.2.1: Document processor.
D3-CAT2.1.2.2: Signature Creation Application (SCA).
D3-CAT2.1.2.3: Cryptographic Service Provider (CSP).
D3-CAT2.1.2.4: SCDev.[1]
D3-CAT2.1.2.5: Signature Verification Application (SVA).
D3-CAT2.1.2.6: Certification Authority (CA).
D3-CAT2.2: Driver.
D3-CAT2.2.1: Keyboard driver.
D3-CAT2.2.2: Video card driver.
D3-CAT2.2.3: Secure Signature Creation Device (SSCDev) driver.
D3-CAT2.2.4: Fingerprint reader driver.
D3-CAT2.2.5: Network driver.
D3-CAT2.3: Operating system.
D3-CAT2.4: Network.
D3-CAT2.4.1: Protocols.

D3-CAT3: Hardware
D3-CAT3.1: Secure Signature Creation Device (SSCDev).
D3-CAT3.2: Computer.
D3-CAT3.2.1: Trusted Platform Module (TPM).
D3-CAT3.2.2: Hard-disk.
D3-CAT3.2.3: Memory.
D3-CAT3.2.4: Peripheral devices.
D3-CAT3.2.4.1: Monitor.
D3-CAT3.2.4.2: Keyboard.
D3-CAT3.2.4.3: Smart card reader.
D3-CAT3.2.4.4: Fingerprint reader.
D3-CAT3.3: Network equipment.
D3-CAT3.3.1: Communication buses.

D3-CAT4: Human user
D3-CAT4.1: Signer.

D3-CAT5: Information
D3-CAT5.1: Document.
D3-CAT5.2: Protocol message.
D3-CAT5.3: Cryptographic material.
D3-CAT5.3.1: Trust store.
D3-CAT5.3.2: Time-stamp.

8. Method of classification

The method of classification associated to a taxonomy must clearly guide a user when a new phenomenon has to be classified. Next steps comprise the method of classification designed to classify an attack under our taxonomy. As can be noted, last step permits further refinements when required.

1. **Attack analysis**. The attack must be analyzed in order to understand its behavior and features. Depending on the available information, the result of the analysis will be more or less detailed and accurate. This information

[1] This subcategory includes the software signature creation device and the software keystore as defined in Section 6.

should, at least, permit the completion of the remaining steps of the classification method.
2. **Identification and classification of the attacker's goal**. The goal of the attack must be identified and classified according to the dimension *Attacker's Goal*.
3. **Analysis and classification of the method of attack**. The method used by the attacker to achieve the identified goal must be classified according to the dimension *Method of Attack* and Table 2. The method of attack must be classified in a subcategory of the deepest level of the selected branch.
4. **Identification and classification of targets of the attack**. The elements affected by the attack must be identified and classified in accordance with the subcategories found in dimension *Target of the Attack*. Like in the previous step, a subcategory of the deepest level of the selected branch must be selected. Only targets directly involved in the signature creation or verification operation should be classified. For instance, any internal attack carried out by means of malware must firstly compromise the system (e.g. due to a vulnerability in the operating system). However, the operating system should not be classified as a target unless it had a vulnerability that allowed the attacker to directly compromise the generation/verification process.
5. **Refine the taxonomy**. In steps 2, 3 and 4, if a more specific or refined subcategory is needed, it must be added to the taxonomy, and the attack classified accordingly. Due to the taxonomy design, the refinement does not need to modify existent categories or subcategories, without having an impact on already classified attacks.

As a result, an attack will be classified using one category of dimension *Attacker's Goal*, one subcategory of dimension *Method of Attack*, and one or more subcategories of dimension *Target of the Attack*.

As mentioned above, the accuracy and detail extracted from the attack analysis depends on the available information. Obscure attacks or attacks from which little information can be obtained will necessarily be more complicated to classify. On the other hand, attacks that can be studied in detail, for instance applying re-engineering techniques to the malware code or during a forensic study, will provide much more information that can be used to accurately define the attack behavior and features, and thus to classify the attack.

Next, and for illustration purposes, a Trojan horse attack on software for electronic signatures (Spalka et al., 2001) is classified using our method of classification.

Firstly, we have studied the information provided in Spalka et al. (2001). The attack is carried out on two of the most deployed signature applications in Germany. The attacker obtains a handle to the Personal Identification Number (PIN) edit control in a Windows operating system environment. Once the user has entered the PIN, the attacker is able to retrieve it and start as many signing processes as desired. The authors do not provide the attack vector to infect the system with the Trojan horse, though we can assume that the environment does not provide effective protection for detecting this specific malware. Otherwise, the attack would have been thwarted in the very beginning. Therefore, it can be assumed that there is a vulnerability in the system that the attacker can exploit for the infection.

Secondly, and using the information above, the goal of the attack has to be classified. In this case, the main objective of the attacker is to use the signature creation data without the user's consent. Therefore, we classify the goal as *D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)*.

The next step establishes that the method used by the attacker to achieve the identified goal must be classified. Table 2 restricts the candidates to two. Because the attack does not retrieve the SCD, and due to the description given, it is obvious that the method intends to use the SCD by invoking the signing function. Therefore, we classify the method of attack as *D2-CAT4: Unauthorized invocation of the signing function*. However, the method of classification stipulates that a subcategory of the deepest level of the selected branch must be selected. Taking a look at D2-CAT4 subcategories, it is clear that the attacker is compromising the signer's authentication data (*D2-CAT4.1: Compromise of the Signer Authentication Data (SAD)*). In particular, the attacker obtains the PIN. Then, in a deeper classification, the next subcategory corresponds to *D2-CAT4.1.2: SAD interception*, and more specifically, *D2-CAT4.1.2.3: Endpoint compromise*, as the PIN is retrieved due to a vulnerability in the PIN edit control of the signature creation application.

Finally, the description of the attack permits us to identify the target of the attack, which is the SCA. Table 3 shows the final result of the classification.

It is important to highlight that every method is subject to the subjectivity of the user in charge of the classification. Even when the available information of the attack is very detailed, two different users can reach contradictory conclusions. The training, skills and perspective of the user are paramount to make the correct decision. However, sometimes there is not only a single correct decision, but also many. For example, the same attack can be correctly classified in two different manners depending on the viewpoint taken. An attack that injects dynamic content into the document to be signed can be classified according to the goal dimension as *D1-CAT1: Deceive the signer to sign a document different to the intended one*, if the attacker is not the signer, and the user concludes that the process being subverted is the signature generation, or as *D1-CAT5: Make the Data To Be Verified (DTBV) show chosen content*, if the attacker is the signer itself or a different malicious entity, but the user considers that the process being subverted is the signature verification. Possibly, the goal pursed by the attacker

Table 3 – Classification of Spalka et al. (2001) Trojan horse attack.

Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT4: Unauthorized invocation of the signing function → D2-CAT4.1: Compromise of the Signer Authentication Data (SAD) → D2-CAT4.1.2: SAD interception → D2-CAT4.1.2.3: Endpoint compromise
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.2: SCA

is both of them, that is, both deceiving the signer respecting the information being signed and deceiving the verifier respecting the information signed.

The method proposed herein intends to reduce the ambiguity during the classification procedure, but we do not claim that the method is deterministic, since we consider that it is not possible in this inexact field of study.

9. Survey and classification of attacks on digital signatures

An intensive survey and classification of 117 attacks on digital signatures found in the literature (a few proposed by the authors) has been made, and its results are included in the Appendix A. We have found a significantly higher number of attacks involved in the signature creation process (85 attacks) than attacks intended to subvert the verification process (32 attacks).

It should be mentioned that the survey of attacks does not intend to demonstrate a statistical distribution of the types of attacks on digital signatures. The attacks have been selected from the literature according to their relevance. As a result, no strong conclusion should be made on the likelihood of occurrence of each type of attack. Despite this, we do think that some conclusions can be made respecting the impact, dangerousness and profile of the surveyed attacks.

Fig. 2 depicts the number of attacks per goal category. It is clear that threats to the signature generation process (represented by categories D1-CAT1, D1-CAT2 and D1-CAT3), and in particular those pursuing goals D1-CAT1 (26 out of 117 (22.2%)) and D1-CAT2 (51 out of 117 (43.6%)), are the most attractive ones for both attackers and researchers. In our opinion, the justification lies in that the generation process is the most critical stage during the life-cycle of a signature, and also the one that is most profitable for the attacker if compromised. In this sense, we see that most of the attacks (almost half of the total) are designed to use the signature creation data for malicious purposes (goal D1-CAT2), followed by attacks aimed at deceiving the user during the signing process (goal D1-CAT1). Few attacks pursued goal D1-CAT3 (8 out of 117 (6.8%)). Observing the attacks on the verification stage, we found few attacks – only 3 – aimed at tricking the verifier respecting the identity of the signer, represented by goal D1-CAT4 (3 out of 117 (2.5%)), while the number of attacks according to goal dimensions D1-CAT5 (13 out of 117 (11.1%)) and D1-CAT6 (16 out of 117 (13.6%)) is more balanced.

We consider that it is important to analyze the distribution of attack categories in two cases: the number of attacks that focused on each target versus the goal dimension, and the number of attacks that employed each method of attack versus the goal dimension. These two viewpoints will permit us to discover the targets and methods involved in the surveyed attacks.

Fig. 3 shows that the most commonly affected targets, at the generation stage (goals D1-CAT1, D1-CAT2 and D1-CAT3), are: the Signature Creation Application (SCA) (15 attacks), the Secure Signature Creation Device (SSCDev) (15 attacks), the Signature Creation Device (SCDev) (14 attacks), the Document processor (11 attacks), and the Document to be signed (11 attacks). On the other hand, most commonly affected targets, from the verification viewpoint, are: the Signature Verification Application (SVA) (13 attacks) and the Document processor (10 attacks). These elements are directly involved during the signing and verification operations. Therefore, it is reasonable to think that they are more likely to be attacked than other system components. Consequently, these elements should be carefully designed and implemented to increase the level of assurance of their correctness and trustworthiness. Nonetheless, the existence of vulnerabilities or weaknesses in other components, like the underlying operating system (6 attacks) or the cryptography used (9 attacks), may open the door for an attack to succeed, no matter how reliable the aforementioned elements are.

Targets with zero mappings mean that they were not found in the surveyed attacks. However, they can also be a potential victim in an attack on digital signatures. Their direct or indirect participation during a signing operation make them a potential target as well.

On the other hand, we present the distribution of methods of attack versus the goal in the cases of the generation and verification phases. Fig. 4 shows that the distribution of methods of attack applicable to the generation stage versus the

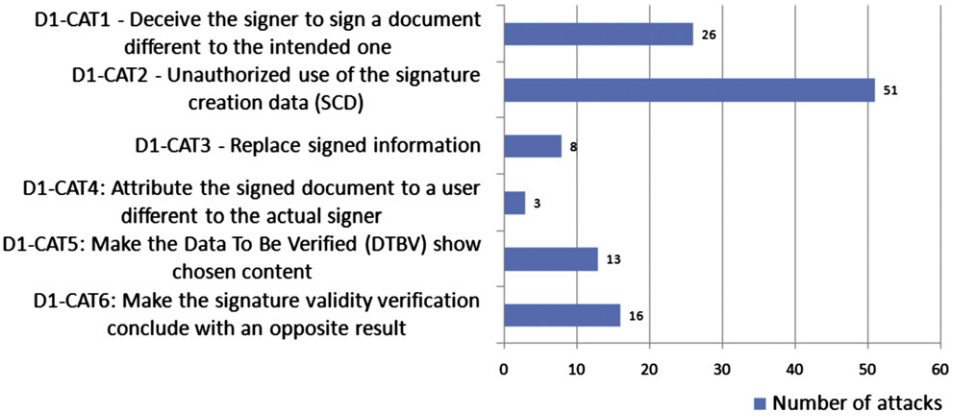

Fig. 2 – Number of attacks per goal category.

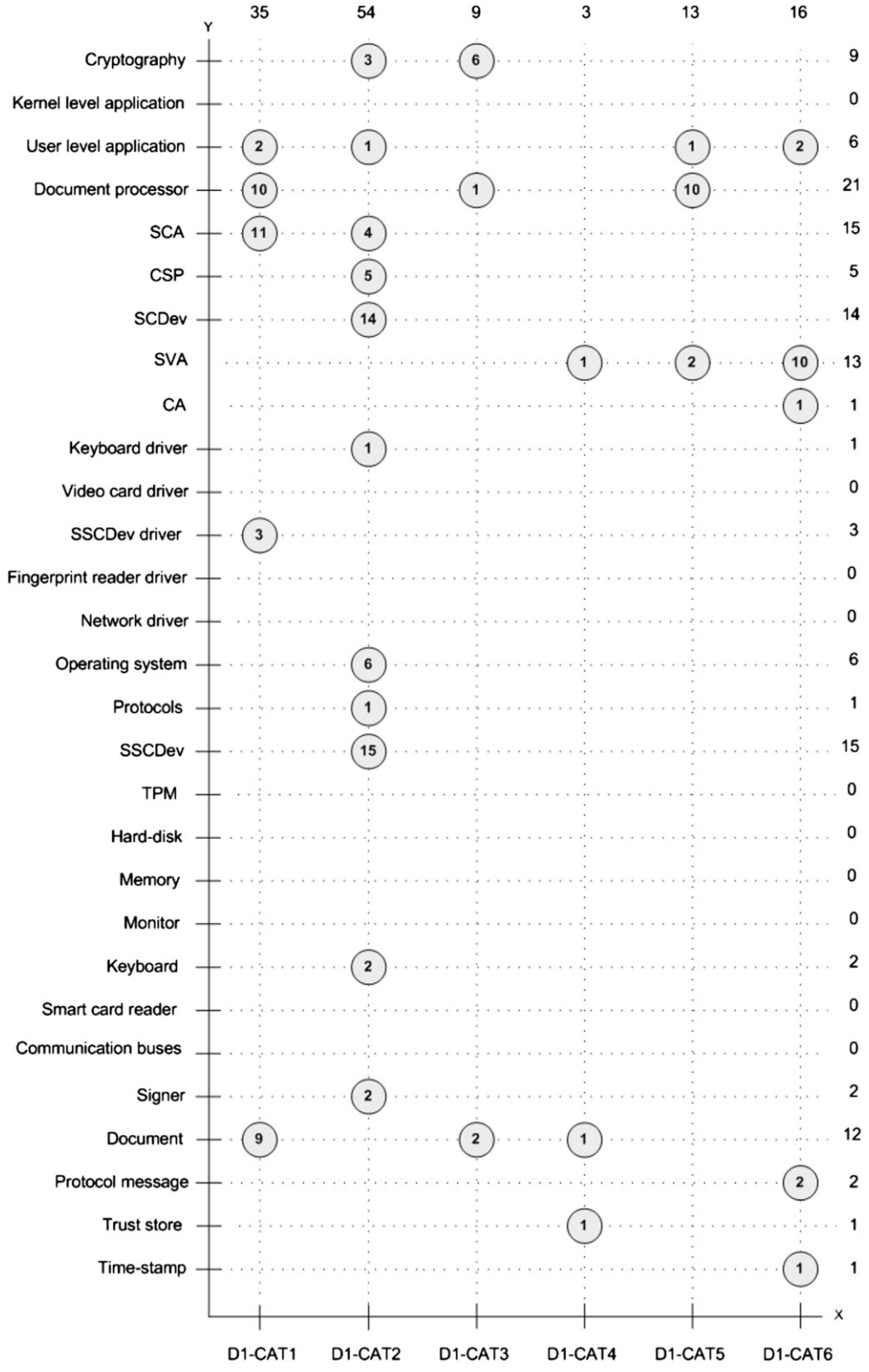

Fig. 3 – Distribution of attacks: target versus goal.

goal dimension is homogeneous, though the collection of side-channel attacks (23 attacks in total), content modification methods (10 attacks) and authentication bypass (9 attacks) prevail. In our opinion, this homogeneous distribution proves that there is an array of attack methods that can undermine the security of a signing operation. Also, the distribution respecting axis Y of Fig. 4 demonstrates the specificity of the attacks, each of which is employed to achieve a single specific goal.

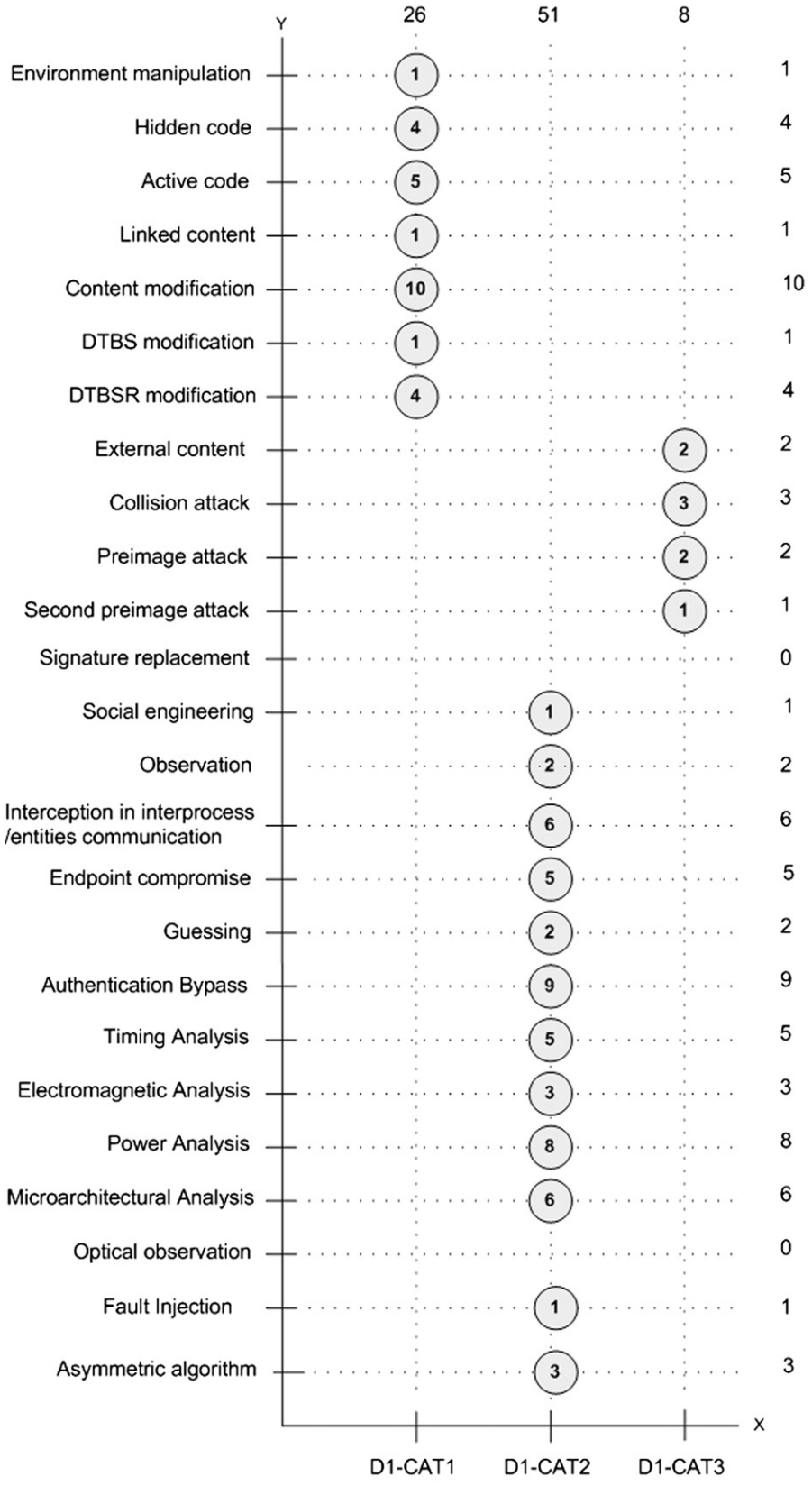

Fig. 4 – Distribution of attacks: method versus goal (generation).

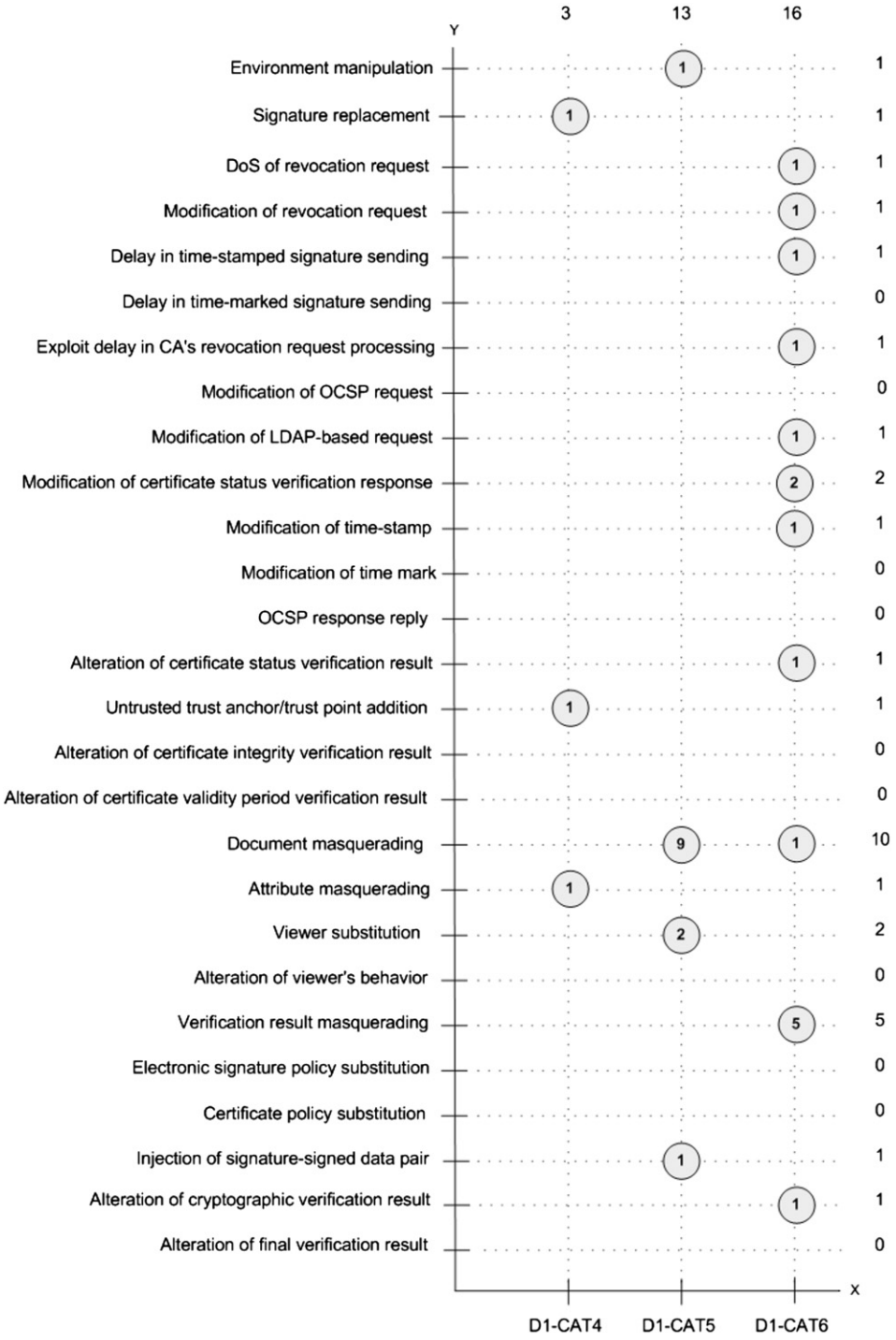

Fig. 5 – Distribution of attacks: method versus goal (verification).

Finally, Fig. 5 illustrates the distribution of methods of attack applicable to the verification stage versus the goal dimension. As in the previous distribution, it can be seen that there is a clear specificity in the attacks surveyed. Also, two methods prevail: those focused on modifying the appearance of the signed document by means of document masquerading attacks (10 attacks), and, to a lesser extent, methods that masquerade the verification result shown (5 attacks).

As can be seen in Figs. 4 and 5, the method of attack raises accurate information about the pursued goal, and vice versa. Each classified method is mapped to just one goal, contrary to the target distribution, where some targets are mapped to more than one goal. This can help making informed decisions when implementing security measures to counteract certain types of attacks or avoid the attacker to achieve a certain goal. As mentioned before, attacks that pursue goal D1-CAT2 are

the most dangerous ones, especially those that compromise the signature creation data. Consequently, systems should be designed and implemented to particularly mitigate the risks associated to attacks that entail goal D1-CAT2.

10. Evaluation of the taxonomy

Lough (2001) reviewed and collected the agreed requirements any taxonomy should fulfill. In this Section we evaluate our taxonomy against these requirements.

A taxonomy should be **generally acceptable** in the field of application for which it is designed. Obviously, this property can be satisfied only if the taxonomy is accessible by others and approved as valid after some time of study. The taxonomy proposed in this paper builds on previous work that has had relevant impact in the scientific community. The taxonomy follows the well-known concept of dimension, which has been proved to be a good way for providing a holistic view of the field of study. Though it is still to be seen if our taxonomy is accepted by the community, we are confident that it will.

A taxonomy should be **exhaustive** in the sense that it covers all known related specimen. This property is hard to fulfill, since the classification of every known phenomenon is near impossible, especially in such a dynamic field like the information technology. However, the evaluation of a taxonomy against real samples is paramount to verify its correctness and completeness. The larger the number of samples classified, the higher the level of confidence in that the taxonomy reliably captures the knowledge in a comprehensive manner. Also, if the taxonomy evolves as depicted in Fig. 1, then it can be assumed that the number of phenomena classified is representative enough, and thus the taxonomy obtained can be considered as reliable to that area of knowledge. To support this hypothesis, the methodology used to select those samples or phenomena is important as well. In the process of creating a taxonomy, a methodology that reduces the probability of a biased approach also increases the confidence in the taxonomy. In our particular case, we have successfully classified 117 attacks (see Section 9), with only two new subcategories created during the last classifications. Notwithstanding, our method of classification permits the taxonomy to evolve along the time due to the refinement stage, being able to create new categories if required. In addition, the methodology followed to identify relevant attacks (those attacks used to create the taxonomy categories) was based on well-known practices for systematic reviews (Kitchenham, 2004), reducing the probability of a biased approach driven by the researcher expectations.

A taxonomy should be **mutually exclusive**. Each specimen should be classified under, at the most, one category of the taxonomy. The method of classification provided in Section 8 and the design of the taxonomy assure that an attack cannot be classified into multiple categories in a dimension. The possibility to select several subcategories in dimension *Target of the Attack* does not violate this principle, but allows to classify several elements affected by the attack, if necessary.

A taxonomy should be **comprehensible** in a manner that it should be understandable and applicable by non-expert users. On the contrary, our taxonomy requires specific IT security knowledge, requiring the person in charge of the classification to have a deep understanding of security and the attack itself.

A taxonomy should be **deterministic** and **repeatable**. The method applied for the classification should be clear and unambiguous, and it should be possible to repeat the classification of a specimen, obtaining the same result as in previous classifications of the same specimen. In this work, a simple but effective method of classification is provided along with the taxonomy, facilitating a trained user to fulfill the classification task. However, we do not guarantee that our method of classification is deterministic, though we hope that it can lead to homogeneous classifications when the available information of the attack is detailed enough.

A taxonomy should **use widely accepted terminology** and be **appropriate**. The terms and definitions used by the taxonomy should comply with established and well-known terminology, and it should be based on a reference model and a well-defined set of restrictions (if any) (Amoroso, 1994). The proposed taxonomy is based on standard system models (CEN Workshop Agreement 14170, 2004; CEN Workshop Agreement 14171, 2004) and a well-defined attacker profile, using terms extracted from widely accepted and standard sources. The provided reference model assures that the person in charge of classifying or searching for an attack can know exactly which is the underlying model of applicability.

A taxonomy should be **focused** in order to be useful, being specific to a certain field of knowledge. This taxonomy is particularly focused on attacks on digital signatures, and more specifically on those that may affect the security of the signing and verification processes, which are the most critical stages in the digital signature life-cycle.

Finally, a taxonomy should be **useful** for the users belonging to the field of application. We believe that this taxonomy fills a current gap in the field of digital signatures, once their relevance and importance have become obvious after the approval of specific legislation and standards, the spread of related technology and their common application in real-life online scenarios. This systematic categorization of attacks on digital signatures will allow developers to build more robust and secure solutions, counteracting current attacks by designing countermeasures of general applicability.

11. Conclusions

The importance of the digital signature, supported by the current legislation and international standards, along with the consequences that derive from its use, encourage us to design robust and secure solutions that counteract the large number of current threats. A taxonomy of attacks would permit to categorize potential attacks in a generic and reusable manner, enabling the rigorous and systematic classification of real attacks and devising countermeasures of general applicability.

In this paper, the first taxonomy that studies the security problem of digital signatures in a holistic approach has been proposed. The taxonomy is focused on the signature creation and signature verification environments, covering the most sensitive stages within the signature life cycle. Three

dimensions have been defined, including the goal, method and targets of the attack. The work is complemented with a method of classification that facilitates the usage of the taxonomy in a clear and straightforward manner. In addition, the paper includes the methodology used to derive the taxonomy, and which provides confidence regarding its completeness and representativeness.

The taxonomy has been validated according to the set of general requirements any taxonomy should fulfill. Moreover, an intensive survey and classification of 117 attacks has been done, supporting the correctness and exhaustiveness of our taxonomy. The results obtained from the survey suggest that there is an array of methods an attacker can employ to subvert the security of digital signatures. Besides this, we have observed that there is a reduced list of most probable targets. Consequently, developers should pay special attention during their design and implementation. Notwithstanding, a vulnerability in any other element can also have an impact on the reliability of the digital signature.

Appendix A. Classified attacks on digital signatures

This appendix includes 117 attacks on digital signatures that have been classified using the taxonomy and method of classification proposed in the paper.

The surveyed attacks are a mix of real-world attacks on specific commercial products and devised theoretical attacks that could be put into practice. Some of the 32 attacks considered on the verification stage are firstly proposed here.

A representative name, the reference to the attack, a short description and possible countermeasures are provided for each classified attack. It is important to remark that, when a countermeasure is provided, the information given by the author of the attack and the specific conditions of the attack itself have been considered. Therefore, it should not be concluded that the countermeasure is of general applicability, and thus it should be evaluated on a case-by-case basis.

Name:	Dali attack
Source:	The Dali Attack on Digital Signature (Buccafurri et al., 2008)
Description:	Attack based on the capability of a file of having a static polymorphic behavior. The attacker modifies the document to be signed to include a secondary content different than the purported one. Thanks to certain formats tagging, the content shown to the verifier varies depending on the file extension, and thus the application chosen to open the file. The attack is limited to the inclusion of HTML as the malicious secondary content.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.1: Dynamic content inclusion → D2-CAT2.1.1.1: Hidden code
Target(s):	D3-CAT5: Information → D3-CAT5.1: Document D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Inclusion of the signed attribute *content-type* in the electronic signature format (e.g. CAdES, XAdES)

Name:	Enhanced Dali attack
Source:	Fortifying the Dali Attack on Digital Signature (Buccafurri et al., 2009)
Description:	Attack that enhances the Dali Attack to permit the usage of tiff and PDF formats for the contents inserted in the document to be signed.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.1: Dynamic content inclusion → D2-CAT2.1.1.1: Hidden code
Target(s):	D3-CAT5: Information → D3-CAT5.1: Document D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Use of PDF/A formats. Use of PDF Advanced Electronic Signature (PAdES) formats. Inclusion of the signed attribute *content-type* in the electronic signature format (e.g. CAdES, XAdES)

Name:	Signature replacement attack
Source:	Signature Replacement Attack and its Counter-Measures (Sinha and Sinha, 2010)
Description:	2-tuple Digital Signature schemes are subject to a specific attack where the attacker substitutes the original signature by a signature generated by himself on the same message, compromising the proof of origin. Due to the nature of these schemes, where the signature and the message may be loosely bound, the verified is not able to detect the attack, attributing the message to a different origin.
Goal:	D1-CAT4: Attribute the signed document to a user different to the actual signer
Method:	D2-CAT3: Modification post signature computation → D2-CAT3.3: Signature replacement
Target(s):	D3-CAT5: Information → D3-CAT5.1: Document
Countermeasures:	Bind the message with the signer's private key or related identity information. For instance, advanced electronic signature formats (e.g. CADES and XAdES) cover the public key certificate as signed data

Name:	Enhanced Dali attack (2)
Source:	Hiding Malicious Content in PDF Documents (Popescu, 2011)

Description:	Another attack that applies the enhanced Dali Attack to permit the usage of tiff and PDF formats for the contents inserted in the document to be signed.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.1: Dynamic content inclusion → D2-CAT2.1.1.1: Hidden code
Target(s):	D3-CAT5: Information → D3-CAT5.1: Document D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Use of PDF/A formats. Use of PDF Advanced Electronic Signature (PAdES) formats. Inclusion of the signed attribute *content-type* in the electronic signature format (e.g. CAdES, XAdES)

Name:	Cut and paste attack
Source:	Cut and paste attacks with Java (Lefranc and Naccache, 2002)
Description:	Attack focused on using a malicious applet to modify regions of the visualization area of a web browser while surfing through the Internet. This attack could be mounted to modify a malicious document visualized by the signer before computing the signature in order to fit with the expected one.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.2: Content modification
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.1: External application → D3-CAT2.1.1.2: User level application
Countermeasures:	Disable Java Virtual Machine

Name:	PIN retrieval
Source:	Trojan Horse Attacks on Software for Electronic Signatures (Spalka et al., 2002)
Description:	The attack is carried out on two of the most deployed signature software in Germany. The attacker obtains a handle to the PIN edit control in a Windows operating system environment. Once the user has entered the PIN, the attacker is able to retrieve it and start as many signing processes as desired. The authors provide an example in Delphi source code.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT4: Unauthorized invocation of the signing function → D2-CAT4.1: Compromise of the signer authentication data (SAD) → D2-CAT4.1.2: SAD interception → D2-CAT4.1.2.3: Endpoint compromise
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.2: SCA
Countermeasures:	Avoid handles belonging to applications different than the one that created the PIN window. Use of specialized hardware (e.g. keyboard with an integrated smart card reader)

Name:	PIN retrieval in email signing software
Source:	Trojan Horse Attacks on Software for Electronic Signatures (Spalka et al., 2002)
Description:	The same attack as in *PIN retrieval* is carried out on two products that consist of a signature plug-in integrated in a widely used email client software. The attacker is able to capture the PIN or password entered by the user.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT4: Unauthorized invocation of the signing function → D2-CAT4.1: Compromise of the Signer Authentication Data (SAD) → D2-CAT4.1.2: SAD interception → D2-CAT4.1.2.3: Endpoint compromise
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.2: SCA
Countermeasures:	Use of a smart card with specialized hardware (e.g. keyboard with an integrated smart card reader)

Name:	PIN retrieval (with keypad)
Source:	Trojan Horse Attacks on Software for Electronic Signatures (Spalka et al., 2002)
Description:	In this case, the *PIN retrieval* attack is performed on a commercial off-the-shelf product that implements a keypad for the secure input of the PIN. The attacker is able to access the permutation information, and thus is able to retrieve the PIN selected by the user.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT4: Unauthorized invocation of the signing function → D2-CAT4.1: Compromise of the Signer Authentication Data (SAD) → D2-CAT4.1.2: SAD interception → D2-CAT4.1.2.3: Endpoint compromise
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.2: SCA
Countermeasures:	Use of a smart card with specialized hardware (e.g. keyboard with an integrated smart card reader)

Name:	Modification of the secure viewer's presentation
Source:	Trojan Horse Attacks on Software for Electronic Signatures (Spalka et al., 2002)
Description:	This attack violates the What-You-See-Is-What-You-Sign (WYSIWYS) principle. The attack consists in manipulating the information shown by the secure viewer of a commercial off-the-shelf signature software. As a result, the attacker is able to modify the data to be signed while deceiving the user during the last confirmation step. The authors provide an example in Delphi source code.

Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.3: DTBS modification
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.2: SCA
Countermeasures:	–

Name:	Modification of the DTBSR
Source:	Trojan Horse Attacks on Software for Electronic Signatures (Spalka et al., 2002)
Description:	The attacker basically monitors the communication between the signature software and the smart card in order to modify the hash of the data sent to the card (the DTBSR).
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.4: DTBSR modification
Target(s):	D3-CAT2: Software → D3-CAT2.2: Driver → D3-CAT2.2.3: SSCDev driver
Countermeasures:	Implement a communication protocol compliant with ISO 7816 security measures

Name:	Mail forgery
Source:	Practical Security Aspects of Digital Signature Systems (TR-Seclab-0606-001, 2006)
Description:	The attacker aims at replacing the content of an email with arbitrary data, retaining the validity of the signature. For this purpose, the attacker launches an SMTP proxy on the compromised computer to intercept the communication between the mail client and the mail server in order to change the mail content. In order to perform the Man-In-The-Middle (MITM) attack, the attacker changes the preference settings of the mail client (Thunderbird in this case) such that the connection to the mail server is redirected to the proxy. The attacker also uses a specific tool called detours for binary interception of Win32 functions, and which permit him to load a dynamic link library (DLL) with the email client. This DLL is then used to intercept the function that initiates the digital signature process, replacing the content being signed with the malicious document.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.2: Content modification
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.2: SCA
Countermeasures:	–

Name:	Secure Viewer manipulation (1)
Source:	Practical Security Aspects of Digital Signature Systems (TR-Seclab-0606-001, 2006)
Description:	In this attack, the secure viewer component delivered for use of the Austrian citizen card (trustview component of trustdesk basic suite) is compromised. The attack consists of two steps. In the first step, detours tool (see previous attack) is used to modify the Windows file access routines in the Windows runtime library in a manner that trustview shows a file different than the one for which the signature request has been made. In the second step, the attacker alters the functions that display the content to be signed in order to show the original one, that is, the one intended by the user. For this purpose, the attacker obtains a window handle to the HTML control used by trustview, being able to edit the content shown therein.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.2: Content modification
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.2: SCA
Countermeasures:	–

Name:	Secure Viewer manipulation (2)
Source:	Practical Security Aspects of Digital Signature Systems (TR-Seclab-0606-001, 2006)
Description:	In this case, the same attack as in Secure Viewer manipulation (1) is carried out on the secure viewer component of HotSign product, also delivered for use of the Austrian citizen card. Here, though no HTML control object is used, the attacker is still capable of changing the appearance of the shown document by obtaining a handler to the secure viewer window and subsequently drawing directly over the window context.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.2: Content modification
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.2: SCA
Countermeasures:	–

Name:	Secure viewer compromise (1)
Source:	Malware Attacks on Electronic Signatures Revisited (Langweg, 2006)
Description:	The attack is carried out on Deutsche Telekom T-Telesec Signet 1.6.0.4 product. The attack does not need administrator privileges and relies on design flaws, not implementation ones. In particular, by using

	Windows messages the Signet software can be made to display a different information regarding the data to be signed. By placing an inactive window at the top of the z-order with a fake button representing the execution of the secure viewer, the user can be tricked into clicking on it, allowing the malware to show the purported document to be signed while sending a different one to the SCD. In addition, it is possible to draw on the viewer's presentation surface, allowing an attacker that has modified the document to be signed to represent it in its original form.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.2: Content modification
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.2: SCA
Countermeasures:	Fix data before it becomes obvious to an attacker that the data is relevant for signing

Name:	PCS/SC card reader communication potential compromise (1)
Source:	Malware Attacks on Electronic Signatures Revisited (Langweg, 2006)
Description:	The attack is carried out on Deutsche Telekom T-Telesec Signet 1.6.0.4 product. The attack does not need administrator privileges and relies on design flaws, not implementation ones. In particular, the attacker installs a modified WINSCARD.DLL in the Signet's folder. This file is thus loaded and executed by Signet, giving access to its address space and permitting arbitrary malicious actions, like DTBSR modification.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.4: DTBSR modification
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.2: SCA
Countermeasures:	Make the signing software to verify the signed code of every module used

Name:	Secure viewer compromise (2)
Source:	Malware Attacks on Electronic Signatures Revisited (Langweg, 2006)
Description:	The attack is carried out on IT Solution trustDesk standard 1.2.0 product. The attack does not need administrator privileges and relies on design flaws, not implementation ones. In particular, the attack draws on the secure viewer's presentation to deceive the user respecting the data to be signed, while the information to be sent for signing differs.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.2: Content modification
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.2: SCA
Countermeasures:	Fix data before it becomes obvious to an attacker that the data is relevant for signing

Name:	PCS/SC card reader communication potential compromise (2)
Source:	Malware Attacks on Electronic Signatures Revisited (Langweg, 2006)
Description:	The attack is carried out on IT Solution trustDesk standard 1.2.0 product. The attack does not need administrator privileges and relies on design flaws, not implementation ones. In particular, the attacker installs a modified driver library file. Upon accessing the card reader trustDesk loads and executes the modified device driver, giving access to the software's address space and permitting arbitrary malicious actions, like DTBSR modification.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.4: DTBSR modification
Target(s):	D3-CAT2: Software → D3-CAT2.2: Driver → D3-CAT2.2.3: SSCDev driver
Countermeasures:	Make the signing software to verify the signed code of every module used

Name:	Secure viewer compromise (3)
Source:	Malware Attacks on Electronic Signatures Revisited (Langweg, 2006)
Description:	The attack is carried out on D-Sign matrix/digiSeal 3.0.1 product. The attack does not need administrator privileges and relies on design flaws, not implementation ones. In particular, the attack modifies the viewer's presentation surface without detection to deceive the user respecting the data to be signed, while the information to be sent for signing differs.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.2: Content modification
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.2: SCA
Countermeasures:	Fix data before it becomes obvious to an attacker that the data is relevant for signing

Name:	PCS/SC card reader communication potential compromise (3)
Source:	Malware Attacks on Electronic Signatures Revisited (Langweg, 2006)
Description:	The attack is carried out on D-Sign matrix/digiSeal 3.0.1 product. The attack does not need administrator privileges and relies on design flaws, not implementation ones. As secure PIN entry is not the default

	option, the attacker can change the reader configuration and specify a new card terminal driver. Thereby, it is possible to load arbitrary malicious code in the software's address space, and perform the execution of malicious actions, like DTBSR modification.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.4: DTBSR modification
Target(s):	D3-CAT2: Software → D3-CAT2.2: Driver → D3-CAT2.2.3: SSCDev driver
Countermeasures:	Make the signing software to verify the signed code of every module used

Name:	Manipulated presentation of data to be signed
Source:	Malware Attacks on Electronic Signatures Revisited (Langweg, 2006)
Description:	The attack is carried out on Ventasoft venta-sign 2.0.0.968 product. The attack does not need administrator privileges and relies on design flaws, not implementation ones. The product does not provide a secure viewer. The attack draws on the application's presentation surface, showing the user a different file name and file information, while the information to be sent for signing differs.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.2: Content modification
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.2: SCA
Countermeasures:	Fix data before it becomes obvious to an attacker that the data is relevant for signing

Name:	Secure viewer compromise (4)
Source:	Malware Attacks on Electronic Signatures Revisited (Langweg, 2006)
Description:	The attack is carried out on 2B Secure FILE 1.0 product. The attack does not need administrator privileges and relies on design flaws, not implementation ones. In particular, the attack modifies the secure viewer's presentation surface without detection to deceive the user respecting the data to be signed (very similar to Secure viewer compromise (1) attack).
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.2: Content modification
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.2: SCA
Countermeasures:	Fix data before it becomes obvious to an attacker that the data is relevant for signing

Name:	Secure viewer compromise (5)
Source:	Malware Attacks on Electronic Signatures Revisited (Langweg, 2006)
Description:	The attack is carried out on Ultimaco SafeGuard Sign & Crypt for Office 3.4.1 product. The attack does not need administrator privileges and relies on design flaws, not implementation ones. In particular, the attack modifies the secure viewer's presentation surface without detection to deceive the user respecting the data to be signed, while the information to be sent for signing differs.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.2: Content modification
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.2: SCA
Countermeasures:	Fix data before it becomes obvious to an attacker that the data is relevant for signing

Name:	False positives in XML
Source:	What You See is Not Always What You Sign (Jsang et al., 2002)
Description:	The attack consists in modifying external parts of the signed XML document (e.g. a referenced schema or DTD). In particular, the attack shown modifies the ATTLIST of the DTD. While the syntactic form remains the same, the semantic varies.
Goal:	D1-CAT3: Replace signed information
Method:	D2-CAT3: Modification post signature computation → D2-CAT3.1: External content
Target(s):	D3-CAT5: Information → D3-CAT5.1: Document
Countermeasures:	Application of canonicalization algorithms. Addition of all involved content, including referenced external content, in the DTBS

Name:	Font type manipulation – Fonts substitution
Source:	What You See is Not Always What You Sign (Jsang et al., 2002)
Description:	An attacker can make a document have a different representation (semantic) by applying customized font types. If these font types are explicitly designed by the attacker for the document processor of the signer, and thus are not available during the verification stage, the glyph of certain characters can vary, changing the meaning of the document while maintaining the integrity of the signature. Though the authors presented this attack from the viewpoint of deceiving the verifier, this attack can be applied to deceive the signer, as described herein.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one

Method:	D2-CAT1: Environment manipulation
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Use of formats (e.g. PDF) that include the fonts definitions inside the content of the document

Name:	Inconsistent handling of HTML table tags
Source:	What You See is Not Always What You Sign (Jsang et al., 2002)
Description:	Web browsers interpret HTML and Javascript code in a different manner. Consequently, the same HTML code can be shown in different ways depending on the web browser used.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.1: Dynamic content inclusion → D2-CAT2.1.1.2: Active code
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.1: External application → D3-CAT2.1.1.2: User level application
Countermeasures:	Avoid the inclusion of dynamic content in the document to be signed

Name:	Substitution of Office document by external content using macros
Source:	Electronic Documents and Digital Signatures (Kain, 2003)
Description:	When opening the signed document, some active code (e.g. a macro programmed in Visual Basic for Applications for a Word document or an Excel spreadsheet) included in it substitutes the content of the document by an external content controlled by the attacker. This attack is feasible on Microsoft Office formats. As the signature is verified against the initial object, the signature integrity is not corrupted.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.1: Dynamic content inclusion → D2-CAT2.1.1.2: Active code
Target(s):	D3-CAT5: Information → D3-CAT5.1: Document D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Avoid the inclusion of dynamic content in the document to be signed

Name:	Substitution of Office document by external content referenced by links
Source:	Electronic Documents and Digital Signatures (Kain, 2003)
Description:	Office documents allow users to insert material from remote documents by reference. As a result, the document only manages a link to an external object, which is loaded on demand. This characteristic permits an attacker to manipulate the linked data without corrupting the signature integrity.
Goal:	D1-CAT3: Replace signed information
Method:	D2-CAT3: Modification post signature computation → D2-CAT3.1: External content
Target(s):	D3-CAT5: Information → D3-CAT5.1: Document D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Avoid the inclusion of links to external content in the document to be signed

Name:	External queries in Excel
Source:	Electronic Documents and Digital Signatures (Kain, 2003)
Description:	Excel includes features to make explicit queries to remote files. The attacker can select an option to get external data and set up a query to a remote text file. The text file should be written with tab spaces between words to specify different fields in the spreadsheet. By right-clicking on the cell and selecting Data Range Properties, the attacker can configure the query to update on open or even regularly (in the background).
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.1: Dynamic content inclusion → D2-CAT2.1.1.3: Linked content
Target(s):	D3-CAT5: Information → D3-CAT5.1: Document D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Avoid the inclusion of dynamic content in the document to be signed

Name:	Substitution of Office document content by means of fields
Source:	Electronic Documents and Digital Signatures (Kain, 2003)
Description:	Several attacks can be performed using the field feature in some Office formats, like Word or Excel. Fields like TIME, USERNAME, etc. can make the visualization of a document content vary according to conditions controlled by the attacker. For instance, depending on the date when a document is opened or the user that opens the document, a piece of text can take one of several different possibilities. The content dependent on a field can be updated automatically in certain versions of Microsoft Word or explicitly via a macro.

Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.1: Dynamic content inclusion → D2-CAT2.1.1.1: Hidden code
Target(s):	D3-CAT5: Information → D3-CAT5.1: Document D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Avoid the inclusion of dynamic content in the document to be signed

Name:	Substitution of PDF content by means of javascript
Source:	Electronic Documents and Digital Signatures (Kain, 2003)
Description:	The attacker can use the form toolbar to create a form field, and then add Javascript code in its calculate field to change the value of the field according to the date.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.1: Dynamic content inclusion → D2-CAT2.1.1.2: Active code
Target(s):	D3-CAT5: Information → D3-CAT5.1: Document D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Avoid the inclusion of dynamic content in the document to be signed

Name:	Modification of HTML email content via Javascript
Source:	Electronic Documents and Digital Signatures (Kain, 2003)
Description:	An attack that modifies the content of an email formatted as HTML is performed by using the *document.write()* Javascript function and the current date.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.1: Dynamic content inclusion → D2-CAT2.1.1.2: Active code
Target(s):	D3-CAT5: Information → D3-CAT5.1: Document D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Avoid the inclusion of dynamic content in the document to be signed

Name:	Modification of HTML email content via embedded image
Source:	Electronic Documents and Digital Signatures (Kain, 2003)
Description:	The attacker embeds an image in an HTML formatted email and, in conjunction with Javascript, is able to modify the visualized content of the signed email.
Goal:	D1-CAT1: Deceive the signer to sign a document different to the intended one
Method:	D2-CAT2: Modification prior to signature computation → D2-CAT2.1: Document modification → D2-CAT2.1.1: Dynamic content inclusion → D2-CAT2.1.1.2: Active code
Target(s):	D3-CAT5: Information → D3-CAT5.1: Document D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Avoid the inclusion of dynamic content in the document to be signed

Name:	Signature creation data retrieval from low-security keys
Source:	Keyjacking: the surprising insecurity of client-side SSL (Marchesini et al., 2005)
Description:	Internet Explorer Web browser relies on Windows keystore and Cryptographic Service Provider (CSP) to store the private keys imported therein. Microsoft's CSP publishes a function called *CryptExportKey* which permits to directly obtain the private key from a keystore. A low-security key, which is the configuration by default, is a key imported in Internet Explorer which is not password-protected. Consequently, and based on previous facts, an attacker that gains access to the user's account or is able to execute malicious code with the user's privileges will the able to access the private key. The attacker could even export the private key for further usages.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.3: Unauthorized access to the SCDev → D2-CAT5.3.2: Authentication Bypass
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.4: SCDev
Countermeasures:	Use stronger configuration settings

Name:	Use of low-security keys
Source:	Keyjacking: the surprising insecurity of client-side SSL (Marchesini et al., 2005)
Description:	This attack is based on the same motivation as the attack *Signature creation data retrieval from low-security keys*. However, in this case the attacker does not retrieve the signature creation data but just performs as

	many signatures as desired without the user consent and knowledge. This attack is an alternative if the key was set to non-exportable.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT4: Unauthorized invocation of the signing function → D2-CAT4.2: Authentication Bypass
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.4: SCDev
Countermeasures:	Use stronger configuration settings

Name:	Signature creation data retrieval from exportable medium-security keys
Source:	Keyjacking: the surprising insecurity of client-side SSL (Marchesini et al., 2005)
Description:	When a medium-security key is to be accessed (for signing or export), a warning is shown to the user, who must confirm the operation. This attack captures the warning event, hiding it to the user, during the key export operation (the key must be set as exportable). To achieve that, the attacker performs an API hijacking in which a function call made by the Internet Explorer process to the system Windows CryptoAPI is intercepted by a malicious DLL previously injected via a Windows Hook. Thereby, the attacker is able to hijack the call which displays the warning window, disabling it.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.3: Unauthorized access to the SCDev → D2-CAT5.3.2: Authentication Bypass
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.4: SCDev
Countermeasures:	–

Name:	Use of medium-security keys
Source:	Keyjacking: the surprising insecurity of client-side SSL (Marchesini et al., 2005)
Description:	This attack applies the same strategy as the attack *Signature creation data retrieval from exportable medium-security keys*. However, in this case the attacker does not retrieve the signature creation data but just performs as many signatures as desired without the user consent and knowledge. This attack is an alternative if the key was set to non-exportable.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT4: Unauthorized invocation of the signing function → D2-CAT4.2: Authentication Bypass
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.4: SCDev
Countermeasures:	–

Name:	Signature creation data retrieval from high-security keys (in unrecommended configuration)
Source:	Keyjacking: the surprising insecurity of client-side SSL (Marchesini et al., 2005)
Description:	A high-security key requires the user to enter the associated password (SAD) each time the key is to be used or exported. However, if the user checked the box marked "Remember password", the level of the key is downgraded to low-security, enabling the attacker to perform the same attack as in *Signature creation data retrieval from low-security keys*.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.3: Unauthorized access to the SCDev → D2-CAT5.3.2: Authentication Bypass
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.4: SCDev
Countermeasures:	Do not select "remember password" in the configuration settings

Name:	Use of high-security keys (in unrecommended configuration)
Source:	Keyjacking: the surprising insecurity of client-side SSL (Marchesini et al., 2005)
Description:	The attacker makes use of the same highly unrecommended configuration as in the attack *Signature creation data retrieval from high-security keys (in unrecommended configuration)*, being able to perform the same attack as in *Use of low-security keys*. This attack is an alternative if the key was set to non-exportable.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT4: Unauthorized invocation of the signing function → D2-CAT4.2: Authentication Bypass
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.4: SCDev
Countermeasures:	Do not select "remember password" in the configuration settings

Name:	Signature creation data retrieval from exportable high-security keys
Source:	Keyjacking: the surprising insecurity of client-side SSL (Marchesini et al., 2005)
Description:	The attack is based on the same strategy as in *Signature creation data retrieval from exportable medium-security keys*. In this case, the attacker captures the invocation to the function that shows a window asking for

	a password each time the key is to be used. Once obtained the first time, the attacker is able retrieve the private key for further usages.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.3: Unauthorized access to the SCDev → Compromise of the Signer Authentication Data (SAD) → D2-CAT4.1.2: SAD interception → D2-CAT4.1.2.2: Interception in interprocess/entities communication
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.3: CSP
Countermeasures:	The authors indicate that there is no countermeasure for this security issue

Name:	Use of high-security keys
Source:	Keyjacking: the surprising insecurity of client-side SSL (Marchesini et al., 2005)
Description:	In case the key is set as non-exportable, the attacker can following the same actions as in *Signature creation data retrieval from exportable high-security keys* to compromise the access password.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT4: Unauthorized invocation of the signing function → D2-CAT4.1: Compromise of the Signer Authentication Data (SAD) → D2-CAT4.1.2: SAD interception → D2-CAT4.1.2.2: Interception in interprocess/entities communication
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.3: CSP
Countermeasures:	The authors indicate that there is no countermeasure for this security issue

Name:	Use of high-security keys during the same session
Source:	Keyjacking: the surprising insecurity of client-side SSL (Marchesini et al., 2005)
Description:	This attack relies on a vulnerability by design in the CryptoAPI. In the context of Internet Explorer Web browser, once the CryptoAPI has authenticated a user when accessing a high-security key, subsequent accesses fail to request for the password. Using a malicious code that makes the same sequence of calls to the CryptoAPI as Internet Explorer, the attacker can perform as many signing operations as desired once the password has been provided by the user, and providing that the browser is not restarted.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT4: Unauthorized invocation of the signing function → D2-CAT4.2: Authentication Bypass
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.4: SCDev
Countermeasures:	Close the Web browser once the desired operation is performed. Clear the SSL State in the Web browser configuration

Name:	Use of keys stored in cryptographic tokens
Source:	Keyjacking: the surprising insecurity of client-side SSL (Marchesini et al., 2005)
Description:	The attack applies the same strategy as in *Use of high-security keys* for keys stored in a particular external cryptographic token.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT4: Unauthorized invocation of the signing function → D2-CAT4.1: Compromise of the Signer Authentication Data (SAD) → D2-CAT4.1.2: SAD interception → D2-CAT4.1.2.2: Interception in interprocess/entities communication
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.3: CSP
Countermeasures:	–

Name:	Deception to use keys stored on cryptographic tokens
Source:	Keyjacking: the surprising insecurity of client-side SSL (Marchesini et al., 2005)
Description:	This attack makes use of social behavior to perform signatures on behalf of the user without their consent and knowledge. When a cryptographic token such as the Spanish electronic Identity Card (eDNI), Spyrus Rosetta USB and many others requests to user to insert the PIN or password in every access to the private key or protected areas of the internal file system, the user gets used to insert the credentials several times for a single operation (i.e. authenticate in a Web site, sign a document, etc.). The attacker will just request the user to enter the SAD in the middle of a normal operation or unexpectedly, and there will be a non-negligible probability for the user to do that.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.3: Unauthorized access to the SCDev → D2-CAT5.3.1: Compromise of the Signer Authentication Data (SAD) → D2-CAT4.1.1: Social engineering
Target(s):	D3-CAT4: Human user → D3-CAT4.1: Signer
Countermeasures:	Apply a different design where the SAD is not required so many times. Caching the SAD during a single operation may lead to the attack *Using cached SAD to perform malicious signatures*

Name:	Using cached SAD to use keys stored on cryptographic tokens
Source:	Keyjacking: the surprising insecurity of client-side SSL (Marchesini et al., 2005)

Description:	In this attack, the attacker can perform as many signatures as desired if the CSP of a cryptographic token is configured to use the key for a specified time interval without asking for permission.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT4: Unauthorized invocation of the signing function → D2-CAT4.2: Authentication Bypass
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.3: CSP
Countermeasures:	Applying a different design where the SAD is required for every single access to the key may lead to the attack *Deception to use keys stored on cryptographic tokens*

Name:	Signature creation data retrieval from password-protected files
Source:	BreakMS – Break Microsoft Private Key Encryption with a dictionary attack (Gutmann, 1997, 1998)
Description:	This attack exploits several design and implementation vulnerabilities found in PKCS12/PFX file format to perform a low-cost dictionary attack to discover the password used to protect the file and further retrieve the private key.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.3: Unauthorized access to the SCDev → D2-CAT5.3.1: Compromise of the Signer Authentication Data (SAD) → D2-CAT4.1.3: Guessing
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.4: SCDev
Countermeasures:	Redesign and careful implementation of PKCS12/PFX format

Name:	Unauthorized usage of platform resources by a malicious Applet in a Java-enabled card
Source:	Software attacks on smart cards (Girard and Giraud, 2003)
Description:	If the Java Card where the Applet is loaded does not implement an access controller, then a Trojan horse embedded in the Applet can perform malicious operations. If there is no domain separation between simultaneous applets, the malicious one could extract sensitive information managed by another, like the PIN code, or modify critical data like the number of authentication attempts. These attacks could later derive in signature forgeries.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT4: Unauthorized invocation of the signing function → D2-CAT4.1: Compromise of the Signer Authentication Data (SAD) → D2-CAT4.1.2: SAD interception → D2-CAT4.1.2.2: Interception in interprocess/entities communication
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.1: External application → D3-CAT2.1.1.2: User level application
Countermeasures:	Correct design and implementation of Java Card, specially Java Virtual Machine. Correct design and implementation of applets. Use of access controller. Use of shareable interfaces between applets (domain separation enforcement). More tips can be found in Girard and Giraud (2003)

Name:	PIN phishing and Fraudulent signatures
Source:	Vulnerabilities of PKI based Smartcards (Dasgupta et al., 2007)
Description:	The attacker reads the signer's authentication data (PIN of the smart card) entered by the user in the keyboard by means of a keylogger. Once the attacker has compromised the SAD, it is able to access the signing function of a smart card without the user knowing.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT4: Unauthorized invocation of the signing function → D2-CAT4.1: Compromise of the Signer Authentication Data (SAD) → D2-CAT4.1.2: SAD interception → D2-CAT4.1.2.2: Interception in interprocess/entities communication
Target(s):	D3-CAT2: Software → D3-CAT2.2: Driver → D3-CAT2.2.1: Keyboard driver
Countermeasures:	Use of secure I/O between the user and the Java Card: PIN entry from a cellular phone; separate hardware channel between the PKI card and a special I/O device that handles the user inputs; match-on-cards with own display

Name:	Remote control of PKI Card
Source:	Vulnerabilities of PKI based Smartcards (Dasgupta et al., 2007)
Description:	An attacker is able to remotely request signing operations on the smart card once the user has unlocked it.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT4: Unauthorized invocation of the signing function → D2-CAT4.2: Authentication Bypass
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.3: CSP
Countermeasures:	–

Name:	Improved Timing Analysis attack on RSA
Source:	Improving Timing Attack on RSA-CRT via Error Detection and Correction Strategy (Chen et al., 2012)
Description:	The algorithm proposed in this paper achieves a practical timing attack with better performance than previous proposals. The algorithm includes an error detection mechanism and correction strategy that can detect and correct the erroneous decision of guessing qk. With an improvement timing attack on the RSA

	algorithm in OpenSSL, the 0–1 gap is enlarged, the neighborhood size is reduced, and the precision of the decision is improved. Moreover, obtaining the factor q is practical, and even recovers a 1024-bit RSA key completely for an interprocess timing attack.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.1: Timing Analysis
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.4: SCDev
Countermeasures:	–

Name:	Timing Analysis attack in controlled environments
Source:	Timing attacks on Implementations of Diffie–Hellman, RSA, DSS and Other Systems (Kocher, 1996)
Description:	This was the first designed timing attack, which implementations were successful against Diffie–Hellman, RSA and DSS cryptosystems. These attacks were carried out in an isolated computing environment where the measured time could not be masked by delays provoked by processes running in the background.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.1: Timing Analysis
Target(s):	D3-CAT3: Hardware → D3-CAT3.1: SSCDev
Countermeasures:	Adapted blinding signatures can prevent attackers from knowing the input to the modular exponentiation function, with only low performance decrease (Kocher, 1996)

Name:	Timing Analysis attack using the Chinese Remainder Theorem
Source:	A Timing Attack against RSA with the Chinese Remainder Theorem (Schindler, 2000)
Description:	In this attack, an RSA-modulus is factorized providing that the exponentiation with the secret exponent uses the Chinese Remainder Theorem and Montgomery's algorithm.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.1: Timing Analysis
Target(s):	D3-CAT3: Hardware → D3-CAT3.1: SSCDev
Countermeasures:	Kocher, 1996

Name:	Remote Timing Analysis attack
Source:	Remote Timing Attacks are Practical (Brumley and Boneh, 2003)
Description:	Brumley and Boneh showed that remote attacks on real applications over a local network and running in general software systems are possible. In this case, they devised a timing attack against OpenSSL, guessing the private key used by the Web server for authenticating itself during the SSL handshake stage. This has been quite an important research since timing attacks are now possible although noisy intermediate elements such as network routers and background processes interact during the attack.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.1: Timing Analysis
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.4: SCDev
Countermeasures:	Enable the blinding feature of OpenSSL

Name:	Improved Remote Timing Analysis attack
Source:	Improving Brumley and Boneh Timing Attack on Unprotected SSL Implementations (Aciimez et al., 2005)
Description:	The authors improve the *Remote Timing Analysis attack* efficiency by a factor of more than ten. In particular, the attack exploit the timing behavior of Montgomery multiplications in the table initialization phase, which increases the number of multiplications that provide useful information to reveal one of the prime factors of RSA moduli.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.1: Timing Analysis
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.4: SCDev
Countermeasures:	Enable the blinding feature of OpenSSL

Name:	Simple Power Analysis attack (SPA)
Source:	Differential Power Analysis (Kocher et al., 1999)
Description:	This type of power analysis attack is imperceptible to the user and can be successfully performed by using simple and cheap equipments. It only needs one or few measurements of power consumption signals to retrieve the private key stored in the cryptographic device.

Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.3: Power Analysis
Target(s):	D3-CAT3: Hardware → D3-CAT3.1: SSCDev
Countermeasures:	Make the power consumption of the cryptographic device independent of the signal values at the internal circuit nodes by either randomizing or flattening the power consumption. However, these techniques do not assure the device to be completely secure against these attacks, and instead they increase the required number of measurements (Tiri, 2007). If the attacker has access to the device for performing an enough number of operations, these countermeasures are useless.

Name:	Mono-bit Differential Power Analysis attack (DPA)
Source:	Differential Power Analysis (Kocher et al., 1999) and An overview of side channel analysis attacks (Le et al., 2008)
Description:	This type of power analysis attack is a statistical approach that examines a large number of power consumptions signals to retrieve secret keys. In particular, the mono-bit DPA analyzes the intermediate values of one bit.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.3: Power Analysis
Target(s):	D3-CAT3: Hardware → D3-CAT3.1: SSCDev
Countermeasures:	Tiri, 2007

Name:	Multi-bit Differential Power Analysis attack (DPA)
Source:	Ways to Enhance DPA (Bevan and Knudsen, 2003) and An overview of side channel analysis attacks (Le et al., 2008)
Description:	The difference between this attack and *Mono-bit Differential Power Analysis attack (DPA)* is that the former analyzes intermediate values of a set of several bits.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.3: Power Analysis
Target(s):	D3-CAT3: Hardware → D3-CAT3.1: SSCDev
Countermeasures:	Tiri, 2007

Name:	First-order Differential Power Analysis attack (DPA)
Source:	Differential Power Analysis (Kocher et al., 1999)
Description:	In this case, the samples are observed at one instant of time.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.3: Power Analysis
Target(s):	D3-CAT3: Hardware → D3-CAT3.1: SSCDev
Countermeasures:	Tiri, 2007

Name:	High-order Differential Power Analysis attack (DPA)
Source:	On Second-Order Differential Power Analysis (Joye et al., 2005)
Description:	Contrary to *First-order Differential Power Analysis attack (DPA)*, this type of DPA attack analyzes the power consumption signals at some instants of time.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.3: Power Analysis
Target(s):	D3-CAT3: Hardware → D3-CAT3.1: SSCDev
Countermeasures:	Tiri, 2007

Name:	Correlation Power Analysis attack (CPA)
Source:	A proposition for Correlation Power Analysis enhancement (Le et al., 2006) and An overview of side channel analysis attacks (Le et al., 2008)
Description:	This type of attack consists of a technique based on the correlation between the real power consumption of the device and a certain power consumption model. DPA and CPA are based on power consumption models, so their efficiency completely depends on the chosen model. In case of wrongly modeling the power consumption, the key obtaining is impossible. Besides, these attacks need a large number of samples, and hence are not very practical.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.3: Power Analysis
Target(s):	D3-CAT3: Hardware → D3-CAT3.1: SSCDev
Countermeasures:	Tiri, 2007

Name:	Template Power Analysis attack
Source:	IPA: A New Class of Power Attacks (Fahn and Pearson, 1999), Template Attacks (Chari et al., 2002) and An overview of side channel analysis attacks (Le et al., 2008)
Description:	This type of attack needs a reference device for executing a profiling stage. In this stage, a large number of signals are obtained from the reference device in order to learn how it works. During the second stage, the key extraction stage, the key is obtained by analyzing very few signals from the attacked device, improving the applicability of the attack respecting other types of power analysis attacks, like DPA or CPA. The reference device must be identical or very closed to the attacked device for the attack to work.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.3: Power Analysis
Target(s):	D3-CAT3: Hardware → D3-CAT3.1: SSCDev
Countermeasures:	Tiri, 2007

Name:	Stochastic Power Analysis attack
Source:	A Stochastic Model for Differential Side Channel Cryptanalysis (Schindler et al., 2005) and An overview of side channel analysis attacks (Le et al., 2008)
Description:	This attack needs a reference device like the *Template Power Analysis attack*. This attack uses a different strategy than the template-based attack. For instance, during the profiling stage, the power consumption is estimated by predefined functions, not from actual measured signals.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.3: Power Analysis
Target(s):	D3-CAT3: Hardware → D3-CAT3.1: SSCDev
Countermeasures:	Tiri, 2007

Name:	Electromagnetic Emanation attack on RSA
Source:	ElectroMagnetic Analysis (EMA): Measures and Counter-measures for Smart Cards (Quisquater and Samyde, 2001) and Electromagnetic Analysis: Concrete Results (Gandolfi et al., 2001)
Description:	An attack to an RSA implementation was successfully carried out, focusing on the RSA modular exponentiation performed in a decapsulated smart card.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.2: Electromagnetic Analysis
Target(s):	D3-CAT3: Hardware → D3-CAT3.1: SSCDev
Countermeasures:	Hardware countermeasures: metal layer addition to the chip; active grid placement on top of the chip, in order to introduce more noise into the EM field, blurring the emanations (Matthews, 2006)

Name:	Electromagnetic Emanation attack by using the channel capacity information
Source:	Evaluation of Information Leakage via Electromagnetic Emanation and Effectiveness of Tempest (Tanaka, 2008)
Description:	In this study, it is shown how to estimate the amount of information leakage by using the value of channel capacity, that it, the communication channel between the measured IT device and the receiver. This IT device can be both a personal computer or a smart card.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.2: Electromagnetic Analysis
Target(s):	D3-CAT3: Hardware → D3-CAT3.1: SSCDev D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.4: SCDev
Countermeasures:	Tiri, 2007

Name:	A low cost Electronic Emanation attack on a smart card
Source:	Low cost attacks on smart cards: The electromagnetic side-channel (Matthews, 2006)
Description:	With this attack, it is demonstrated that performing EMA attacks using limited technical knowledge as well as cheap resources is possible. EM traces are successfully acquired from the sample card, and an analysis software correctly identifies the key.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.2: Electromagnetic Analysis
Target(s):	D3-CAT3: Hardware → D3-CAT3.1: SSCDev
Countermeasures:	Tiri, 2007

Name:	Fault-based attack on RSA
Source:	Fault-Based Attack of RSA Authentication (Pellegrini et al., 2010)
Description:	In this paper, a theoretical systematic fault-based attack on the modular exponentiation algorithm for RSA is developed. Later on, the authors carry out a practical and complete end-to-end fault-attack on a microprocessor system, exploiting the vulnerabilities of an FPGA implementation of the system under attack and which runs a flawed OpenSSL software implementation. The authors inject transient faults in the target machine by regulating the voltage supply of the system, not requiring access to the system's internal components but just proximity to it. The authors are able to extract the 1024-bit RSA private key in approximately 100 h.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.6: Fault Injection
Target(s):	D3-CAT3: Hardware → D3-CAT3.1: SSCDev D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.4: SCDev
Countermeasures:	–

Name:	Fault Attack against ECDSA
Source:	A Novel Fault Attack Against ECDSA (Barenghi et al., 2011)
Description:	A novel fault attack against ECDSA is proposed, and by which the secret signing key (of any length) can be retrieved by means of injecting faults during the computation of the signature primitive. The proposed method relies on faults injected during a multiplication employed to perform the signature recombination at the end of the ECDSA signing algorithm.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.6: Fault Injection
Target(s):	D3-CAT3: Hardware → D3-CAT3.1: SSCDev D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.4: SCDev
Countermeasures:	–

Name:	A Branch Prediction Analysis attack on RSA: Exploiting the Predictor directly (Direct Timing Attack)
Source:	Predicting Secret Keys via Branch Prediction (Aciimez et al., 2007a)
Description:	This is a type of microarchitectural side-channel attack called Branch Prediction Analysis (BPA) attack, by which the branch prediction capability, common to all modern high-performance CPUs, is exploited to know the private key used in a software cryptographic algorithm. In particular, the penalty payed (extra clock cycles) for a mispredicted branch can be used for cryptanalysis of cryptographic primitives that employ a data-dependent program flow. This attack relies on the fact that the prediction algorithms are deterministic, and assume that the RSA implementation employs Square-and-Multiply exponentiation and Montgomery Multiplication. Though this attack is experimentally carried out on a simple RSA implementation, the underlying ideas can be used to develop similar attacks on different implementations of RSA and/or on other ciphers based upon ECC.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.4: Microarchitectural Analysis
Target(s):	D3-CAT2: Software → D3-CAT2.3: Operating system
Countermeasures:	–

Name:	A Branch Prediction Analysis attack on RSA: Forcing the BPU to the Same Prediction (Asynchronous Attack)
Source:	Predicting Secret Keys via Branch Prediction (Aciimez et al., 2007a)
Description:	In this attack it is assumed that the cipher runs on a simultaneous multi-threading computer. The attacker can run a dummy process simultaneously with the cipher process, but the two parallel threads are isolated and share only the common Branch Prediction Unit (BPU) resource. Also, the attacker does not need to know any detail of the prediction algorithm., like in the previous attack.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.4: Microarchitectural Analysis
Target(s):	D3-CAT2: Software → D3-CAT2.3: Operating system
Countermeasures:	–

Name:	A Branch Prediction Analysis attack on RSA: Forcing the BPU to the Same Prediction (Synchronous Attack)
Source:	Predicting Secret Keys via Branch Prediction (Aciimez et al., 2007a)
Description:	In this attack, the malicious process needs some sort of synchronization with the simultaneous crypto-process. It is also assumed that the RSA implementation employs Square-and-Multiply exponentiation.

	Any implementation of a cryptosystem is vulnerable to this kind of attack if the execution flow is key-dependent, including several implementations that had been considered to be immune to certain types of side-channel attacks.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.4: Microarchitectural Analysis
Target(s):	D3-CAT2: Software → D3-CAT2.3: Operating system
Countermeasures:	–

Name:	A Branch Prediction Analysis attack on RSA: Trace-driven Attack against the BTB (Asynchronous Attack)
Source:	Predicting Secret Keys via Branch Prediction (Aciimez et al., 2007a)
Description:	In this attack, it is assumed that the attacker can run a spy process simultaneously with the cipher, but it does not need to be synchronized with it. The same cryptographic implementations vulnerable to the previous attack are vulnerable to this one. Furthermore, this attack is much easier to be put in practice, and, in the authors' opinion, this attack puts many of the current public-key implementations in danger.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.4: Microarchitectural Analysis
Target(s):	D3-CAT2: Software → D3-CAT2.3: Operating system
Countermeasures:	–

Name:	A Simple Branch Prediction Analysis attack on RSA
Source:	On the Power of Simple Branch Prediction Analysis (Aciimez et al., 2007b)
Description:	This is a BPA variation by which almost all of the RSA key bits can be extracted during a single RSA operation.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.4: Microarchitectural Analysis
Target(s):	D3-CAT2: Software → D3-CAT2.3: Operating system
Countermeasures:	–

Name:	An Instruction Cache Analysis attack on the RSA implementation of OpenSSL
Source:	Yet another MicroArchitectural Attack: Exploiting I-cache (Aciimez, 2005)
Description:	This attack exploits the behavior of the Instruction Cache – which is used to reduce the average time to read instruction codes from main memory – to extract sensitive information regarding the execution of a cryptosystem. More specifically, this attack targets the OpenSSL sliding Window exponentiation of its RSA implementation.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.2: Eavesdropping (side-channel) → D2-CAT5.2.4: Microarchitectural Analysis
Target(s):	D3-CAT2: Software → D3-CAT2.3: Operating system
Countermeasures:	–

Name:	PIN/Password recovering from keyboard acoustic emanations
Source:	Keyboard Acoustic Emanations Revisited (Zhuang et al., 2005)
Description:	The authors built a prototype that can bootstrap a keyboard acoustic recognizer from about 10 min of English text typing, using about 30 min of computation on an average desktop computer. After that, the prototype can recognize keystrokes in real time, including random ones such as passwords, with an accuracy rate of about 90%. The keystrokes must be typed by the same person, with the same keyboard, under the same recording conditions. These conditions can easily be satisfied by, for example, placing a wireless microphone in the user's work area or by using parabolic microphones. This attack could be mounted to compromise the signer's authentication data. As a result, it would be the earliest stage before accessing the signature creation data or the signing function. As such, this partial attack can be classified under the two methods indicated below.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT4: Unauthorized invocation of the signing function → D2-CAT4.1: Compromise of the Signer Authentication Data (SAD) → D2-CAT4.1.3: Guessing D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.3: Unauthorized access to the SCDev → D2-CAT5.3.1: Compromise of the Signer Authentication Data (SAD) → D2-CAT4.1.3: Guessing
Target(s):	D3-CAT3: Hardware → D3-CAT3.2: Computer → D3-CAT3.2.4: Peripheral devices → D3-CAT3.2.4.2: Keyboard
Countermeasures:	Ensure the physical security of the machine and the room. Use of two-factor authentication (e.g. password and biometrics) to access the signature creation data

Name:	Chosen-prefix MD5 Collisions
Source:	Chosen-prefix Collisions for MD5 and Applications (Stevens et al., 2012)
Description:	An attack to find differential paths for MD5 is presented. Its main application is in the construction of chosen-prefix collisions, showing that at an approximate expected cost of 2^{39} calls to the MD5 compression function, for any two chosen message prefixes P and P′, sufixes S and S′ can be constructed such that the concatenated values P–S and P′–S′ collide under MD5. A particular example of the attack on colliding documents is presented. Using a document format that allows insertion of color images, inserting one message per document, two documents can be made to collide by appending carefully crafted color images after the messages. A short one pixel wide line will do – for instance hidden inside a layout element, a company logo, or a nicely colored barcode – and preferably scaled down to hardly visible size (or completely hidden from view, as possible in PDF).
Goal:	D1-CAT3: Replace signed information
Method:	D2-CAT3: Modification post signature computation → D2-CAT3.2: Cryptanalysis → D2-CAT3.2.1: Hash function → D2-CAT3.2.1.1: Collision attack
Target(s):	D3-CAT1: Cryptography
Countermeasures:	Use stronger hash functions

Name:	Finding collisions in several MD3,MD5, HAVAL, RIPEMD and SHA-0
Source:	How to Break MD5 and Other Hash Functions (Wang and Yu, 2005)
Description:	A new differential attack on several hash functions is described. The attack, called modular differential, unlike most differential attacks, uses modular integer subtraction as the measure instead of the exclusive-or. In the case of MD3, the attack can find a collision within less than a second, and can also find second preimages for many messages. For MD5, it finds collisions in about 15 min up to an hour computation time. As the attack can be carried out following two different methods (collision or second preimage), the method of the attack could be classified attending to both approaches.
Goal:	D1-CAT3: Replace signed information
Method:	D2-CAT3: Modification post signature computation → D2-CAT3.2: Cryptanalysis → D2-CAT3.2.1: Hash function → D2-CAT3.2.1.1: Collision attack
Target(s):	D3-CAT1: Cryptography
Countermeasures:	Use stronger hash functions

Name:	Finding MD5 collisions using tunnels
Source:	Tunnels in hash functions: MD5 collisions within a minute (Klima, 2006)
Description:	The author proposes a new strategy to find collisions in hash functions named tunneling. Tunnels replace multi-message modification methods and exponentially accelerate collision search. In particular, the author describe several tunnels in hash function MD5. By using them, an MD5 collision is found in approximately 1 min on a standard notebook PC (Intel Pentium, 1.6 GHz). This attack is a collision attack, since it finds two messages which hash coincides. The method works for any initializing value. For this attack to succeed, the attacker must trick the user to sign one of the messages (possibly the message is aligned with the user's interests), and afterward replace it by the fraudulent one (see birthday attack (Coppersmith, 1985))
Goal:	D1-CAT3: Replace signed information
Method:	D2-CAT3: Modification post signature computation → D2-CAT3.2: Cryptanalysis → D2-CAT3.2.1: Hash function → D2-CAT3.2.1.1: Collision attack
Target(s):	D3-CAT1: Cryptography
Countermeasures:	Use stronger hash functions

Name:	Using Expandable Messages to Find Second Preimages
Source:	Second preimages on n-bit hash functions for much less than 2^n work (Kelsey and Schneier, 2005)
Description:	The authors describe a generic way to carry out long-message second preimage attacks, despite the Damgard–Merkle strengthening done on all modern hash functions (including SHA-1). The work required to achieve the attack is substantially lower than the reference one (2^n). For instance, using SHA-1 as an example, the attack can find a second preimage for a 2^{60} byte message in 2^{106} work, rather than the previously expected 2^{160} work. Though the attack is theoretical (e.g. the messages for which second preimages may be found are generally impractically long), the authors showed that an n-bit iterated hash function cannot provide the expected second-preimage resistance for long messages. As a second preimage attack, the attacker would be able to compose a malicious document which hash value matched the one of the signed document.
Goal:	D1-CAT3: Replace signed information
Method:	D2-CAT3: Modification post signature computation → D2-CAT3.2: Cryptanalysis → D2-CAT3.2.1: Hash function → D2-CAT3.2.1.3: Second preimage attack
Target(s):	D3-CAT1: Cryptography
Countermeasures:	Use stronger hash functions

Name:	Herding attack on hash functions
Source:	Herding Hash Functions and the Nostradamus Attack (Kelsey and Kohno, 2006)
Description:	The authors define a property of a hash function, Chosen Target Forced Prefix (CTFP) preimage resistance, which is both important for real-world applications of hash functions, and dependent on collision resistance of the hash function. More specifically, the described attack, called the herding attack, affects Damgard–Merkle hash functions in a way that the attacker who can find many collisions on the hash function by brute force can first provide the hash of a message, and later "herd" any given starting part of a message (P) to that hash value by the choice of an appropriate suffix (S). This attack can be considered a practical improvement of *Using Expandable Messages to Find Second Preimages* where the resulting message can be of a reasonable size. The authors provide concrete examples of carrying out the attack. One of them, named *Tweaking a Signed Document*, considers the case where a signer can later produce a modified message while still resulting in the same hash. As stated by the authors, many applications of hashing for signatures which are not vulnerable to attack by straightforward collision-finding techniques are broken by an attacker who can violate CTFP preimage resistance. When the CTFP definition is relaxed somewhat the attacks become still cheaper and more practical. For instance, if the attacker has control over the format of P – easy if the attacker intercepts the document to be signed, giving him prior knowledge of the full (large) set of possible P strings that might be presented (this is possible in certain transactions where the skeleton of the DTBS is fixed and just few parts of the document can vary). This is a preimage attack since the attacker manipulates part of the data entered in the hash function in order to obtain the desired hash value.
Goal:	D1-CAT3: Replace signed information
Method:	D2-CAT3: Modification post signature computation → D2-CAT3.2: Cryptanalysis → D2-CAT3.2.1: Hash function → D2-CAT3.2.1.2: Preimage attack
Target(s):	D3-CAT1: Cryptography
Countermeasures:	Use stronger hash functions

Name:	Preimage attack on RIPEMD
Source:	Preimage Attack on Hash Function RIPEMD (Wang and Wang, 2009)
Description:	The first preimage attack on the RIPEMD hash function is described. Three variants are shown: an attack on the compression function of the 26-step reduced RIPEMD, with complexity 2^{110} compression function computations; an attack on the 26-step reduced RIPEMD with complexity $2^{115.2}$ instead of 2^{128}; and an attack on 29 steps with the same complexity. Furthermore, the complexity of the preimage attack on the full RIPEMD without the padding rule is reduced to 2^{127}, which optimizes the complexity order to brute-force attack.
Goal:	D1-CAT3: Replace signed information
Method:	D2-CAT3: Modification post signature computation → D2-CAT3.2: Cryptanalysis → D2-CAT3.2.1: Hash function → D2-CAT3.2.1.2: Preimage attack
Target(s):	D3-CAT1: Cryptography
Countermeasures:	Use stronger hash functions

Name:	Parallel RSA factorization using the Multiple Polynomial Quadratic Sieve (MPQS)
Source:	A Study on Parallel RSA Factorization (Yeh et al., 2009)
Description:	In this paper, a factorization of a 100-digit RSA modulus into the former primer numbers is presented. The experimental result shows that it takes 6.6 days for factoring the 100-digit number using the enhanced MPQS by 32 workstations.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.4: Cryptanalysis → D2-CAT5.4.1: Asymmetric algorithm
Target(s):	D3-CAT1: Cryptography
Countermeasures:	Use large RSA key lengths (currently recommended 1024 bits and above)

Name:	Integer factorization with TWINKLE
Source:	Analysis and optimization of the TWINKLE factoring Device (Lenstra and Shamir, 2000)
Description:	TWINKLE (The Weizmann Institute Key Locating Engine) is an optoelectronic device designed to be capable of factoring large integers by speeding up the sieving step of the Quadratic Sieve and Number Field Sieve factoring algorithms. The authors consider that a TWINKLE-assisted factorization of a 768-bit number is feasible in about 9 months using a set of 80,000 standard Pentium II PC's and 5000 TWINKLE devices. The advances in computers since 2000 let us foresee that the time needed to factoring large numbers would imply a bound lower than 9 months.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.4: Cryptanalysis → D2-CAT5.4.1: Asymmetric algorithm
Target(s):	D3-CAT1: Cryptography
Countermeasures:	Use large RSA key lengths (currently recommended 1024 bits and above)

Name:	Integer factorization with TWIRL
Source:	Special-Purpose Hardware for Factoring: the NFS Sieving Step (Shamir and Tromer, 2005)
Description:	As the authors comment, it is commonly claimed that 1024-bit RSA keys are safe in a medium term (15 years, maybe more), since when applying the Number Field Sieve (NFS) to such composites both the sieving step and the linear algebra step would be unfeasible. However, the introduction of special-purpose hardware architectures for NFS, like TWINKLE or TWIRL, has reduced the predicted cost of factoring 1024-bit numbers by several orders of magnitude. The authors estimate that factoring a 1024-bit integer using TWIRL – the evolution of TWINKLE (see *Integer factorization with TWINKLE*) – would be possible in one year at the cost of a few dozen million US dollars.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.4: Cryptanalysis → D2-CAT5.4.1: Asymmetric algorithm
Target(s):	D3-CAT1: Cryptography
Countermeasures:	Use even larger RSA key lengths (2048 or 4096 bits)

Name:	Signature application substitution
Source:	None
Description:	This kind of attack tries to compromise sensitive data by replacing the SCA by a fake one. If the user does not notice the difference, he will have completely felt into the hands of the attacker. Depending on the purpose of the attack and the nature of the SCD, the attacker would be able to compromise either the SAD or the SCD itself. As such, two methods of attacks are applied.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.3: Unauthorized access to the SCDev → D2-CAT5.3.1: Compromise of the Signer Authentication Data (SAD) → D2-CAT4.1.2: SAD interception → D2-CAT4.1.2.3: Endpoint compromise D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.1: SCD interception → D2-CAT5.1.2: Endpoint compromise
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.2: SCA
Countermeasures:	Verify the integrity of the software before installing it. Implement integrity verification routines (e.g. TPM) for critical software during start-up

Name:	SCD compromise during issuance
Source:	None
Description:	The SCD is exposed and can be intercepted by an attacker if the Certification Authority sends the SCD through an unprotected channel to the entity in charge of writing the SCD in the SSCDev.
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT5: Compromise of the Signature Creation Data (SCD) → D2-CAT5.1: SCD interception → D2-CAT5.1.1: Interception in interprocess/entities communication
Target(s):	D3-CAT2: Software → D3-CAT2.4: Network → D3-CAT2.4.1: Protocols
Countermeasures:	Use of protected channels

Name:	SAD compromise by shoulder surfing
Source:	Information Systems Security: A Practitioner's Reference (Fites and Kratz, 1993).
Description:	The attacker observes the signer introducing the SAD in the Platform of the SCS (e.g. before generating a signature).
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT4: Unauthorized invocation of the signing function → D2-CAT4.1: Compromise of the Signer Authentication Data (SAD) → D2-CAT4.1.2: SAD interception → D2-CAT4.1.2.1: Observation
Target(s):	D3-CAT4: Human user → D3-CAT4.1: Signer
Countermeasures:	–

Name:	SAD compromise by optical emanation
Source:	Information Leakage from Optical Emanations (Loughry and Umphress, 2002)
Description:	The authors describe two implementations of a Trojan horse that manipulates the LEDs on a standard keyboard to implement a high-bandwidth covert channel. The attack can be mounted to obtain the information stored in the computer or typed by the user (e.g. the SAD).
Goal:	D1-CAT2: Unauthorized use of the Signature Creation Data (SCD)
Method:	D2-CAT4: Unauthorized invocation of the signing function → D2-CAT4.1: Compromise of the Signer Authentication Data (SAD) → D2-CAT4.1.2: SAD interception → D2-CAT4.1.2.1: Observation
Target(s):	D3-CAT3: Hardware → D3-CAT3.2: Computer → D3-CAT3.2.4: Peripheral devices → D3-CAT3.2.4.2: Keyboard
Countermeasures:	–

Name:	Font type manipulation – Fonts name change
Source:	What You See is Not Always What You Sign (Jsang et al., 2002)
Description:	This attack improves *Font type manipulation – Fonts substitution* attack by using a customized font type renamed to the expected one. As a result, the verifier is not able to distinguish whether the computer where the signature was computed had a different font type installed.
Goal:	D1-CAT5: Make the Data To Be Verified (DTBV) show chosen content
Method:	D2-CAT1: Environment manipulation
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Use of formats (e.g. PDF) that include the fonts definitions inside the content of the document

Name:	False positives in ASN.1
Source:	What You See is Not Always What You Sign (Jsang et al., 2002)
Description:	If the verifier uses an ASN.1 encoding rules different than the certificate issuer, it permits an attacker to generate a signature with a revoked certificate without being detected in the CRLs.
Goal:	D1-CAT6: Make the signature validity verification conclude with an opposite result
Method:	D2-CAT6: Influence on certificate verification result → D2-CAT6.2: Alteration of certificate status verification → D2-CAT6.2.6: Alteration of certificate status verification result
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.5: SVA
Countermeasures:	Correct application of encoding rules

Name:	Secure viewer compromise for fraudulent signature verification (1)
Source:	Malware Attacks on Electronic Signatures Revisited (Langweg, 2006)
Description:	The attack is carried out on Deutsche Telekom T-Telesec Signet 1.6.0.4 product. The attack does not need administrator privileges and relies on design flaws, not implementation ones. In particular, the attack modifies the viewer's presentation surface without detection to deceive the user respecting the result of the signature verification.
Goal:	D1-CAT6: Make the signature validity verification conclude with an opposite result
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.3: Verification result masquerading
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.5: SVA
Countermeasures:	–

Name:	Secure viewer compromise for fraudulent signature verification (2)
Source:	Malware Attacks on Electronic Signatures Revisited (Langweg, 2006)
Description:	The attack is carried out on IT Solution trustDesk standard 1.2.0 product. The attack does not need administrator privileges and relies on design flaws, not implementation ones. In particular, the attack modifies the viewer's presentation surface without detection to deceive the user respecting the result of the signature verification.
Goal:	D1-CAT6: Make the signature validity verification conclude with an opposite result
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.3: Verification result masquerading
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.5: SVA
Countermeasures:	–

Name:	Secure viewer compromise for fraudulent signature verification (3)
Source:	Malware Attacks on Electronic Signatures Revisited (Langweg, 2006)
Description:	The attack is carried out on D-Sign matrix/digiSeal 3.0.1 product. The attack does not need administrator privileges and relies on design flaws, not implementation ones. In particular, the attack modifies the viewer's presentation surface without detection to deceive the user respecting the result of the signature verification.
Goal:	D1-CAT6: Make the signature validity verification conclude with an opposite result
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.3: Verification result masquerading
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.5: SVA
Countermeasures:	–

Name:	Manipulated presentation of signed data for fraudulent verification
Source:	Malware Attacks on Electronic Signatures Revisited (Langweg, 2006)
Description:	This attack violates the What-Is-Presented-Is-What-Is-Signed (WIPIWIS) principle. The attack is carried out on Ventasoft venta-sign 2.0.0.968 product. The attack does not need administrator privileges and relies on design flaws, not implementation ones. In particular, the attack modifies the application's presentation surface without detection to deceive the user respecting the signature verification and integrity checker software results.

Goal:	D1-CAT6: Make the signature validity verification conclude with an opposite result
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.1: DTBV masquerading → D2-CAT7.1.1.1: Document masquerading
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.5: SVA
Countermeasures:	–

Name:	Secure viewer compromise for fraudulent signature verification (4)
Source:	Malware Attacks on Electronic Signatures Revisited (Langweg, 2006)
Description:	The attack is carried out on 2B Secure FILE 1.0 product. The attack does not need administrator privileges and relies on design flaws, not implementation ones. In particular, the attack modifies the viewer's presentation surface without detection to deceive the user respecting the result of the signature verification.
Goal:	D1-CAT6: Make the signature validity verification conclude with an opposite result
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.3: Verification result masquerading
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.5: SVA
Countermeasures:	–

Name:	Secure viewer compromise for fraudulent signature verification (5)
Source:	Malware Attacks on Electronic Signatures Revisited (Langweg, 2006)
Description:	The attack is carried out on Ultimaco SafeGuard Sign & Crypt for Office 3.4.1 product. The attack does not need administrator privileges and relies on design flaws, not implementation ones. In particular, the attack modifies the viewer's presentation surface without detection to deceive the user respecting the result of the signature verification.
Goal:	D1-CAT6: Make the signature validity verification conclude with an opposite result
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.3: Verification result masquerading
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.5: SVA
Countermeasures:	–

Name:	Collisions in PDF Signatures
Source:	Collisions in PDF Signatures (Zumbiehl, 2010)
Description:	This attack violates the What-Is-Presented-Is-What-Is-Signed (WIPIWIS) principle. The author describes a vulnerability in the PDF standard. Using this vulnerability, an attacker is capable of producing a PDF document which is shown differently when opened, and due to the way the signature blob had been injected by the attacker. Therefore, two different (as shown) documents produce the same signature.
Goal:	D1-CAT5: Make the Data To Be Verified (DTBV) show chosen content
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.1: DTBV masquerading → D2-CAT7.1.1.1: Document masquerading
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	–

Name:	Dali attack (verification)
Source:	The Dali Attack on Digital Signature (Buccafurri et al., 2008)
Description:	This attack violates the What-Is-Presented-Is-What-Is-Signed (WIPIWIS) principle. Attack based on the capability of a file of having a static polymorphic behavior. The attacker prepares the signed document to include a secondary content. Thanks to certain formats tagging, the content shown to the verifier varies depending on the file extension, and thus the application chosen to open the file. The attack is limited to the inclusion of HTML as the malicious secondary content.
Goal:	D1-CAT5: Make the Data To Be Verified (DTBV) show chosen content
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.2: Viewer manipulation → D2-CAT7.1.2.1: Viewer substitution
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Inclusion of the signed attribute *content-type* in the electronic signature format (e.g. CAdES, XAdES)

Name:	Enhanced Dali attack (verification)
Source:	Fortifying the Dali Attack on Digital Signature (Buccafurri et al., 2009)
Description:	Attack that enhances the Dali Attack to permit the usage of tiff and PDF formats for the contents inserted in the signed document.
Goal:	D1-CAT5: Make the Data To Be Verified (DTBV) show chosen content
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.2: Viewer manipulation → D2-CAT7.1.2.1: Viewer substitution

Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Use of PDF/A formats. Use of PDF Advanced Electronic Signature (PAdES) formats. Inclusion of the signed attribute *content-type* in the electronic signature format (e.g. CAdES, XAdES)

Name:	Inconsistent handling of HTML table tags (verification)
Source:	What You See is Not Always What You Sign (Jsang et al., 2002)
Description:	This attack violates the What-Is-Presented-Is-What-Is-Signed (WIPIWIS) principle. Web browsers interpret HTML and Javascript code in a different manner. Consequently, the same HTML code can be shown in different ways depending on the web browser used.
Goal:	D1-CAT5: Make the Data To Be Verified (DTBV) show chosen content
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.1: DTBV masquerading → D2-CAT7.1.1.1: Document masquerading
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.1: External application → D3-CAT2.1.1.2: User level application
Countermeasures:	Detect the existence of dynamic content in the signed document

Name:	Substitution of Office document by external content using macros (verification)
Source:	Electronic Documents and Digital Signatures (Kain, 2003)
Description:	This attack violates the What-Is-Presented-Is-What-Is-Signed (WIPIWIS) principle. When opening the signed document, some active code (e.g. a macro programmed in Visual Basic for Applications for a Word document or an Excel spreadsheet) included in it substitutes the content of the document by an external content controlled by the attacker. This attack is feasible on Microsoft Office formats. As the signature is verified against the initial object, the signature integrity is not corrupted.
Goal:	D1-CAT5: Make the Data To Be Verified (DTBV) show chosen content
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.1: DTBV masquerading → D2-CAT7.1.1.1: Document masquerading
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Detect the existence of dynamic content in the signed document

Name:	External queries in Excel (verification)
Source:	Electronic Documents and Digital Signatures (Kain, 2003)
Description:	This attack violates the What-Is-Presented-Is-What-Is-Signed (WIPIWIS) principle. Excel includes features to make explicit queries to remote files. The attacker can select an option to get external data and set up a query to a remote text file. The text file should be written with tab spaces between words to specify different fields in the spreadsheet. By right-clicking on the cell and selecting Data Range Properties, the attacker can configure the query to update on open or even regularly (in the background).
Goal:	D1-CAT5: Make the Data To Be Verified (DTBV) show chosen content
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.1: DTBV masquerading → D2-CAT7.1.1.1: Document masquerading
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Detect the existence of dynamic content in the signed document

Name:	Substitution of Office document content by means of fields (verification)
Source:	Electronic Documents and Digital Signatures (Kain, 2003)
Description:	This attack violates the What-Is-Presented-Is-What-Is-Signed (WIPIWIS) principle. Several attacks can be performed using the field feature in some Office formats, like Word or Excel. Fields like TIME, USERNAME, etc. can make the visualization of a document content vary according to conditions controlled by the attacker. For instance, depending on the date when a document is opened or the user that opens the document, a piece of text can take one of several different possibilities. The content dependent on a field can be updated automatically in certain versions of Microsoft Word or explicitly via a macro.
Goal:	D1-CAT5: Make the Data To Be Verified (DTBV) show chosen content
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.1: DTBV masquerading → D2-CAT7.1.1.1: Document masquerading
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Detect the existence of dynamic content in the signed document

Name:	Substitution of PDF content by means of javascript (verification)
Source:	Electronic Documents and Digital Signatures (Kain, 2003)
Description:	

	This attack violates the What-Is-Presented-Is-What-Is-Signed (WIPIWIS) principle. The attacker can use the form toolbar to create a form field, and then add Javascript code in its calculate field to change the value of the field according to the date.
Goal:	D1-CAT5: Make the Data To Be Verified (DTBV) show chosen content
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.1: DTBV masquerading → D2-CAT7.1.1.1: Document masquerading
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Detect the existence of dynamic content in the signed document

Name:	Modification of HTML email content via Javascript (verification)
Source:	Electronic Documents and Digital Signatures (Kain, 2003)
Description:	This attack violates the What-Is-Presented-Is-What-Is-Signed (WIPIWIS) principle. An attack that modifies the content of an email formatted as HTML is performed by using the *document.write()* Javascript function and the current date.
Goal:	D1-CAT5: Make the Data To Be Verified (DTBV) show chosen content
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.1: DTBV masquerading → D2-CAT7.1.1.1: Document masquerading
Target(s):	D3-CAT2: Software → 3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Detect the existence of dynamic content in the signed document

Name:	Modification of HTML email content via embedded image (verification)
Source:	Electronic Documents and Digital Signatures (Kain, 2003)
Description:	This attack violates the What-Is-Presented-Is-What-Is-Signed (WIPIWIS) principle. The attacker embeds an image in an HTML formatted email and, in conjunction with Javascript, is able to modify the visualized content of the signed email.
Goal:	D1-CAT5: Make the Data To Be Verified (DTBV) show chosen content
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.1: DTBV masquerading → D2-CAT7.1.1.1: Document masquerading
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.1: Document processor
Countermeasures:	Detect the existence of dynamic content in the signed document

Name:	Modification of the request of revocation of a compromised certificate to achieve successful fraudulent signature verification
Source:	This document
Description:	The premise of this attack is that the attacker has compromised a private key with which he wants to sign a document on behalf of the legitimate owner. It is also assumed that the owner of the key has detected such compromise, and thus proceeds to revoke the corresponding certificate. In this potential attack, the revocation request is modified by the attacker before it is authenticated by the owner of the certificate. For the attack to be effective, the attacker must change the information of the request that identifies the certificate which revocation is being requested. As a result, the revocation will not become effective, and the verifier will conclude that the signature is valid.
Goal:	D1-CAT6: Make the signature validity verification conclude with an opposite result
Method:	D2-CAT6: Influence on certificate verification result → D2-CAT6.1: Alteration of subscriber's revocation request → D2-CAT6.1.2: Modification of revocation request
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.1: External application → D3-CAT2.1.1.2: User level application
Countermeasures:	–

Name:	Deny the revocation of a compromised certificate to achieve successful fraudulent signature verification
Source:	This document
Description:	The premise of this attack is that the attacker has compromised a private key with which he wants to sign a document on behalf of the legitimate owner. It is also assumed that the owner of the key has detected such compromise, and thus proceeds to revoke the corresponding certificate. In this potential attack, the revocation request is intercepted by the attacker. If the revocation protocol does not incorporate a revocation response (e.g. as permitted by IETF CMP (RFC 4210, 2005), the owner of the certificate will not notice whether the revocation reached the certification authority or not. As a result, the revocation will not become effective, and the verifier will conclude that the signature is valid.
Goal:	D1-CAT6: Make the signature validity verification conclude with an opposite result
Method:	D2-CAT6: Influence on certificate verification result → D2-CAT6.1: Alteration of subscriber's revocation request → D2-CAT6.1.1: DoS of revocation request
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.1: External application → D3-CAT2.1.1.2: User level application
Countermeasures:	–

Name:	Identity theft by untrusted trust anchor addition
Source:	This document
Description:	In this potential attack, the attacker produces either a self-signed certificate or a certificate issued by a faked certification authority. This certificate contains the identity of the victim. Afterward, the attacker compromises the trusted store of the verifier to inject the trust anchor that will allow a successful certification chain validation. Thereby, the attacker is able to sign documents masquerading as another entity (the victim), and the verifier will trust the fake certificate.
Goal:	D1-CAT4: Attribute the signed document to a user different to the actual signer
Method:	D2-CAT6: Influence on certificate verification result → D2-CAT6.3: Untrusted trust anchor/trust point addition
Target(s):	D3-CAT5: Information → D3-CAT5.3: Cryptographic material → D3-CAT5.3.1: Trust store
Countermeasures:	–

Name:	Successful fraudulent signature verification by delaying the time-stamped signature sending
Source:	This document
Description:	CEN CWA 14171 (2004) establishes that the verifier, before assessing the validity of the certificate associated to the signature, should ascertain that at least the grace period has elapsed since a signature relevant time. The grace period is defined as the *time period which permits the certificate revocation information to propagate through the revocation process to relying parties; it is the minimum time period an initial verifier has to wait to allow any authorized entity to request a certificate revocation and the relevant revocation status provider to publish revocation status*. CEN CWA 14171 also indicates that the signature relevant time should be the time indicated in an associated TST or in an associated time mark. On the other hand, the cautionary period is defined at Certification Practices Statement level (RFC 3647, 2003), which allows the legitimate owner of a digital certificate to withdraw the validity of a recently generated signature by revoking the corresponding certificate a posteriori, that is, once the signature has been computed. Assuming a delay between the time when a key is compromised and the time when the user notices it and requests the revocation of the corresponding certificate(s), the cautionary period offers the users a mechanism for preventing the attackers to benefit from the signatures performed during this time frame. The verifier should wait a period (the cautionary period) after receiving a signature to allow certificate revocation requests to be processed by the CA, even when these requests were made after the signature computation. In this situation, grace and cautionary periods mean the same concept. In this potential attack, it is being assumed that the legitimate owner of the certificate (user) cannot detect the private key compromise before the attacker makes use of the signed document and the corresponding signature. On the other hand, it is also assumed that the attacker cannot benefit from the signed document before the cautionary period expires, diminishing the attacker's chances. Section 5.2 of CEN CWA 14171 permits that a signer acts as an initial verifier as well, being capable of adding a trusted time-stamp or time-mark to the signature. Suppose that an attacker compromises a user's private key, signs a desired document with it and time stamps the generated signature. Let's consider that the user detects the key compromise once another entity, like the verifier, receives the signature. If an entity different to the attacker knows the existence of the signature, it is possible that the user is somehow notified about that (possibly during the grace period) and then he could proceed to request the certificate(s) revocation, preventing the attacker to benefit from the forged signature. However, if the attacker delays the signature sending until the CRL is updated, then the verifier will possess a CRL issued after the signing time (specified by the time-stamp), and will not wait for any further update. The CRL next update value can be easily guessed by the attacker just by taking a look at the 'nextUpdate' field of the CRL data structure (RFC 5280, 2008). As a result, and though made, the revocation request will have no effect. The signature will be considered valid and the attacker will be able to benefit from it although the certificate revocation is afterward published. This attack could also be performed using time-marks.
Goal:	D1-CAT6: Make the signature validity verification conclude with an opposite result
Method:	D2-CAT6: Influence on certificate verification result → D2-CAT6.2: Alteration of certificate status verification → D2-CAT6.2.1: Grace or cautionary period bypassing → D2-CAT6.2.1.1: Delay in time-stamped signature sending
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.5: SVA
Countermeasures:	If the verifier receives a signature a long time after the time indicated in the time-stamp included in the signature by the signer, then the attack described herein could have been applied. A security policy should indicate whether the signature should be considered as invalid or not, depending on such elapsed time

Name:	Successful fraudulent signature verification by exploiting the delay in CA's revocation request processing
Source:	This document
Description:	In this potential attack, an attacker has compromised a private key and generated a signature with it. Let's suppose that the user detects it, and requests the revocation of their certificate c1, indicating time t_0 as the time on which he suspects that the private key was compromised (i.e. invalidityDate, according to RFC 5280 (2008)). The certification authority (CA) receives the revocation request at time t_1, but does not process it till time t_3. Meanwhile, at time t_2 ($t_1 < t_2 < t_3$) the CA publishes a new CRL without the

	revocation information about c1. Therefore, delay t_3 − t_1 prevents the CA from publishing a properly updated CRL at time t_2. A verifier that is validating certificate c1 at a time later than t_0 but before t_2, and following current standards recommendations, waits the grace period before concluding about the validity or invalidity of such certificate. Because next CRL is published at time t_2, that is the one used for the certificate status validation, reaching the conclusion that certificate c1 is valid.
Goal:	D1-CAT6: Make the signature validity verification conclude with an opposite result
Method:	D2-CAT6: Influence on certificate verification result → D2-CAT6.2: Alteration of certificate status verification → D2-CAT6.2.1: Grace or cautionary period bypassing → D2-CAT6.2.1.3: Exploit delay in CA's revocation request processing
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.6: CA
Countermeasures:	Use updated revocation information, possibly by accessing an Online Certificate Status Protocol (OCSP) service

Name:	Low-level LDAP injection techniques to avoid detection of revoked certificate
Source:	This document
Description:	An attacker that is capable of modifying the status validation request made by the verifier will prevent him from checking the actual status of the certificate. Therefore, although the certificate was revoked by the user due to a key compromise, the attacker will make the verifier conclude that the signature is valid. LDAP injection techniques (Alonso et al., 2008) can be used to modify the LDAP query that contains the certificate subject Distinguished Name, making the LDAP server search for a different or nonexistent object. Contrary to classical LDAP injection techniques, where the LDAP query is altered by the attacker due to the malicious input entered from a client application (e.g. Web browser), in this attack the query must be modified at a lower level, for example, before the SVA sends the query to the LDAP server, and once it has been composed.
Goal:	D1-CAT6: Make the signature validity verification conclude with an opposite result
Method:	D2-CAT6: Influence on certificate verification result → D2-CAT6.2: Alteration of certificate status verification → D2-CAT6.2.2: Modification of certificate status verification request → D2-CAT6.2.2.2: Modification of LDAP-based request
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.5: SVA
Countermeasures:	Protect queries and responses from integrity attacks (e.g. LDAP-s), and check whether the given response's search criteria matches with the desired one

Name:	Modification of the OCSP response to avoid detection of revoked certificate (1)
Source:	This document
Description:	This potential attacks requires the attacker to be capable of modifying the OCSP response and subvert the OCSP-response signature verification mechanism in order to prevent the verifier from detecting the violation of the signature integrity. Therefore, it is assumed that the OCSP response has been signed by the OCSP server. In this particular attack, the attacker modifies the field *OCSPResponse.responseBytes.response.tbsResponseData.responses[i].certStatus*, setting its value to 'good'. To subvert the signature verification mechanism, the attacker should apply mechanisms covered by *D2-CAT7: Influence on signature verification result* category, what would fall into a secondary attack not considered herein for classification.
Goal:	D1-CAT6: Make the signature validity verification conclude with an opposite result
Method:	D2-CAT6: Influence on certificate verification result → D2-CAT6.2: Alteration of certificate status verification → D2-CAT6.2.3: Modification of certificate status verification response
Target(s):	D3-CAT5: Information → D3-CAT5.2: Protocol message
Countermeasures:	–

Name:	Modification of the OCSP response to avoid detection of revoked certificate (2)
Source:	This document
Description:	This potential attacks requires the attacker to be capable of modifying the OCSP response, signing it with a certificate of their own, and subvert the mechanisms that verify the certification chain. In particular, the attack would cover the modification of the field *OCSPResponse.responseBytes.response.tbsResponseData.responses[i].certStatus*, setting its value to 'good'. The operations of signing the modified OCSP response with a certificate of their own, and injecting as trust point such certificate, fall into a secondary attack, covered by *D2-CAT6: Influence on certificate verification result → D2-CAT6.3: Untrusted trust anchor/trust point addition* subcategory.
Goal:	D1-CAT6: Make the signature validity verification conclude with an opposite result
Method:	D2-CAT6: Influence on certificate verification result → D2-CAT6.2: Alteration of certificate status verification → D2-CAT6.2.3: Modification of certificate status verification response
Target(s):	D3-CAT5: Information → D3-CAT5.2: Protocol message
Countermeasures:	–

Name:	Modification of time-stamp to avoid detection of revoked certificate
Source:	This document
Description:	This potential attack requires the attacker to be able to modify the time-stamp of the signature without detection. Possible mechanisms that can be used further to avoid such detection include subcategories under *D2-CAT6: Influence on certificate verification result* category and *D2-CAT7: Influence on signature verification result* category.
Goal:	D1-CAT6: Make the signature validity verification conclude with an opposite result
Method:	D2-CAT6: Influence on certificate verification result → D2-CAT6.2: Alteration of certificate status verification → D2-CAT6.2.4: Alteration of time reference verification → D2-CAT6.2.4.1: Modification of time-stamp
Target(s):	D3-CAT5: Information → D3-CAT5.3: Cryptographic material → D3-CAT5.3.2: Time-stamp
Countermeasures:	If the verifier receives a signature a long time after the time indicated by the time-stamp, then the attack described herein could have been applied. A security policy should indicate whether the signature should be considered as invalid or not, depending on such elapsed time

Name:	Document masquerading during a document authorization chain
Source:	This document
Description:	This potential attack violates the What-Is-Presented-Is-What-Is-Signed (WIPIWIS) principle. In a situation where a signer has to authorize or approve a signed document authored by another (e.g. by countersigning a signature) but after its verification, it might be of interest to the attacker to alter the visualization of the signed document in order to show the intended one. As a result, the authorization would be produced, but over the fraudulent document. In this attack, it is assumed that the attacker has been able to obtain a signature on behalf of the purported signer over a fraudulent document, and that the attacker possesses the intended document as well. Afterward, the attacker sends to the SVA the pair fraudulent document-signature, what is correctly verified, but makes the SVA show the intended document to the second signer.
Goal:	D1-CAT5: Make the Data To Be Verified (DTBV) show chosen content
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.1: DTBV masquerading → D2-CAT7.1.1.1: Document masquerading
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.5: SVA
Countermeasures:	–

Name:	Showing a different signer during the signature verification
Source:	This document
Description:	This potential attack violates the What-Is-Presented-Is-What-Is-Signed (WIPIWIS) principle, regarding the signed attribute signing-certificate, as defined by Advanced Electronic Signature Formats (AdES) (ETSI TS 101 733 v1.7.4, 2008; ETSI TS 101 903 v1.3.2, 2006). In this attack, the attacker makes the SVA show a signer different that the actual one. This attack could be launched once the SVA has read the information contained in the certificate signed as attribute (signing-certificate attribute), and possibly by modifying regions of the visualization area of the application (see Cut and paste attacks with Java (Lefranc and Naccache, 2002)).
Goal:	D1-CAT4: Attribute the signed document to a user different to the actual signer
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.1: Presentation manipulation → D2-CAT7.1.1: DTBV masquerading → D2-CAT7.1.1.2: Attribute masquerading
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.5: SVA
Countermeasures:	–

Name:	Injection of different signature-signed data pair during verification
Source:	This document
Description:	In this potential attack, it is assumed that the attacker possesses a document signed by the signer and the corresponding signature, but different to the signed document and signature that is to be verified. Therefore, the attacker replaces the information during the verification process by injecting into the SVA the former pair of signed document-signature. For example, if two versions of a draft document have been signed by the author, but he only wanted to distribute the newest one for approval, the attacker might want to replace the draft and corresponding signature by the oldest pair.
Goal:	D1-CAT5: Make the Data To Be Verified (DTBV) show chosen content
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.3: Alteration of verification process → D2-CAT7.3.1: Injection of signature-signed data pair
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.5: SVA
Countermeasures:	–

Name:	Modification of cryptographic verification result
Source:	This document
Description:	In this potential attack, if the attacker had access to the routine of the cryptographic verification, then the attacker would be able to make a signature be verified as valid when the integrity was broken.

Goal:	D1-CAT6: Make the signature validity verification conclude with an opposite result
Method:	D2-CAT7: Influence on signature verification result → D2-CAT7.3: Alteration of verification process → D2-CAT7.3.2: Alteration of cryptographic verification result
Target(s):	D3-CAT2: Software → D3-CAT2.1: Application → D3-CAT2.1.2: Related application → D3-CAT2.1.2.5: SVA
Countermeasures:	–

REFERENCES

Aciimez O. Yet another microarchitectural attack: exploiting i-cache. In: Proceedings of the 2007 ACM workshop on computer security architecture; 2005. p. 11–8.

Aciimez O, Schindler W, Ko K. Improving Brumley and Boneh timing attack on unprotected SSL implementations. In: Proceedings of the 12th ACM conference on computer and communications security. ACM Press; 2005. p. 139–46.

Aciimez O, Ko K, Seifert J-P. Predicting secret keys via branch prediction. In: Topics in cryptology – RSA conference 2007. Springer-Verlag; 2007a. p. 225–42. LNCS 4377.

Aciimez O, Ko K, Seifert J-P. On the power of simple branch prediction analysis. In: ACM symposium on Information, Computer and Communications Security (ASIACCS '07). ACM Press; 2007b. p. 312–20.

Alonso JM, Bordon R, Beltran M, Guzman A. LDAP injection techniques. In: 11th IEEE Singapore International Conference on Communication Systems (ICCS 2008); 2008. p. 980–6.

Amoroso E. Fundamentals of computer security technology. Prentice-Hall; 1994.

Barenghi A, Bertoni G, Palomba A, Susella R. A novel fault attack against ecdsa. Proceedings of the IEEE international symposium on Hardware-Oriented Security and Trust (HOST) 2011;(June):161–6.

Bevan R, Knudsen E. Ways to enhance DPA. In: Proceedings of the ICISC 2002. Springer-Verlag; 2003. p. 327–42. LNCS 2587.

Bishop M. Technical report CSE–9510-a taxonomy of (Unix) system and network vulnerabilities. Technical report. Department of Computer Science, University of California; 1995.

Broderick MA, Gibson VR, Tarasewich P. Electronic signatures: they're legal, now what? Internet Research: Networking Applications and Policy 2001;11:423–34.

Brumley D, Boneh D. Remote timing attacks are practical. In: Proceedings of the 12th Usenix security symposium; 2003.

Buccafurri F, Caminiti G, Lax G. The Dali attack on digital signature. Journal of Information Assurance and Security 2008:185–94.

Buccafurri F, Caminiti G, Lax G. Fortifying the dali attack on digital signature. In: 2nd ACM international conference on Security of Information and Networks (SIN 2009). North Cyprus: ACM Press; 2009. p. 278–87.

CEN Workshop Agreement 14170. Security requirements for signature creation applications. The European Committee for Standardization; 2004.

CEN Workshop Agreement 14169. Secure signature-creation devices. The European Committee for Standardization; 2004.

CEN Workshop Agreement 14171. General guidelines for electronic signature verification. The European Committee for Standardization; 2004.

Chari A, Rao J, Rohatgi P. Template attacks. In: Proceedings of the CHES 02. Springer Verlag; 2002. p. 13–28. LNCS 2523.

Chen C, Wang T, Tian J. Improving timing attack on rsa-crt via error detection and correction strategy. Elsevier Information Sciences 2012;0.

Coppersmith D. Another birthday attack. In: Advances in cryptology – proceedings of Crypto '85. Springer-Verlag; 1985. p. 369–78. LNCS 218.

Dasgupta P, Chatha K, Gupta SKS. Vulnerabilities of PKI based smartcards. In: IEEE Military Communications Conference (MILCOM 2007); 2007. p. 1–5.

Department of Justice, Government of Canada. Personal information protection and electronic documents act; 2008.

Diffie W, Hellman M. New directions in cryptography. IEEE Transactions of Information Theory 1976;22:644–54.

ECRYPT. The side-channel cryptanalysis lounge. Available at: http://www.emsec.rub.de/research/projects/sclounge/; 2008.

European Directive 1999/93/CE of the European Parliament and of the Council of 13 December 1999 on a Community framework for electronic signatures; 1999.

ETSI TS 101 903 v1.3.2. XML Advanced Electronic Signatures (XAdES). European Telecommunications Standards Institute; 2006.

ETSI TS 101 733 v1.7.4. Electronic Signatures and Infrastructures (ESI); CMS Advanced Electronic Signatures (CAdES). European Telecommunications Standards Institute; 2008.

Fahn P, Pearson P. Ipa: a new class of power attacks. In: Proceedings of the CHES 1999. Springer Verlag; 1999. p. 173–86. LNCS 1717.

Federal Trade Commission, Department of Commerce, United States of America. Electronic signatures in global and national commerce act (e-sign); 2000.

Fites P, Kratz MP. Information systems security: a practitioner's reference. New York: Van Nostrand Reinhold; 1993.

Gandolfi K, Mourtel C, Olivier F. Electromagnetic analysis: concrete results. In: Proceedings of the cryptographic hardware and embedded systems (CHES 2001). Springer-Verlag; 2001. p. 251–61. LNCS 2162.

Girard P, Giraud J-L. Software attacks on smart cards. Information Security Technical Report 2003;8:55–66.

Gutmann P. Breakms – break microsoft private key encryption with a dictionary attack. Available at: http://www.artofhacking.com/tucops/etc/crypto/live/aoh_breakms.htm; 1997.

Gutmann P. How to recover private keys for Microsoft internet explorer, internet information server, outlook express, and many others – or – where do your encryption keys want to go today?. Available at: http://www.cs.auckland.ac.nz/pgut001/pubs/breakms.txt; 1998.

Hansman S, Hunt R. A taxonomy of network and computer attacks. Computers & Security 2005;24:31–43.

Hill B. A taxonomy of attacks against xml digital signatures & encryption; 2004.

Howard JD, Longstaff TA. A common language for computer security incidents. Technical report. Sandia National Laboratories; 1998.

ISO/IEC 13888-1. Information technology – security techniques – non repudiation – part 1: general. ISO/IEC; 2009.

ISO/IEC 13888-3. Information technology – security techniques – non repudiation – part 3: mechanisms using asymmetric techniques. ISO/IEC; 2009.

Joye M, Paillier P, Schoenmarkers B. On second-order differential power analysis. In: Proceedings of the CHES 2005. Springer Verlag; 2005. p. 293–308. LNCS 3659.

Jsang A, Povey D, Ho A. What you see is not always what you sign. Melbourne: Proc. of the Australian UNIX User Group; 2002.

Kain K. Electronic documents and digital signatures. Master's thesis. Department of Computer Science, Dartmouth College; 2003.

Kelsey J, Kohno T. Herding hash functions and the nostradamus attack. In: Advances in cryptology – EUROCRYPT 2006. Springer-Verlag; 2006. p. 183–200. LNCS 4004.
Kelsey J, Schneier B. Second preimages on n-bit hash functions for much less than 2n work. In: EUROCRYPT 2005. Springer-Verlag; 2005. p. 474–90. LNCS 3494.
Kitchenham B. Technical report TR/SE-0401-procedures for performing systematic reviews. Technical report. Keele University; 2004.
Klima V. Tunnels in hash functions: MD5 collisions within a minute. IACR ePrint Archive; 2006.
Kmmerling O, Kuhn MG. Design principles for tamper-resistant smartcard processors. In: Proceedings of the USENIX Workshop on Smartcard Technology (WOST '99); 1999.
Kocher PC. Timing attacks on implementations of Diffie-Hellman, RSA, DSS and other systems. In: Advances in cryptology – CRYPTO '96. Springer Verlag; 1996. p. 104–13. LNCS 1109.
Kocher P, Jaffe J, Jun B. Differential power analysis. In: Proceedings of the CRYPTO 1999. Springer Verlag; 1999. p. 388–97. LNCS 1666.
Landwehr CE, Bull AR, McDermott JP, Choi WS. A taxonomy of computer program security flaws. ACM Computing Surveys 1994;26:211–54.
Langweg H. Malware attacks on electronic signatures revisited. In: Sicherheit 2006: Jahrestagung Fachbereich Sicherheit der Gesellschaft fr Informatik. Springer-Verlag; 2006. p. 244–55. LNI.
Le T, Cldire J, Canovas C, Servière C, Lacoume J, Robisson B. A proposition for correlation power analysis enhancement. In: Proceedings of the CHES 2006. Springer Verlag; 2006. LNCS 4249.
Le T-H, Canovas C, Clediere J. An overview of side channel analysis attacks. In: Proceeedings of the 2008 ACM Symposium on Information, Computer and Communications Security (ASIACCS 2008); 2008. p. 33–43.
Lefranc S, Naccache D. Cut and paste attacks with java. Cryptology ePrint Archive, report; 2002.
Lenstra AK, Shamir A. Analysis and optimization of the Twinkle factoring device. In: EUROCRYPT 2000. Springer-Verlag; 2000. p. 35–52. LNCS 1807.
Lindqvist U, Johsson E. How to systematically classify computer security intrusions. Proceedings of the IEEE Security and Privacy 1997:154–63.
Lough DL. A taxonomy of computer attacks with applications to wireless networks. PhD thesis Virginia Polytechnic Institute and State University; 2001.
Loughry J, Umphress DA. Information leakage from optical emanations. ACM Transactions on Information and System Security 2002;5:262–89.
Marchesini J, Smith S, Zhao M. Keyjacking: the surprising insecurity of client-side ssl. Computers & Security 2005;24:109–23.
Matthews A. Low cost attacks on smart cards: the electromagnetic side-channel. Next generation security software. Available at: http://www.ngssoftware.com/research/papers/EMA.pdf; 2006.
McCullagh A, Caelli W. Electronic signatures: they're legal, now what? Non-repudiation in the digital environment, vol. 5; 2000.
Pellegrini A, Bertacco V, Austin T. Fault-based attack of RSA authentication. In: Design, Automation and Test in Europe conference (DATE-2010); 2010.
Popescu D-S. Hiding malicious content in PDF documents. Journal of Mobile, Embedded and Distributed Systems 2011;3(3):120–7.
Quisquater J-J, Samyde D. Electromagnetic analysis (EMA): measures and counter-measures for smart cards. In: Proceedings of the international conference on research in smart cards (E-smart 2001). Springer-Verlag; 2001. p. 200–10. LNCS 2140.
Rae AJ, Wildman LP. A taxonomy of attacks on secure devices. In: Proceedings of the 4th Australian information warfare and IT security conference; 2003. p. 251–64.
RFC 2560. Internet X.509 public key infrastructure. Online certificate status protocol – OCSP. Internet Engineering Task Force; 1999.
RFC 3647. Internet X.509 public key infrastructure. Certificate policy and certification practices framework. Internet Engineering Task Force; 2003.
RFC 4210. Internet X.509 public key infrastructure. Certificate management protocol (CMP). Internet Engineering Task Force; 2005.
RFC 4949. Internet security glossary, version 2. Internet Engineering Task Force; 2007.
RFC 5280. Internet X.509 public key infrastructure. Certificate and certificate revocation list (CRL) profile. Internet Engineering Task Force; 2008.
Rivero DC. In: Pons M, editor. Eficacia formal y probatoria de la firma electrnica; 2006.
Rivest RL, Shamir A, Adleman L. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM 1978;21:120–6.
Rushby J. Technical report csl-93-01-critical system properties: survey and taxonomy. Technical report. Computer Science Laboratory. SRI International; 1994.
Schindler W. A timing attack against RSA with the Chinese remainder theorem. In: Proceedings of the Cryptographic Hardware and Embedded Systems (CHES 2000). Springer Verlag; 2000. p. 110–25. LNCS 1965.
Schindler W, Lemke K, Paar C. A stochastic model for differential side channel cryptanalysis. In: Proceedings of the CHES 2005. Springer Verlag; 2005. p. 30–46. LNCS 3659.
Shamir A, Tromer E. Special-purpose hardware for factoring: the NFS sieving step. Invited talk at the workshop on Special purpose Hardware for Attacking Cryptographic Systems (SHARCS); 2005.
Sinha S, Sinha SK. Signature replacement attack and its counter–measures. In: Proceedings of the IEEE 2nd International Advance Computing Conference (IACC); 2010. p. 229–35.
Spalka A, Cremers AB, Langweg H. The fairy tale of what you see is what you sign. Trojan horse attacks on software for digital signatures. In: Proceedings of the IFIP working conference on security and control of IT in society-II; 2001. p. 75–86.
Spalka A, Cremers AB, Langweg H. Trojan horse attacks on software for electronic signatures. Informatica 2002;26: 191–204.
Stevens M, Lenstra A, de Weger B. Chosen-prefix collisions for md5 and applications. International Journal of Applied Cryptography 2012;(0–4):42–80.
Tanaka H. Evaluation of information leakage via electromagnetic emanation and effectiveness of tempest. IEICE Transactions on Information and Systems 2008;91: 1439–46.
Tiri K. Side-channel attack pitfalls. In: Proceedings of the 44th annual conference on design automation; 2007. p. 15–20.
TR-Seclab-0606-001. Practical security aspects of digital signature systems. International Secure Systems Lab; 2006.
United Nations. UNCITRAL model law on electronic signatures with guide to enactment; 2001.
Wang G, Wang S. Preimage attack on hash function RIPEMD. In: Proceedings of the 5th international conference on information security practice and experience. Springer-Verlag; 2009. p. 274–84. LNCS 5451.
Wang X, Yu H. How to break MD5 and other hash functions. In: Advances in cryptology – EUROCRYPT 2005, 24th annual international conference on the theory and applications of

cryptographic techniques. Springer-Verlag; 2005. p. 19–35. LNCS 3494.
Yeh Y-S, Huang T-Y, Lin H-Y, Chang Y-H. A study on parallel RSA factorization. Journal of Computers 2009;4:112–8.
Zhou J, Gollmann D. Evidence and non-repudiation. Journal of Network and Computer Applications 1997;20:267–81.
Zhuang L, Zhou F, Tygar JD. Keyboard acoustic emanations revisited. In: Proceedings of the 12th ACM conference on computer and communications security; 2005. p. 373–82.
Zumbiehl F. Collisions in PDF signatures. Available at: http://pdfsig-collision.florz.de/; 2010.

Jorge L. Hernandez-Ardieta is Part Time Professor and Affiliated Researcher in the Computer Science and Engineering Department at the University Carlos III of Madrid (UC3M), and Senior Engineer in the Cybersecurity Unit at Indra Sistemas (Spain). He received a B.Sc. and M.Sc. degree in Computer Engineering from the University Autonoma of Madrid, and a Ph.D. in Computer Science from UC3M. He participates in various industrial and standardization initiatives, being a member of ISO/IEC JTC1 SC27, CEN TC 224 and IEEE. His main research interests cover non-repudiation, attack modeling and categorization, cyber security information sharing, and security evaluation methodologies.

Ana I. Gonzalez-Tablas is Associate Professor in the Computer Science and Engineering Department at the University Carlos III of Madrid (UC3M). She is Telecommunications Engineer by the Polytechnic University of Madrid, Spain, since 1999 and received her Ph.D. degree in Computer Science from UC3M, Spain, in 2005. Her main research interests are security and privacy for location based services and digital signature applications. She has participated in several national and international research projects and has published in several conferences and journals. She is IEEE Member since 2004 and ACM Member since February 2006.

Jose M. de Fuentes is Teaching Assistant in the Computer Science and Engineering Department at the University Carlos III of Madrid (UC3M). He was the valedictorian of the M.Sc. programme, and received his Ph.D. degree in Computer Science from UC3M. His main research interests are evidence management and non-repudiation issues in vehicular networks.

Benjamin Ramos is Associate Professor in the Computer Science and Engineering Department at University Carlos III of Madrid (UC3M). His research is mainly focused on non-repudiation issues of electronic signatures. He received his Ph.D. degree in Computer Science from University Carlos III of Madrid, Spain, in 1999.