Received January 28, 2020, accepted February 11, 2020, date of publication February 19, 2020, date of current version February 28, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2975034

New XML Signature Scheme That Is Resistant to Some Attacks

GERARD WAWRZYNIAK AND IMED EL FRAY
Faculty of Computer Science and Information Technology, West Pomeranian University of Technology, 71-210 Szczecin, Poland

Corresponding author: Imed El Fray (ielfray@zut.edu.pl)

This work was supported by the West Pomeranian University of Technology.

ABSTRACT Supporting the execution of transactions through the use of electronic documents requires security. The scope of this security primarily involves ensuring the integrity, authenticity and non-repudiation of the origin of information. The signed XML document is a powerful tool that ensures the above features and the ease of processing and integration with various systems. An XML document may have many signatures, and each of them may sign different parts of the document. This feature is highly attractive, but in order to use it, the signature and structure of the document must be carefully designed. This article presents the existing risks associated with the use of XML signatures, focusing on XML signature wrapping vulnerability. This vulnerability is a consequence of the relationship between the XML signature and the signed document. The authors suggest that without neglecting the need for protection against the possibility of moving and replacing a fragment of the document, the use of secure XML signature references should also be considered and applied. The article proposes the use of secure signature templates as a countermeasure against the threat of an improper indication of the signed content defined in the signature reference. This threat is serious in automatic signature processing, where it is important to correctly indicate the signed content.

INDEX TERMS XML signature, signature content, XML signature reference, XML signature transformation, XML signature vulnerability, XML signature attack.

I. INTRODUCTION

One of the fundamental problems facing transaction execution is ensuring interoperability of services and legal security of information accompanying such executions or being the subject of the transaction. Information, in order to be safely used in the execution of various types of transactions, must have certain features, such as integrity, authenticity, non-repudiation and durability. Electronic signatures are used to provide information to ensure security (information possessing the mentioned features is a document). Over the years, subsequent formats of electronic signatures of various capabilities were created (Cryptographic Message Syntax v.1.5 - PKCS#7 [1], Cryptographic Message Syntax - CMS [16], or CMS Advanced Message Syntax - CADES [38] format and PDF Advanced Electronic Signature Profiles - PAdES [17]) and dedicated to documents. In the presented formats of signatures, the content of signatures is defined arbitrarily (rigidly defined by format specifications), which is a problem for interoperability that is understood as the possibility of flexible adjustment of the signature content for different types of documents, supporting the execution of transactions in a heterogeneous, dispersed environment with the participation of many parties. For example, the signature should cover different parts of the document.

The appearance of the XML format [15] as a means of implementing various types of documents in an interoperable way provided the possibility of defining syntax and implementing various types of documents supporting transactions. Extending the XML concept by XML electronic signature specification - XMLdSig [37] additionally made it possible to place in the XML signature various types of information extending the signature itself (as an equivalent of CADES). A considerable difference between standard signature formats (in which the content of the signature must comply with the specification of the signature standard) and XMLdSig signature is the ability to place considerable information (for example, XML Advanced electronic Signatures - XAdES [4]), especially many different signed contents, in a single signature.

The XML signature specification does not directly specify the content of the signature. In return, this signature provides a mechanism for indicating this content in many ways. The content of the XML signature is defined by a set of references

The associate editor coordinating the review of this manuscript and approving it for publication was Yassine Maleh.

indicating the signed resources (reference content), and the set of indicated contents is the final content of the XML signature. This approach makes it possible to sign parts of XML documents [2], [3], [5], [8], [14], [32], [33], [35] and even documents in any format [2], [3], [33], [35] and enables [6] multiple parties in a heterogeneous distributed environment with data security (entered and secured/signed by different parties for the transaction under consideration). Moreover, it is possible to define the syntax of any document [30], [31] to support the execution of transactions that involve multiple parties in a heterogeneous and dispersed environment with the assurance of data security through the use of the XML signature.

The presented features of an XML document and the XMLdSig signature provide considerable possibilities for modelling both the structure (syntax) of the document and signatures, thereby securing the specified fragments of documents (including external documents in any format). This approach enables automatic retrieval of the signature content and automatic verification of signatures in the document. However, it should be noted that in addition to the large benefits of such an approach, careless, incompetent or dishonest use of this feature can lead to abuse. There are risks that are not directly related to cryptography but are related to the specific nature of XML documents and signatures. In [6], [9], several attacks on XML signatures (and encryption) were presented and included denial of service by entity expansion. During DTD parsing, the definitions of the referred elements lead to multilevel document expansion and memory and processor load. Another group of attacks are attacks that are based on injecting code into the document itself (e.g., SOAP message [10]–[13]) or signature elements and are related to the processing of signatures provided that the references are built correctly; hence, the signature is built and verified correctly. A code (content) can appear in the document that which is not the content of the signature (is not covered by the signature), but it is processed and can lead to misinterpretation of the content of the document.

In the opinion of the authors, there may be another abuse (in addition to the above) that may occur at the moment of creating the signature (an intentional action of the signer). The attacker may indicate in the reference any content that is irrelevant from the point of view of the transaction being executed. This leads to the possibility of any modification of the content that is relevant to the executed transaction without affecting the validity of the signature. Automatic document processing leads to the processing of incorrect content despite a positive result of signature verification and, consequently, to an unexpected procedure of transaction execution.

The novelty of our work is therefore the extension of the signature scheme by the verification of the validity of the XML signature content. Such verification, based on the signature template proposed by the authors, protects against signing any undesirable content and the possibility of modifying relevant parts of the document.

Therefore, our contribution to solving the problem related to transaction security is of significance because we extend the knowledge in the field of automatic execution of secure transactions supported by XML-signed documents.

II. RELATED WORKS

An XML signature is a structure with syntax defined in the form of XML schema and detailed in the W3C recommendation [15], [31], [37]. The signature is a result of a sequence of cryptographic operations performed on the signed document [39]. These operations are based on asymmetric cryptography and hash functions. The distinctive feature of an XML signature according to [3] is its flexibility and the fact that it is embedded in an XML document. The characteristics of these features are detailed and described in the literature, e.g., [2], [8], [33], [36].

According to the W3C specification of the XML signature [37], the signature structure consists of four parts structurally constituting XML elements. "`SignedInfo`" is the element where the method of constructing the signature is specified, i.e., information about the method of canonicalization ("`CanonicalizationMethod`"), the signature algorithm ("`SignatureMethod`", the algorithm of the signature hash and encryption of the hash value) and the reference element ("`Reference`"), where the content of the signature is defined as the sum of content specified in particular references by means of the so-called transformation of signed resources. Each reference, apart from the transformation specification, also contains the indication of the hash algorithm and the value of the hash. The second element of the signature is the "`SignatureValue`" element, created during the generation of the signature and contains the signature value, i.e., the encrypted value of the hash (from each reference hash) using the algorithms specified in the "`SignatureMethod`" element. Next, the third element "`KeyInfo`" stores information about the data used to retrieve a public key to verify the signature and/or the identity of the signatory (e.g., public key, certificate or other). The last and fourth element is an optional "`Object`" element in which any content is placed in accordance with specific signature requirements, e.g., data related to the extended XML XAdES [4] signature. The presented signature structure is widely discussed and interpreted in the literature, e.g., [3] or [7].

The ability to define the content of the signature (in the "`SignatureInfo`" element) allows the creation of the following basic types of signatures: disconnected signature, in which the signed data are outside the XML document containing the signature (usually used to sign data in a format other than XML); and enveloped signature, including the content of the signature and the enveloping signature, in the signed document.

The concept of reference and transformation in an XML ("`Reference`") signature allows multiple signatures to be placed in the document, flexible modelling of the signature content and signing of any separable/non-continuous parts of the document. An example is the patient's medical history card, maintained by various doctors responsible for particular records (diagnoses) [8]. In this case, each new diagnosis that is added must be signed individually. This important feature is supported by the XML signature, as underlined by [2], [5]. The abovementioned XML signature possibilities are realized by

1. the signature of document fragments (XPath [45]), and
2. multiple signatures.

XML signature processing (creation and verification) differs from the traditional electronic signatures presented above, in which the signed content is not indicated in the signature and results directly or indirectly from the specification of standards for these types of signatures. An XML signature requires a standard cryptographic verification of the correctness of the signature value calculated from the value of the hash function. Moreover, the content of the signature and the hash values of the individual references (that are not a result of the standard describing the signature) should be calculated.

The XML signature processing method originates from the XMLDSIG specification, but different authors [3], [7], [32], [33], [41], [46]–[49] differently emphasize the actions that would be performed.

A full description of the processing of an XML signature that does not contradict the solutions of the abovementioned authors is presented in [29]. Below, a brief description is provided of the processing of an XML signature, highlighting the processing of the transformation (in the processes of processing XML signatures).

Generating a signature:

1. Creation of References
    - Locating and downloading resources via URI,
    - Performing the transformation described in the Transform element on the referred data objects, and
    - Using the algorithm of calculating the value of the hash function (shown in the "`DigestMethod`" element) and storing it in the "`DigestValue`" element.
2. Signature creation
    - Execution of the "`SignedInfo`" element canonization algorithm (includes Reference element),
    - Execution of a hash algorithm on the "`SignedInfo`" element (canonicalized) to calculate the hash value, and
    - Encrypting the hash value using a private key, where the received signature value is saved in the "`SignatureValue`" element.

Signature validation:

1. Reference validation
    - Performing document transformation in the Transform item,
    - Execution of a hash calculation algorithm to calculate the hash value of the transformed document, and
    - Comparison of the calculated value of the hash value with the value obtained from the XML signature.
2. Signature validation
    - Reading key information from a "`KeyInfo`" element or an external resource,
    - Calculation of the hash value for the "`SignedInfo`" element using the algorithm indicated in the "`SignedInfo`" element,
    - Decrypting the signature value "`Signature Value`" using the signer's public key and the "`SignatureMethod`", and
    - Comparison of the calculated signature value with the signature value obtained from the "`SignatureValue`" element.

In the area of integrity verification, the authors of [3], [7], [32], [29], [33], [46] present a procedure consistent with the above reference validation scheme. It should be noted that the presented method of this verification

- is limited to formal verification of the integrity of the content of individual references obtained as a result of the execution of the transformation references (reference specifications) and
- does not include verification of the correctness of the reference specification of the signature verification, understood as the compliance of the scope (fragments) of the signed document with the scope specified for a specific signature. This should be specified in the process of defining designing) the transaction execution, i.e., checking whether the reference specifications do not indicate non-agreed upon and inadequate fragments of the document, thus omitting the agreed upon and adequate fragments.

As a consequence, there is a risk that the integrity of the part of the document that is critical to the security and execution of the transaction or the step of the transaction, may not be guaranteed. The authors of [6] show that the signed object pointed out by URI (Uniform Resource Identifier) can be moved while keeping the correct result of the signature integrity verification and making it possible to insert malicious data in this place. When the data position is relevant for the interpretation of the meaning of the data, it can be used by the attacker to gain unauthorized access to the protected resources. In this way, other data are verified and other data are processed. Such an attack can be defined as an injection of unauthorized data into a signed XML document, preserving the integrity and authenticity of the signature. In particular, this type of attack is described in relation to the signature in the SOAP [10]–[13] message structure, where reference pointing through URI in combination with SOAP-specific processing can be used to perform the attack. The attacks described in [6] occur in specific reference cases in the signature, where the existence of an element at a specific location in the document is optional and where the unexpected presence of an element at another location is tolerated. As a countermeasure, it is proposed to use XPath expressions in the reference when a simple indication in the reference is inappropriate. In addition to countermeasures using XPath, attacks of this type can be detected using a security policy that prevents the use of signed and not understood elements. Unfortunately, this limits the full use of the processing possibilities offered by the SOAP protocol.

A similar risk was also observed in the work of [24]. As a countermeasure, the authors propose limiting the possibility of occurrence of the signed element to a clearly defined position in the document, which may lead to a reduction in the vulnerability of the document and increase its security. One of the ways of implementing such a concept may be to use the XML Schema definition [15]. At the same time, they emphasize that this approach does not guarantee security but is an important element in securing XML structures that are resistant to the Signature Wrapping attack.

Similarly, the work in [28] noted that the XML Signature Wrapping (XSW) attack is one of the most serious threats. It was noticed, however, that the countermeasures proposed so far usually provide protection only for specific cases. There is still no general, comprehensive approach. As a countermeasure, the authors propose a holistic and integrated approach consisting of the use of a dedicated architecture and a dedicated XML signature library, which allows for a secure and effective protection of XML data in accordance with standards. The proposed solution is dedicated to XML messages in SOAP message structures and utilizes the features of this protocol.

The work in [34], similar to that of other authors, notes a similar weakness in the use of XML signatures to secure SOAP messages. As a countermeasure, the authors propose not to use the Id attribute or XPath expression to indicate the signed element (within a SOAP message) and instead sign the entire SOAP message.

Reference [40] notes that with network services, despite the use of standards such as WS-Security [WS-Security spec.], some attacks on SOAP messages appear and lead to significant security failures. As an example of such an attack, the authors cite the work [6], emphasizing that such an attack can happen because the XML signature associated with an object in an XML document does not depend on the location of the object in the document. Moreover, the SOAP extension model has small limitations on the presence of the ''headers'' element and the elements inside the SOAP message. The authors propose to detect the XSW attack by counting the number of occurrences of each node in the SOAP message. The authors present the algorithm of counting nodes in SOAP messages and comparing the number of nodes resulting from the specification of references in the signature.

The work [42] also noted the discussed threat related to the possibility of moving the original message from outside of the ''Body'' element and injecting malicious content in place of the original message. The authors propose a new signature verification process using the length of the ''`header`'' and ''`body`'' elements, the ''`Timestamp`'' element, and the distance between ''`id`'' attributes in ''`header`'' and ''`body`'' elements.

In the paper [43], which was dedicated to security threats in cloud computing, the authors presented the problem of the XSW attack, in addition to the security issues of browsers and provider lockout. Four categories of such attacks are listed: simple ancestry context attacks, optional element context attacks, sibling value context attacks and sibling older context. In the scope of these attacks, the authors propose the use of the WS-Security concept and end-to-end encryption for SOAP messages. In this way, the attacker will not be able to listen to and retrieve the decrypted message in order to modify it.

In the paper [46], the authors note that in the context of XML documents, the use of SSL does not fully resolve the issue of integrity and non-repudiation of the data origin, and even the signed SOAP message is vulnerable to interception and subsequent manipulation of its content. According to [6], a SOAP message protected by an XML digital signature can be falsified without affecting the correctness of the signature. The authors propose a method of detecting an attack in a signed request (network service) using an additional ''position token'' as an addition to the signed XML elements. The attack can be detected during signature verification by comparison of the calculated request hash with the hash of the content signed in the request.

The analysis of the above literature shows that one of the basic standards for building network services is the SOAP protocol based on XML format and secured with an electronic XML signature. The use of a signature within SOAP uses the indication of the signed content by the reference, which indicates the value of the attribute ''id'' of the signed element. Such an approach causes a threat consisting of the possibility of ''moving'' the signed element to another location (within the XML document) and injecting the falsified content in its place, which leads to the verification of the signature with the use of different (real) content in the processing of the document within the transaction (falsified content). All authors present the problem in the context of SOAP news. The authors of the presented papers propose various methods of preventing the XSW attack in relation to SOAP messages. However, the problem may involve messages other than SOAP in the general approach of each XML document. The authors of this article propose, without questioning the necessity of detecting XSW type attacks, to protect the document against abuse involving the specification of references in the signature, which causes effects that are similar to those of XSW (the possibility of modifying the allegedly ''signed'' content). Such security should be based on verification of the correctness of the reference specification in the signature. This means checking whether the references in the signature indicate those elements in the XML document (not only SOAP) that are consistent with the parties' arrangements when designing the way and formats of data exchange for a specific transaction. This finding is observed because an automatically verified signature in an XML document without verifying the correctness of the signed content specification can lead to the same results as the discussed XSW attack but with the use of a different vulnerability (incorrect and unverified specification of the signed content). For this purpose, the authors of the article propose to extend the processes of creating and verifying XML signatures to include verification of references in the signature, rather than only the verification of the content indicated and "returned" by references. In other words, this approach would check whether references indicate resources in accordance with objective and agreed intentions of the parties implementing the transaction through

- defining the content of the signature (references) outside the signature structure, i.e., creating the so-called signature template, in which the signature template should be a reliable document created during the implementation of automatic transaction support,
- building (automatically) a signature based on a template, and
- verification of the signature based on the references in the signature and the references built based on the template. Different values of digest values mean violation of the signature.

III. SIGNATURE SCHEME

The basic signature scheme describes the method of generating the signature S and verifying the signature V. The signature scheme usually consists of three elements [22]:

- The algorithm for generating a key pair $G(k_s, k_v)$, where k_s is a private key (also called the signatory), and k_v is a public key (verifying). The verification key can also be a symmetric key (e.g., derived from a password). In the next part of the discussion, the generation of keys (or keys) G, usually presented in the literature as exceeding the area of interest presented in the article, is omitted.
- Signing algorithm S, whose task is to create a signature value σ_v for a specific message m using the signing key k_s, and
- Signature verification algorithm V, i.e., the algorithm checks the validity of the signature value σ_v of a specific message m, with the use of the k_v verification key, which is complementary to the k_s signing key used to create the signature.

A. SIGNATURE CREATION AND VERIFICATION ALGORITHM

The signing algorithm S is a set of five tuples consisting of the signed message m_o, the signature value σ_v, the verification key k_v, the indication of the encryption algorithm for the hash value $e(m)$ and the indication of the algorithm for calculating the hash value $h(m)$ from the following message:

$$\{m_o, \sigma_v, k_v, e(m), h(m)\} \tag{1}$$

where

- using the assumed hash function $h(m)$, we obtain the hash values of the $m_h = h(m_o)$ for the given m_o message (which is the content of the signature),
- using the encryption algorithm $e(m)$, we obtain the encryption value called the signature value $\sigma_v = e(m_h, k_s)$ with the use of the signing key k_s and the hash value m_h. The signing algorithm can be written as shown below:

$$\sigma_v \leftarrow S(m_o, k_s) = e(h(m_o), k_s) \tag{2}$$

The result of the process of executing the signing algorithm is data that are necessary for the later verification V of the signature. Verification takes place as follows:

- Decrypting the signature value σ_v, using the decryption function d(m) and the verification key k_v, which leads to obtaining the hash value $m_h = d(\sigma_v, k_v)$. The hash value can be written as shown below:

$$m_h = d(\sigma_v = e(m_h, k_s), k_v) \tag{3}$$

- Calculation of the hash value m'_h from the signed message m_0 using the $h(m)$ function:

$$m'_h = h(m_o) \tag{4}$$

- The result of the comparison of the hash value m_h calculated while creating the signature and m_h obtained during the verification of the signature is the result of the signature verification v

$$v \leftarrow \left(m'_h == m_h\right) \tag{5}$$

Similar to the signing algorithm, the signature verification algorithm can be written:

$$\begin{aligned} v &\leftarrow V(m_o, k_v) \\ &= \left((m_h = d(\sigma_v, k_v)) == \left(m'_h = h(m_o)\right)\right) \end{aligned} \tag{6}$$

Depending on the type of problems being solved, various mutations of the signature schemes are created. Reference [26] presents the sanitizable signature scheme and its implementation. This signature allows a defined third party (sanitizer) to change the signed document without invalidating the signature. Reference [25] categorizes signature schemes based on increased efficiency, increased security, anonymity of services, and extended possibilities of signing and verifying signatures. In particular, signature schemes are presented: batch scheme, forward-secure scheme, blind scheme or a scheme with a proxy. Reference [44] presents the RSA signature scheme in which, apart from the signature classes determined by the type of certificates, it focuses on cryptographic issues related to RSA, DSA and ElGamal keys. Reference [50] proposes the designated verifier signature (DVS) scheme to verify the signature of a message, preventing the transfer of the conviction to any third party. In turn, [51] discusses new types of signature schemes, such as the so-called BLT, based on the idea of combining one-time time keys with the time stamping service.

1) XML SIGNATURE SCHEME

The XML signature [37] conforms to the basic scheme shown above. The difference is the extension of the signature scheme, given the necessity of specifying the way of creating the signature. This extension concerns the applied algorithms and creating the signed content.

The XML signature σ_{XML} structure (syntax) consists of the specification (definition) of the signature (information about the signature, definition of the signature) D, the signature value σ_v, the information about the method of obtaining the verification key K and an optional object O containing any information (e.g., XAdES [4] extension or other signed contents):

$$\{D, \sigma_v, K, O\} \tag{7}$$

The signature specification D contains

- the indication of the method of canonicalization of the signature content $c(m)$,
- the indication of the signature algorithm $eh(m)$ consisting of the encryption algorithm $e(m)$ and the hash function algorithm $h(m)$

$$eh(m) = e(h(m)) \tag{8}$$

- the set of references $R = \{R_1, R_1, \ldots, R_n\}$ used to specify the method to retrieve the signed contents and the algorithm of the hash function. Each R_i reference contains:
  - the indication of the signed resource uri_i,
  - the optional definition of the transformation sequence $\mathbf{T_i} = \left\{T_{1_i}, T_{2_i}, \ldots, T_{n_i},\right\}$, and
  - the indication of the hash function algorithm (for the result of the reference resolution) $h_i(m)$.

The information content of the references can be shown as

$$R_i = \{uri_i, T_i, h_i(m)\} \tag{9}$$

The content of the XML signature definition (specification) has the following form:

$$D = \{c(m), eh(m), R\} \tag{10}$$

The definition of the signature D is the basis for the process of preparing the signature content P

$$m_{XML} \leftarrow P(D) \tag{11}$$

In regard to equation (2), the creation of the signature S can be written:

$$\sigma_{XML} \leftarrow S(M_{XML}, k_s) \tag{12}$$

Additionally, the signature verification V considers equation (6):

$$v \leftarrow V(m_{XML}, \sigma_v, k_v) \tag{13}$$

a: PREPARATION OF THE SIGNATURE CONTENT P

An XML signature requires that before creating or verifying a signature, a signature specification is defined by defining the elements of set D. Based on this set of data D, an algorithm for creating a signature is executed. It is an element that distinguishes an XML signature from other signature formats, where this information is an element of the format specification and does not allow for flexible modelling of the signature and adaptation to specific needs.

The process of XML signature preparation is presented below:

- For each reference R_i in the reference set R, the $r_i(uri_i)$ resource indicated by uri_i is retrieved, and the T_i transformation sequence is executed on this resource. The results of individual transactions are transferred to the subsequent transactions.

$$m_i = \sum_{j=1}^{n_i} T_j(r_i(uri_i)) \tag{14}$$

  This approach enables the partial content of the signature (signed content) m_i to be obtained.
- For each partial signature content m_i (derived from the R_i reference), the hash value m_{h_i} is calculated using the $h_i(m_i)$ algorithm

$$m_{h_i} = h_i(m_i) \tag{15}$$

- The information of individual references is extended by the value of the hash function of the resources indicated and transformed in the references. In this way, the D set, or more precisely the R references, are extended by the values of the m_{h_i} hash functions of the corresponding R_i references.

$$R_i = R_i \cup m_{h_i} \tag{16}$$

  After this operation, the content of the reference has the following form:

$$R_i = \left\{uri_i, T_i, h_i(m), m_{h_i} = h_i\left(m_{h_i}\right)\right\} \tag{17}$$

- The structure of set D is subject to canonicalization process $c(D)$ and is the content of the signature m_{XML}

$$m_{XML} = c(D) \tag{18}$$

  Finally, the preparation of the content of the signature P can be written as

$$m_{XML} \leftarrow P(D) = P(c(m), eh(m), R) \tag{19}$$

b: SIGNATURE ALGORITHM S

Considering formula (12) and the stage of preparing the signature content, the signature creation algorithm has the following form:

$$\begin{aligned}\sigma_{XML} \leftarrow S(m_{XML.}, k_s) &= S(m_{XML} \leftarrow P(D) \\ &= P(c(m), eh(m), R), k_s \in \{k_s, k_v\})\end{aligned} \tag{20}$$

Signature verification algorithm V

Considering formula 13, the signature verification algorithm has the following form:

$$v \leftarrow V\left(m'_{XML.}, \sigma_{XML}, k_v\right) \tag{21}$$

where m'_{XML} is the signature content obtained during verification according to formula (19) using specifications taken from the signature:

$$m'_{XML} \leftarrow P\left(D \in \sigma_{XML}\right) = P\left(c\left(m\right), eh\left(m\right), R\right) \tag{22}$$

The verification key $k_v \in K$ (K is the method of retrieval of the verification key, which is an element of the XML signature).

The algorithm for creating and verifying an XML signature at the stage of cryptographic processing of the signed content (integrity) is the same as that presented in the scheme described in paragraph (Signature scheme - signature signing and verification algorithm, formulae 1 - 5). The difference lies in the way of obtaining the signed content. In the basic scheme, the content of the signature is determined arbitrarily, and the method of its calculation originates from the specified specifications of the signature format. In the XML signature, the way of defining the content of the signature is flexible and defined in the specification (the definition of the signature is contained in the signature). Similarly, in the definition of the signature, cryptographic algorithms used in the signature (functions of the hash and encryption) and methods of canonicalization of the signed content are indicated. Such a way of processing XML signatures (creation and verification) is consistent with the schemes presented in the literature [27].

The XML signature scheme presented above is vulnerable to various threats, especially the XML signature wrapping attacks described in the "Related works" section. These threats result from the features of the XML document and the possibilities (benefits) provided by the XML signature. They are caused by the possibility of modelling the content of the signature on the one hand. On the other hand, they are caused by the possibility of modifying the content of the document in parts not covered by the signature, without the consequences of violating the integrity of the signature. Regardless of the solutions proposed by various authors to detect changes in the structure and content of XML documents, the authors propose to use secure signature references. Such references must indicate the appropriate content of the signature. Below is an example of a threat resulting from the lack of verification of the correctness of references in the signature.

To illustrate the problem, a sample implementation of signing and verifying the document in XML format was made. The document shown in Fig. 1 was taken as an example. The presented XML structure represents the order document and contains such information as data about the supplier and the ordering party (recipient), information about the time and place of delivery and payment details.

In this example, processing of the document is as follows:

- It is created by the customer in accordance with the agreed syntax, and then signed electronically by the recipient. The electronic signature is placed in the "`RecipientSignature`" element.
- Then, it is sent (using any communication protocol) to the supplier,
- Software supporting automatic processing of orders from the supplier verifies the signature of the ordering party and
- Generates a signature (electronic stamp with a signature structure) confirming the fact that the order has been accepted.
- The process of further order processing is launched.

```
-    <?xml version="1.0" encoding="UTF-8"?>
-    <Contract>
-         <GeneralInfo Id="GneralData">
-                   <Supplier Id="SupplierData">
-                             ...
-                   </Supplier>
-                   <Recipient Id="RecipientData">
-                             ...
-                   </Recipient>
-         </GeneralInfo>
-         <Delivery Id="DeliveryData">
-                             ...
-         </Delivery>
-         <Order Id="OrderData">
-                             ...
-         </Order>
-         <PaymentDetails>
-                   <Total>200000</Total>
-                   <FromAccount>
-                             <AccountNumber>1234-4342-0000-32342-
     3568-3532</AccountNumber>
-                   </FromAccount>
-                   <ToAccount>
-                             <AccountNumber>4532-0903-0032-0943-
     9546-3434</AccountNumber>
-                   </ToAccount>
-         </PaymentDetails>
-         <Authorisation>
-                   <RecipientSignature>
-                   </RecipientSignature>
-                   <SupplierSignature>
-                   </SupplierSignature>
-         </Authorisation>
-    </Contract>
```

FIGURE 1. Example of a document supporting the execution of a transaction, ready to be signed (`RecipientSignature`, `SupplierSignature`).

```
<RecipientSignature>
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
Id="RecipientSignatureId">
          <ds:SignedInfo>
                    ...
          </ds:SignedInfo>
          <ds:SignatureValueId="RecipientSignatureValueId">
                    K1cXlt...Ebg==
          </ds:SignatureValue>
          <ds:KeyInfo>
                    ...
          </ds:KeyInfo>
</ds:Signature>
</RecipientSignature>
```

FIGURE 2. Recipient signature in the document structure.

The presented order document is signed by the ordering party and the signature is placed in the structure of the order document ("`RecipientSignature`") as shown in Fig. 2.

The signature creation algorithm is executed according to the XML signature scheme (equation 12), where the signature content is created (equation 11) based on the D signature definition, which is one of the elements of the signature (equation 7).

```
VERIFICATION: 'resources/FinalContractRecipient.xml.xml'
Document is signed by 1 signatures.
VERYFYING SIGNATURE: 0
VERYFYING SIGNATURE: RecipientSignatureId
Core validity status : true
SignatureValue status: true
=================================================================================
Reference[0] Id: 'RecipientGeneralDataRef' validity status: true  URI: '#GneralData'
      Transform: 'http://www.w3.org/2000/09/xmldsig#enveloped-signature'
    Calc Digest: ccd825561da14588cc079437b45fe2c7291a19f79b3cd9f50cf6112a418e0805
Expected Digest: ccd825561da14588cc079437b45fe2c7291a19f79b3cd9f50cf6112a418e0805
=================================================================================
Reference[1] Id: 'RecipientDeliveryRef' validity status: true  URI: ''
      Transform: 'http://www.w3.org/2002/06/xmldsig-filter2'
    Calc Digest: c10389fb37e9348cf4d0cb069f9689771a6762c3437cbe41f6da184a8ad2aa5c
Expected Digest: c10389fb37e9348cf4d0cb069f9689771a6762c3437cbe41f6da184a8ad2aa5c
=================================================================================
Reference[2] Id: 'RecipientOrderRef' validity status: true  URI: ''
      Transform: 'http://www.w3.org/2002/06/xmldsig-filter2'
    Calc Digest: 2bc0aa81aed66ed7fe07aea39ab052a3ac436c7ad547b915d937c07df92583d8
Expected Digest: 2bc0aa81aed66ed7fe07aea39ab052a3ac436c7ad547b915d937c07df92583d8
=================================================================================
Reference[3] Id: 'RecipientPaymentRef' validity status: true  URI: ''
      Transform: 'http://www.w3.org/2002/06/xmldsig-filter2'
    Calc Digest: e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
Expected Digest: e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
=================================================================================
Signature passed core validation
```

FIGURE 3. Verification of the recipient signature (document unchanged) executed with the authors' software.

```
<SupplierSignature>
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
          Id="SupplierSignatureValueId">
          <ds:SignedInfo>
          ...
          </ds:SignedInfo>
          <ds:SignatureValue>TT..g==</ds:SignatureValue>
          <ds:KeyInfo>
                    <ds:KeyValue>
                              ...
                    </ds:KeyValue>
          </ds:KeyInfo>
</ds:Signature>
</SupplierSignature>
```

FIGURE 4. Supplier's signature.

```
<PaymentDetails>
          <Total>100000</Total>
          <FromAccount>
                    <AccountNumber>4532-0903-0032-0943-9546-
3434</AccountNumber>
          </FromAccount>
          <ToAccount>
                    <AccountNumber>1234-4342-0000-32342-3568-
3532</AccountNumber>
          </ToAccount>
</PaymentDetails>
```

FIGURE 5. Manipulated content of the document.

The signature created in this way will be positively verified by the supplier's system. Signature verification (equation 13) is based on the information recorded in the definition of signature D by preparing the content of the signature (equation 11). An example of a signature verification report is shown in Fig. 3.

After positive verification of the document by the supplier's system, it is confirmed by the supplier's signature (as an element in "`SupplierSignature`"), resulting in a commitment to the customer. The supplier's signature is shown in Fig. 4.

It should be noted that the process of preparing the signature content m_{xml} is executed in the $P(D)$ process based on the same set of information (signature specification) D, and in particular, the identical transformations T_i that constitute the reference definitions R_i (formula 14), which leads to a positive verification of the value of the hash function (equations 19 and 21).

In the meantime, the document has been manipulated by replacing account numbers (Fig. 2). The changed fragment of the document is shown in Fig. 5:

As a result of such actions, there is a conflict and two documents are in circulation: "Original" and "altered". The solution to the conflict should be verification of the signature in the document. Such verification was performed, as shown in Fig. 6:

The comparison of the results of verification of signatures presented in Fig. 3 and Fig. 6 shows that despite the manipulation of the document, in the part significant for the transaction, a valid verification result was obtained. Both the value of the signature and all the references were verified.

Fig. 7 shows the result of verification of both signatures in the document sent to the supplier (after generating the supplier's signature).

As seen, all signatures in both documents are formally valid and a positive result of their verification is obtained (as mentioned above, it is natural if the same transformations are applied to the signature references).

The cause of the described problem is an intentional or accidental incorrect specification of the reference to the "`PaymentDetails`" element in the order document (R_i reference in the signature definition D), as shown in Fig. 8 (element marked green).

```
VERIFICATION: 'resources/FinalContractRecipient-FALSED.xml'
Document is signed by 1 signatures.
VERYFYING SIGNATURE: 0
VERYFYING SIGNATURE: RecipientSignatureId
Core validity status : true
SignatureValue status: true
================================================================================
Reference[0] Id: 'RecipientGeneralDataRef' validity status: true  URI: '#GneralData'
      Transform: 'http://www.w3.org/2000/09/xmldsig#enveloped-signature'
    Calc Digest: ccd825561da14588cc079437b45fe2c7291a19f79b3cd9f50cf6112a418e0805
Expected Digest: ccd825561da14588cc079437b45fe2c7291a19f79b3cd9f50cf6112a418e0805
================================================================================
Reference[1] Id: 'RecipientDeliveryRef' validity status: true  URI: ''
      Transform: 'http://www.w3.org/2002/06/xmldsig-filter2'
    Calc Digest: c10389fb37e9348cf4d0cb069f9689771a6762c3437cbe41f6da184a8ad2aa5c
Expected Digest: c10389fb37e9348cf4d0cb069f9689771a6762c3437cbe41f6da184a8ad2aa5c
================================================================================
Reference[2] Id: 'RecipientOrderRef' validity status: true  URI: ''
      Transform: 'http://www.w3.org/2002/06/xmldsig-filter2'
    Calc Digest: 2bc0aa81aed66ed7fe07aea39ab052a3ac436c7ad547b915d937c07df92583d8
Expected Digest: 2bc0aa81aed66ed7fe07aea39ab052a3ac436c7ad547b915d937c07df92583d8
================================================================================
Reference[3] Id: 'RecipientPaymentRef' validity status: true  URI: ''
      Transform: 'http://www.w3.org/2002/06/xmldsig-filter2'
    Calc Digest: e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
Expected Digest: e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
================================================================================
Signature passed core validation
```

FIGURE 6. Verification of the recipient's signature (document unchanged) Executed with the authors' software.

```
VERIFICATION: 'resources/SupplierSignature'
Document is signed by 2 signatures.
VERYFYING SIGNATURE: 0
VERYFYING SIGNATURE: RecipientSignatureId
Core validity status : true
SignatureValue status: true
================================================================================
Reference[0] Id: 'RecipientGeneralDataRef' validity status: true  URI: '#GneralData'
      Transform: 'http://www.w3.org/2000/09/xmldsig#enveloped-signature'
    Calc Digest: ccd825561da14588cc079437b45fe2c7291a19f79b3cd9f50cf6112a418e0805
Expected Digest: ccd825561da14588cc079437b45fe2c7291a19f79b3cd9f50cf6112a418e0805
================================================================================
Reference[1] Id: 'RecipientDeliveryRef' validity status: true  URI: ''
      Transform: 'http://www.w3.org/2002/06/xmldsig-filter2'
    Calc Digest: c10389fb37e9348cf4d0cb069f9689771a6762c3437cbe41f6da184a8ad2aa5c
Expected Digest: c10389fb37e9348cf4d0cb069f9689771a6762c3437cbe41f6da184a8ad2aa5c
================================================================================
Reference[2] Id: 'RecipientOrderRef' validity status: true  URI: ''
      Transform: 'http://www.w3.org/2002/06/xmldsig-filter2'
    Calc Digest: 2bc0aa81aed66ed7fe07aea39ab052a3ac436c7ad547b915d937c07df92583d8
Expected Digest: 2bc0aa81aed66ed7fe07aea39ab052a3ac436c7ad547b915d937c07df92583d8
================================================================================
Reference[3] Id: 'RecipientPaymentRef' validity status: true  URI: ''
      Transform: 'http://www.w3.org/2002/06/xmldsig-filter2'
    Calc Digest: e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
Expected Digest: e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
================================================================================
Signature passed core validation
VERYFYING SIGNATURE: 1
VERYFYING SIGNATURE: SupplierSignatureId
Core validity status : true
SignatureValue status: true
================================================================================
Reference[0] Id: 'RecipientSignature' validity status: true  URI: ''
      Transform: 'http://www.w3.org/2002/06/xmldsig-filter2'
    Calc Digest: 94d26cb5c02d4e877a289cfe094fbfe603a00793349b9ca399d77879b2ec8fce
Expected Digest: 94d26cb5c02d4e877a289cfe094fbfe603a00793349b9ca399d77879b2ec8fce
================================================================================
Signature passed core validation
```

FIGURE 7. Verification of signatures in a document signed by the supplier and the recipient (document not changed).

```
<ds:Reference Id="RecipientPaymentRef" URI="">
          <ds:Transforms>
                    <ds:Transform
Algorithm="http://www.w3.org/2002/06/xmldsig-filter2">
                              <XPath
xmlns="http://www.w3.org/2002/06/xmldsig-filter2"
Filter="intersect">/Contract/PaymentDetail</XPath>
                    </ds:Transform>
          </ds:Transforms>
          <ds:DigestMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
          <ds:DigestValue>47DEQpj8HBSa+/TImW+5JCeuQeRkm5NMpJWZG3hS
uFU=</ds:DigestValue>
</ds:Reference>
```

FIGURE 8. **Incorrect reference specification in the recipient signature.**

```
<ds:Reference Id="r-id-1"
Type="http://www.w3.org/2000/09/xmldsig#Object" URI="#o-id-x">
          <ds:Transforms>
                    <ds:Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#base64"/>
          </ds:Transforms>
          <ds:DigestMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
          <ds:DigestValue>cy5VQ4V2A5zIb...8Hgow4KKeRZo=</ds:Digest
Value>
</ds:Reference>
```

FIGURE 9. **Erroneous (malicious) specification of a reference to an object in a signature in a JPK file.**

```
<ds:Reference Id="r-id-1" Type="" URI="">
          <ds:Transforms>
                    <ds:Transform
Algorithm="http://www.w3.org/TR/1999/REC-xpath-19991116">
                              <ds:XPath>not(ancestor-or-
self::ds:Signatures)</ds:XPath>
                    </ds:Transform>
                    <ds:Transform
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
          </ds:Transforms>
          <ds:DigestMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
          <ds:DigestValue>Hxu4zfNo1pVpYH4bPC8iWCHQ9cSRUAjnh9oT/+LP
y/s=</ds:DigestValue>
</ds:Reference>
```

FIGURE 10. **Dangerous specification of the XPath expression in the JPK file.**

Fig. 9 shows the manipulated specification of content in the reference to the signed object "`#o-id-x`" pointing to an incorrect object instead of the correct indication "`#o-id-x`".

The threat, an example of which is presented above, becomes very dangerous, especially in those documents in which the XML signature is used on a large scale. An example can be documents used in tax proceedings and controls (e.g., Standard Audit File for Tax - SAF-T used by several countries such as Portugal, Poland, etc. [52]).

There appears to be a real risk of manipulating the JPK file, caused by using the XPath expression to indicate the signed content if it is possible to provide the wrong XPath expression and sign other than expected content, as shown in Fig. 10, where the XPath expression is changed (correct "`ds:Signature`", incorrect "`ds:Signatures`").

The use of standard XML signature verification without considering the verification of the content defined by references does not protect the parties against misuse of the indication of incorrect signature content. Therefore, it is necessary to verify the references in the signature. It seems necessary to securely apply the definition of signature D by creating a signature and verifying the signature on the basis of the same specification D.

2) XML SIGNATURE SCHEME PROPOSAL WITH REFERENCE VERIFICATION

The use of XML signatures requires careful design that takes into account two aspects:

1. Protection against manipulation of the structure and content of the signed document (moving and changing the content) by using the means discussed in Related Work, (such as appropriate use of XPath [6], use of dedicated architecture and libraries [28], limiting the use of XPath and the Id attribute and proposing instead to sign entire SOAP structures [10]–[13], etc.).
2. Protection against intentional or accidental use of references and transformations in the signature, allowing the modification of the document through the use of a signature scheme with the verification of references.

The authors of the article suggest that the XML signature scheme, presented above, should be extended to include verification of references. The proposed approach is designed to eliminate most of the above described vulnerabilities. For this purpose, during the design and construction of systems supporting transaction execution, it is necessary to develop templates of signatures M for XML documents created (and verified) in each stage of the transaction. Such a template should contain all the elements necessary to create and verify a signature. It is equivalent in content to the signature definition D.

$$M \equiv D \tag{23}$$

Therefore, considering formula (10), we obtain it as

$$M = \{c(m), eh(m), R\} \tag{24}$$

Signature verification should, in this scheme, be extended to include verification of the correctness of the signature content that is the result of the description in template M with the result obtained using the signature specification D. For the avoidance of doubt during the implementation of the XML signature, the template M should be used to create the specification in the signature D and to verify the signature.

a: PREPARATION OF THE SIGNATURE CONTENT $\underline{P}$

Preparation of an XML signature is done in the same way as described above and involves processing in accordance with the D signature specification. The proposed scheme requires that the definition of the signature is created in accordance with the template M, in which the references to create the content of the signature are saved in accordance with the expectations of the parties.

b: SIGNATURE VERIFICATION ALGORITHM S

The signature creation process is analogous to that described in section 1.2.1. The only difference is the change in the description of formula 20:

$$\sigma_{XML} \leftarrow S\left(m_{XML} \leftarrow P\left(D \leftarrow M\right), k_s \in \{k_s, k_v\}\right) \tag{25}$$

As seen from the above formula, the content of the signature m_{XML} is derived from the definition (specification) of the signature D, which is derived from the signature template M.

Signature verification algorithm V

The signature verification in a schema with reference verification (signature content) is performed by:

- Calculation of the hash value by decrypting the signature value:

$$m_h = d\left(m_{XML} \leftarrow \sigma_{XML}, k_v \in \{k_v, k_s\}\right) \tag{26}$$

- Calculation of the hash value for the signed content according to the definition saved in the signature:

$$m'_h = h\left(m'_{XML} \leftarrow P\left(D \leftarrow \sigma_{XML}\right)\right) \tag{27}$$

- Calculation of the hash function value for the signed content according to the definition derived from the signature template:

$$m''_h = h\left(m''_{XML} \leftarrow P\left(M\right)\right) \tag{28}$$

- Comparison of the obtained hash values:

$$m_h == m'_h == m''_h \tag{29}$$

- A positive result of the comparison of these values proves the integrity and validity of the references used in the signature.
- It is possible, but not necessary, to compare the values of hash values of individual references taken from the signature, calculated from the definition of signature D and calculated from the definition derived from the template M:

$$m_{h_i} == m'_{h_i} == m''_{h_i} \tag{30}$$

  to check which reference does not match the established template.

The difference between the verification algorithm and the XML scheme is that the verification algorithm considers the values of the hash functions received on the basis of the signature template, apart from being obtained from the signature value and the definition saved in the signature.

The above proposal of the XML signature scheme extended by the verification of the signature reference is, according to the authors, resistant to vulnerability based on the intentional or accidental construction and use of references during the creation and signature verification. This vulnerability makes it possible to modify fragments of XML documents while maintaining the integrity of the signature value.

The structure (syntax) of the signature template and the presentation of the transaction execution with the verification of the signature, including the verification of references, is presented below

c: SECURITY MODEL

Generating a signature:

Fig. 11, above the dashed line, presents a diagram of creating the XML signature. The process consists of the following:

1. Creating a definition (specification) of a signature, which may be based on the signature template (reliable, secure information) or other unknown data,
2. Then, according to the definition created,
   a. the content of the signature is produced, which consists of the digest values of the contents indicated by all references, and
   b. the value of the signature is calculated by encrypting the digest value from the received content,
3. The definition and the signature value are placed in the XML signature structure.

The generated signature is formally always valid, but it can sign the inappropriate content.

It should be emphasized that it is not necessary to use a template when generating a signature, and the correct definition can also be created without a template.

Verification of the signature

In Fig. 11, below the dashed line, there is a diagram of XML signature verification. The verification path using the signature template is marked in green colour, and without the signature template in red.

The process consists of the following:

1. Creating a signature definition that can be undertaken by simply reading from an XML signature or from a template,
2. Then, according to the definition:
   a. the content of the signature is produced, which consists of the digest values of the contents indicated by all references, and
   b. the digest value of the resulting content is calculated.

   It should be noted that the resulting signature content is correct whenever the definition is based on a template. Otherwise, there is no such certainty.
3. Calculation of the digest value received by deciphering the signature value, which is read from the XML signature structure,
4. The validity of the verification is determined by the result of comparison of the digest value obtained from the definition, with the digest value obtained from the XML signature structure.

While the method of creating a definition (based on or without a template) is not important when generating a signature, it is important during verification.

If in this case (verification), the signature template is used to create the definition (green path), then the result of the presented verification is reliable. This result is observed because the signature is created for the same content as defined in the signature template.

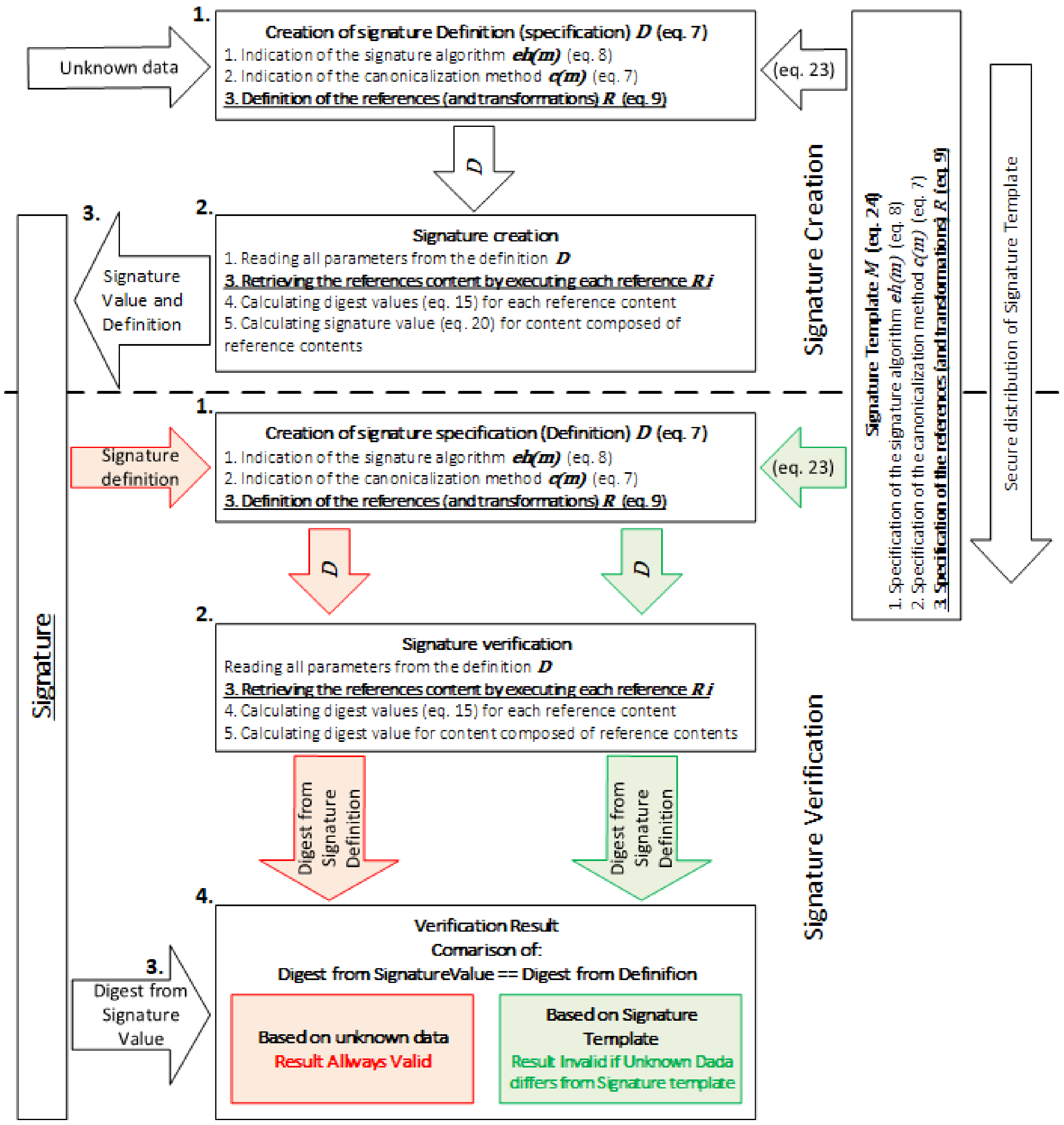

1.
Unknown data
Creation of signature Definition (specification) D (eq. 7)
1. Indication of the signature algorithm eh(m) (eq. 8)
2. Indication of the canonicalization method c(m) (eq. 7)
3. Definition of the references (and transformations) R (eq. 9)
(eq. 23)
D
2.
3.
Signature Value and Definition
Signature creation
1. Reading all parameters from the definition D
3. Retrieving the references content by executing each reference Ri
4. Calculating digest values (eq. 15) for each reference content
5. Calculating signature value (eq. 20) for content composed of reference contents
Signature Creation
Signature Template M (eq. 24)
1. Specification of the signature algorithm eh(m) (eq. 8)
2. Specification of the canonicalization method c(m) (eq. 7)
3. Specification of the references (and transformations) R (eq. 9)
Secure distribution of Signature Template
1.
Signature definition
Creation of signature specification (Definition) D (eq. 7)
1. Indication of the signature algorithm eh(m) (eq. 8)
2. Indication of the canonicalization method c(m) (eq. 7)
3. Definition of the references (and transformations) R (eq. 9)
(eq. 23)
D
D
Signature
2.
Signature verification
Reading all parameters from the definition D
3. Retrieving the references content by executing each reference Ri
4. Calculating digest values (eq. 15) for each reference content
5. Calculating digest value for content composed of reference contents
Digest from Signature Definition
Digest from Signature Definition
Signature Verification
4.
3.
Digest from Signature Value
Verification Result
Comarison of:
Digest from SignatureValue == Digest from Definifion
Based on unknown data
Result Allways Valid
Based on Signature Template
Result Invalid if Unknown Dada differs from Signature template

FIGURE 11. Signature security model with references verification.

Otherwise, if the definition taken from the XML signature structure is used for verification (red path), then the result of this verification will be positive even if the signature content does not match the content indicated in the template.

As seen from this example, it is possible to effectively detect the use of a signature with manipulated references. The basis is the introduction of an additional verification step - verification of the content of references based on a secure signature specification using the signature template.

3) PROPOSAL FOR THE STRUCTURE OF THE SIGNATURE TEMPLATE M

As shown above, the key element ensuring correct processing of XML signatures in accordance with the parties' agreement to the transaction is the signature template M. The authors proposed the structure (syntax) of the XML document presented in Fig. 12.

The structure of the signature template M contains all the information necessary for the creation and verification of the

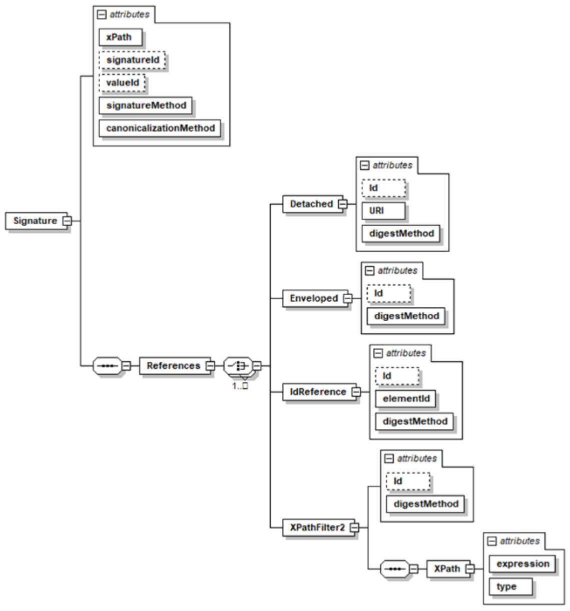


FIGURE 12. XML signature template syntax.

signature (equation 10), in particular

- the method of canonicalization $c(m)$,
- the name of the signature algorithm $eh(m)$ (name of the algorithm for calculating the hash value and the algorithm for encrypting this value),
- the way of preparing the content of the signature references with R_i and data necessary to build/verify various types of references - for example: detached, XPath, XPath filter2, enveloping,

as well as other information relating to, for example, the signature in a document,

- the signature position in the document (using the XPath expression),
- the optional name of the signature identifier and the identifier of the signature value.

a: IMPLEMENTATION AND TESTS

Based on the file containing the signature template M with the above presented syntax, it is possible to create content of individual references R_i independent of the D specification in the signature (i.e., secure) and to perform calculation of the hashes from these contents and comparison of the hashes received based on the template with the hashes calculated as standard and placed in the signature. (equations 19 and 30). For the example shown above, the signature template for the ordering party was developed, which is shown in Fig. 13.

The same solution can be applied for Standard Audit File for Tax (SAF-T), whose sample signature template is shown in Fig. 14.

Using the data stored in the signature template, an extended (apart from the standard, described above) signature verification is performed, i.e., verification of hash values written in the signature with the hash values obtained on the basis of the description of references in the template.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<SignatureTemplate xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xsi:noNamespaceSchemaLocation="...\SignatureTemplate.xsd">
<Signature canonicalizationMethod="INCLUSIVE_WITH_COMMENTS"
signatureId="RecipientSignatureValueId"
signatureMethod="RSA_SHA256"
valueId="RecipientSignatureValueId"
xPath="/Contract/Authorisation/RecipientSignature">

<References>
          <IdReference Id="RecipientGeneralDataRef"
digestMethod="SHA256"
elementId="GneralData"/>
          <Detached Id="DetachedRefId"
URI="http://www.w3.org/TR/xml-stylesheet"
digestMethod="SHA256"/>
          <XPathFilter2 Id="RecipientDeliveryRef"
digestMethod="SHA256">
                    <XPath expression="/Contract/Delivery"
type="intersect"/>
          </XPathFilter2>
          <XPathFilter2 Id="RecipientOrderRef"
digestMethod="SHA256">
                    <XPath expression="/Contract/Order"
type="intersect"/>
          </XPathFilter2>
          <XPathFilter2 Id="RecipientPaymentRef"
digestMethod="SHA256">
                    <XPath expression="/Contract/PaymentDetails"
type="intersect"/>
          </XPathFilter2>
</References>

</Signature>
<Authorization>
          <ds:Signature
xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
Id="AuthorizationSignature">
                    <ds:SignedInfo/>

          <ds:SignatureValue>aYDKJ1+YJ6Q==</ds:SignatureValue>
                    <ds:KeyInfo/>
          </ds:Signature>
</Authorization>
</SignatureTemplate>
```

FIGURE 13. Example of XML signature template.

```
<SignatureTemplate xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="J:\XMLDSig-
workspace\XMLSignature\resources\SignatureTemplate.xsd">
<Signature signatureId="value-id-f80fcb47a1716069ca994e2132c0e12d"
xPath="//" canonicalizationMethod="INCLUSIVE_WITH_COMMENTS"
signatureMethod="RSA_SHA256" valueId="value-id-
f80fcb47a1716069ca994e2132c0e12d">
          <References>
                    <IdReference digestMethod="SHA256"
elementId="o-id-1" Id="r-id-1"
type="http://www.w3.org/2000/09/xmldsig#Object">
                              <Transform>BASE_64</Transform>
                    </IdReference>
                              ...
          </References>
          </Signature>
          <Authorization>
          </Authorization>
</SignatureTemplate>
```

FIGURE 14. Indication of the signature reference in the JPK file.

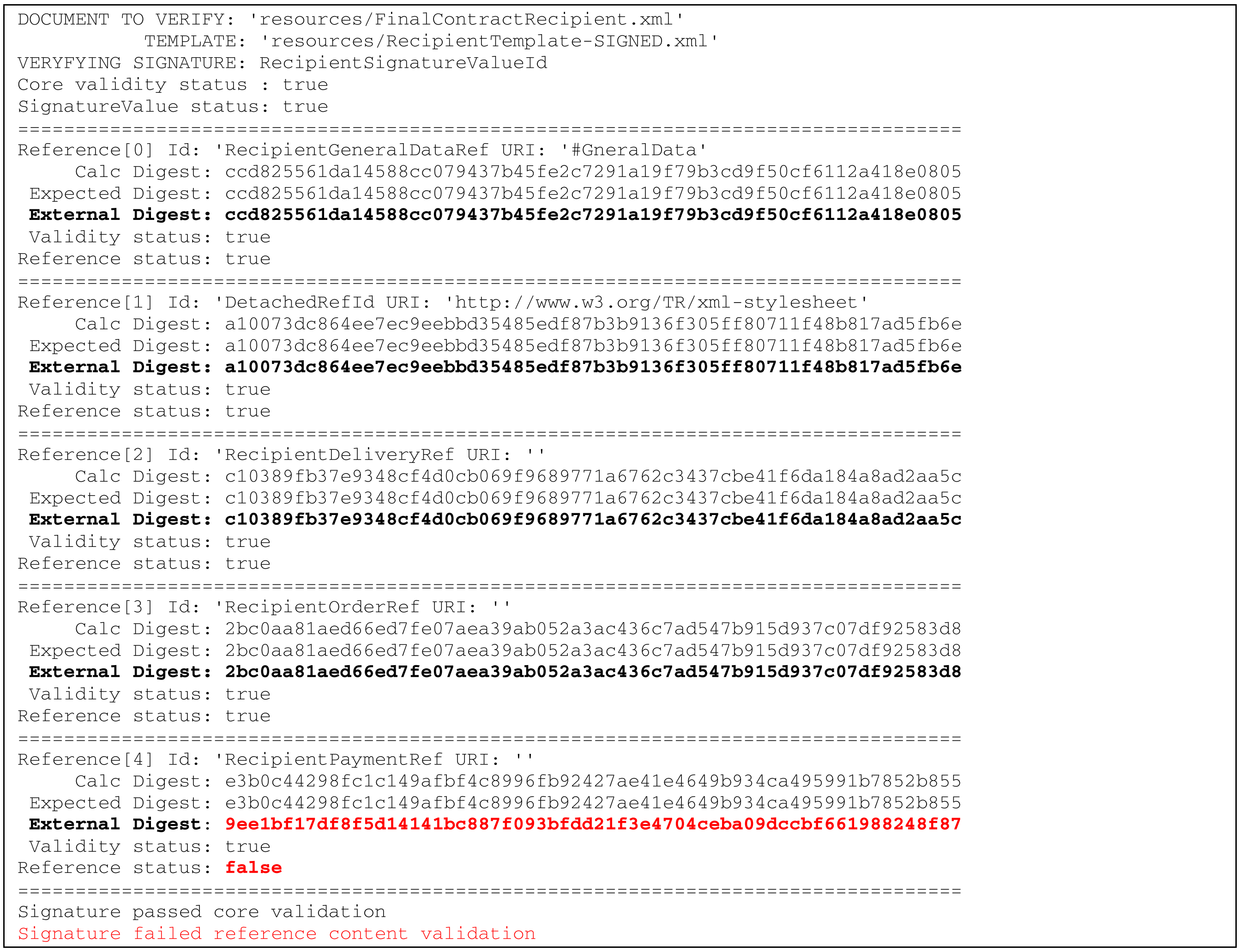
Fig. 15 shows the result of the signature verification. As seen from the drawing, the result of the verification additionally contains:

- values of hashes m'_h calculated on the basis of references in the signature (“CALC”),
- values of hashes m_h placed in the signature (“EXPECTED”),
- values of hashes m''_h calculated on the basis of specifications in the signature template (“EXTERNAL”),

which are the basis for “traditional” and “extended – proposed” verification of integrity of XML signatures.

The result of such verification, for a signature with manipulated “RecipientPayment” reference, is shown in Fig. 15.

As seen from this example, it is possible to effectively detect the use of a signature with manipulated references.



```
DOCUMENT TO VERIFY: 'resources/FinalContractRecipient.xml'
          TEMPLATE: 'resources/RecipientTemplate-SIGNED.xml'
VERYFYING SIGNATURE: RecipientSignatureValueId
Core validity status : true
SignatureValue status: true
=================================================================================
Reference[0] Id: 'RecipientGeneralDataRef URI: '#GneralData'
     Calc Digest: ccd825561da14588cc079437b45fe2c7291a19f79b3cd9f50cf6112a418e0805
 Expected Digest: ccd825561da14588cc079437b45fe2c7291a19f79b3cd9f50cf6112a418e0805
 External Digest: ccd825561da14588cc079437b45fe2c7291a19f79b3cd9f50cf6112a418e0805
 Validity status: true
Reference status: true
=================================================================================
Reference[1] Id: 'DetachedRefId URI: 'http://www.w3.org/TR/xml-stylesheet'
     Calc Digest: a10073dc864ee7ec9eebbd35485edf87b3b9136f305ff80711f48b817ad5fb6e
 Expected Digest: a10073dc864ee7ec9eebbd35485edf87b3b9136f305ff80711f48b817ad5fb6e
 External Digest: a10073dc864ee7ec9eebbd35485edf87b3b9136f305ff80711f48b817ad5fb6e
 Validity status: true
Reference status: true
=================================================================================
Reference[2] Id: 'RecipientDeliveryRef URI: ''
     Calc Digest: c10389fb37e9348cf4d0cb069f9689771a6762c3437cbe41f6da184a8ad2aa5c
 Expected Digest: c10389fb37e9348cf4d0cb069f9689771a6762c3437cbe41f6da184a8ad2aa5c
 External Digest: c10389fb37e9348cf4d0cb069f9689771a6762c3437cbe41f6da184a8ad2aa5c
 Validity status: true
Reference status: true
=================================================================================
Reference[3] Id: 'RecipientOrderRef URI: ''
     Calc Digest: 2bc0aa81aed66ed7fe07aea39ab052a3ac436c7ad547b915d937c07df92583d8
 Expected Digest: 2bc0aa81aed66ed7fe07aea39ab052a3ac436c7ad547b915d937c07df92583d8
 External Digest: 2bc0aa81aed66ed7fe07aea39ab052a3ac436c7ad547b915d937c07df92583d8
 Validity status: true
Reference status: true
=================================================================================
Reference[4] Id: 'RecipientPaymentRef URI: ''
     Calc Digest: e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
 Expected Digest: e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
 External Digest: 9ee1bf17df8f5d14141bc887f093bfdd21f3e4704ceba09dccbf661988248f87
 Validity status: true
Reference status: false
=================================================================================
Signature passed core validation
Signature failed reference content validation
```

FIGURE 15. Result of signature verification executed with the use of the signature template including verification of reference content.

The basis is the introduction of an additional verification step - verification of the content of references based on a secure signature specification using the signature template.

b: PERFORMANCE

The performance of the process of creating and verifying the signature in both the traditional scheme and with reference verification is equal. This is because the same methods and algorithms with the same data processing are used in both cases. The only difference is the source of information for building signature specifications (definitions). It is either the "`SignedInfo`" element in the signature in the traditional case or the "Signature Template" in the case of providing the correctness of the content (reference) of the signature.

The difference occurs when individual references need to be verified when the content of the signature (as a whole) is found to be incorrect. In this case, it is necessary to obtain the content of individual references and calculate their digest function values both from the definition of the signature received from "`SignedInfo`" and the signature template. In both cases, the same data are processed using the same methods and algorithms, which leads to a doubling of verification time.

IV. DISCUSSION

XML format [15] is a powerful tool for creating and processing documents in a simple, standardized way. This tool enables the implementation of various documents regardless of the area of application and technological processing platform. The XML signature [37], based on XML format, is a considerable extension of the possibility of electronic signature in relation to other formats [1], [16], [17], [38]. Depending on the needs, this signature can be in different relations with the signed document (enveloping, enveloped, and detached). It can sign different fragments of the XML document, and many signatures signing different contents can be placed in the XML document. The features of the document and the XML signature provide a very wide range of possibilities of electronic support for transaction execution in a dispersed heterogeneous environment with ensuring security in terms of integrity, authenticity and non-repudiation of the origin of various parts of documents.

On the other hand, the use of a signature and a document in XML format requires careful design of both the structure (syntax) of the document and the specifications of the signature. This is due to the identified threats classified by [Hill] and considered in the literature, starting with [6] as the so-called XML signature wrapping. Although these papers describe problems with regard to the SOAP protocol, their background is more general. These outcomes result from the possibility of signing XML document fragments, which ensures integrity in signed document nodes by bypassing other nodes. It should be emphasized that between the signed document and the signature there is a link expressed by references in the signature, indicating the signed content. Hence, the necessity to

- protect against XSW attacks (using the possibility of moving the signed content) but also
- protect against intentional or accidental misuse of references and transformations in the signature, which can lead to the same effects as XSW.

There are solutions indicated in the literature to counteract XSW frauds. Most of them are dedicated to the SOAP protocol, but some of the proposed mechanisms can be used outside SOAP solutions and are related to the general use and signing of XML documents. The authors of the article notice and focus on the second aspect of risks associated with the definition and processing of the signature and XML document, which is the problem of secure use of references in the XML signature during both the creation and verification of the signature. The presented proposal is not a proposal to replace the methods of fighting XSW attacks; rather, the rationale is to supplement the security measures in the area of the secure definition of signed content. According to the authors, the basis of both threats is the same (relations between the signature and the signed document) and the essence of the difference is protection against displacement and injection of content on the one hand, and on the other hand, the safe use of references in the signature (safe specification of the signature content).

The authors, without questioning the solutions presented in the literature, propose to focus on the correct and secure definition of the signature content and to verify signatures with the certainty that the signed content is consistent with the legal intentions of the parties using the document. To this end, the verification of the signature should be extended by checking the signature content using the so-called signature template. This template contains reliable and secure information necessary for creating specifications (“`SignedInfo`”) and verifying the signature (“`SignatureValue`”). This approach:

- extends the security area to secure signature references (signature content) in a way that ensures secure, reliable and automatic processing of documents,
- can be applied to any XML documents, not limited to SOAP messages, as it is the case with solutions presented by [6], [34], [40], [42] or [46],
- does not impose limitations on the scope of XML signature use such as the XPath expression for signature reference specification as proposed in [34],
- does not introduce new information in comparison with current standards such as the node counting proposed in [40] or the “position token” in the article [46],
- does not require encryption of signed content as described in [42], and
- the solution proposed by the authors uses standard algorithms and functions that are commonly used in the processing of XML signatures.

In the authors’ opinion, the use of trusted certificates in XML signatures, by using the “`KeyInfo`” element or extended XAdES signature structures, effectively prevents man-in-the-middle attacks. The use of a signature in XAdES format also provides the possibility of the standardized use of time stamps, which effectively protects the signature against repetition attacks.

As the risks associated with XML signatures may hamper the full use of XML signatures, further work should be done on secure use of XML signatures, treating their features as a benefit and focusing on counteracting these risks (e.g., denial of service). Regardless of the detailed considerations presented, the concern to ensure an adequate level of safety should also apply to the general architecture and the entire life cycle of the solution [23]. This security can be achieved using various techniques, such as reactive and proactive approaches, checklists, security requirements gap analysis or architectural threat analysis.

V. CONCLUSION

Naive (careless) use of an XML signature may lead to the emergence of vulnerabilities based on the possibility of modifying parts of the document despite the use of the XML signature. The article refers to the proposals published in the literature and, foremost, relates to the protection against the so-called XSW. The authors pointed out the vulnerability associated with the lack of verification of the validity of the reference in the signature, which may lead to similar effects as those encountered with XSW. A solution was also proposed involving the use of the so-called signature template containing information on how to create a signature, and thus providing the basis for the creation and verification of XML signatures. The syntax of such a template was also presented, and a positive test of the solution’s functionality was conducted.

In specific applications, the security level is determined by two elements in combination: specification (XML signature definition – “`SignedInfo`”) and the structure of the signed document. It seems reasonable to conduct further work to extend the signature template with aspects related to the structure and perhaps even processing of the document based on secure information. Such information should be created during the development of a solution supporting a specific transaction execution and delivered to parties implementing IT support for transaction execution.

According to the authors, XML documents and signatures are a powerful tool supporting the implementation of various transactions regardless of the areas of application and technologies used. The threats identified and presented in the article do not disqualify electronically signed XML documents. These threats are caused by the positive features of the XML signature, and it is more important to ensure the safe use of these features than to limit or end the use of this signature.

REFERENCES

[1] *PKCS#7: Cryptographic Message Syntax Version 1.5, IETF 1998.* document RFC 2315, Accessed: Dec. 1, 2019. [Online]. Available: https://tools.ietf.org/html/rfc2315

[2] G. Karlinger, "Application according to the international standard XML Signature Syntaa, and Processing," in *Proc. 5th Joint Working Conf. Commun. Multimedia Secur.*, 2013, pp. 15, doi: 10.1007/978-0-387-35413-2.

[3] A. Selkirk, "XML and security," in *Journal BT Technology Journal archive*, vol. 19, no. 3. Norwell, MA, USA: Kluwer, Jul. 2001, pp. 23–34, doi: 10.1023/A:1011977913258.

[4] (Feb. 2003). *XML Advanced Electronic Signatures (XAdES) W3C Note 20*. Accessed: Dec. 1, 2019. [Online]. Available: https://www.w3.org/TR/XAdES/

[5] W. Kubbilun, S. Gajek, M. Psarros, and J. Schwenk, "Trustworthy verification and visualisation of multiple XML-signatures," in *Communications and Multimedia Security. CMS* (Lecture Notes in Computer Science), vol. 3677, J. Dittmann, S. Katzenbeisser, and A. Uhl, Eds. Berlin, Germany: Springer, 2005. [Online]. Available: https://doi.com/10.1007/11552055_41

[6] M. McIntosh and P. Austel, "XML signature element wrapping attacks and countermeasures," in *Proc. workshop Secure Web services*, Fairfax, VA, USA, Nov. 2005, pp. 20–27.

[7] R. Richards, "XML security," in *Pro PHP XML and Web Services*. Berkeley, CA, USA: Apress, 2006, doi: 10.1007/978-1-4302-0139-7_12.

[8] C. A. Ardagna, E. Damiani, S. De Capitani di Vimercati, and P. Samarati, "XML security," in *Security, Privacy, and Trust in Modern Data Management. Data-Centric Systems and Applications*, M. Petković and W. Jonker, Eds. Berlin, Germany: Springer, 2007, doi: 10.1007/978-3-540-69861-6_6.

[9] B. Hill. Taxonomy of Attacks Against XML Digital Signature & Encryption, Command Injection in XML Signatures and Encryption. iSEC Partners, 2007. Accessed: Dec. 1, 2019. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-64719-7_22

[10] (Apr. 27, 2007). *SOAP W3C Recommendations—SOAP Version 1.2 Part 0: Primer, W3C Recommendation*, 2nd ed. Accessed: Dec. 1, 2019. [Online]. Available: http://www.w3.org/TR/2007/REC-soap12-part0-20070427/

[11] (Apr. 27, 2007). *Part 1: Messaging Framework, W3C Recommendation*, 2nd ed. Accessed: Dec. 1, 2019. http://www.w3.org/TR/2007/REC-soap12-part1-20070427/

[12] (Apr. 27, 2007). *Part 2: Adjuncts, W3C Recommendation*, 2nd ed. Accessed: Dec. 1, 2019. [Online]. Available: http://www.w3.org/TR/2007/REC-soap12-part2–20070427/

[13] (Apr. 27, 2007). *Specification Assertions and Test Collection, W3C Recommendation*, 2nd ed. Accessed: Dec. 1, 2019. [Online]. Available: https://www.w3.org/TR/2007/REC-soap12-testcollection-20070427/

[14] T. Knap and I. Mlýnková, "Towards more secure Web services-exploiting and analysing xml signature security issues," in *Proc. 3rd Int. Conf. Res. Challenges Inf. Sci.*, Apr. 2009, pp. 49–58, doi: 10.1109/RCIS.2009.5089267.

[15] (Nov. 2018). *Extensible Markup Language (XML) 1.0, W3C Recommendation*. Accessed: Dec. 1, 2019. [Online]. Available: http://www.w3.org/TR/xml/

[16] *Cryptographic Message Syntax (CMS), IETF 2009.* document RFC 5652, Accessed: Dec. 1, 2019. [Online]. Available: https://tools.ietf.org/html/rfc5652

[17] *Technical Specification; Electronic Signatures and Infrastructures (ESI); PDF Advanced Electronic Signature Profiles; Part 1: PAdES Overview—A Framework Document for PAdES*, document ETSI TS 102 778-1 V1.1.1, Jul. 2009. Accessed: Feb. 19, 2020. [Online]. Available: https://www.etsi.org/deliver/etsi_ts/102700_102799/10277801/01.01.01_60/ts_10277801v010101p.pdf

[18] *Technical Specification; Electronic Signatures and Infrastructures (ESI); PDF Advanced Electronic Signature Profiles; Part 2: PAdES Basic—Profile Based on ISO 32000-1*, document ETSI TS 102 778-2 V1.2.1, Jul. 2009. Accessed: Feb. 19, 2020. [Online]. Available: https://www.etsi.org/deliver/etsi_ts/102700_102799/10277802/01.02.01_60/ts_10277802v010201p.pdf

[19] *Technical Specification; Electronic Signatures and Infrastructures (ESI); PDF Advanced Electronic Signature Profiles; Part 3: PAdES Enhanced—PAdES-BES and PAdES-EPES Profiles*, document ETSI TS 102 778-3 V1.2.1, Jul. 2010. Accessed: Feb. 19, 2020. [Online]. Available: https://www.etsi.org/deliver/etsi_ts/102700_102799/10277803/01.02.01_60/ts_10277803v010201p.pdf

[20] *Technical Specification; Electronic Signatures and Infrastructures (ESI); PDF Advanced Electronic Signature Profiles; Part 4: PAdES Long Term—PAdES-LTV Profile*, document ETSI TS 102 778-4 V1.1.2, Dec. 2009. Accessed: Feb. 19, 2020. [Online]. Available: https://www.etsi.org/deliver/etsi_ts/102700_102799/10277804/01.01.02_60/ts_10277804v010102p.pdf

[21] *Technical Specification; Electronic Signatures and Infrastructures (ESI); PDF Advanced Electronic Signature Profiles; Part 5: PAdES for XML Content—Profiles for XAdES Signatures*, document ETSI TS 102 778-5 V1.1.2, Dec. 2009. Accessed: Feb. 19, 2020. [Online]. Available: https://www.etsi.org/deliver/etsi_ts/102700_102799/10277805/01.01.02_60/ts_10277805v010102p.pdf

[22] J. Katz, "Digital signatures: Background and definitions," in *Digital Signatures*. Boston, MA, USA: Springer, 2010, doi: 10.1007/978-0-387-27712-7_1.

[23] J. Diamant, "Resilient security architecture: A complementary approach to reducing vulnerabilities," *IEEE Secur. Privacy*, vol. 9, no. 4, pp. 80–84, Jul./Aug. 2011.

[24] M. Jensen, C. Meyer, J. Somorovsky, and J. Schwenk, "On the effectiveness of XML schema validation for countering XML signature wrapping attacks," in *Proc. 1st Int. Workshop Securing Services Cloud (IWSSC)*, Sep. 2011, pp. 6–8, doi: 10.1109/IWSSCloud.2011.6049019.

[25] M. Alidoost Nia, A. Sajedi, and A. Jamshidpey, "An introduction to digital signature schemes," 2014, *arXiv:1404.2820*. [Online]. Available: http://arxiv.org/abs/1404.2820

[26] H. C. Pöhls, K. Samelin, and J. Posegga, "Sanitizable signatures in XML signature—Performance, mixing properties, and revisiting the property of transparency," in *Applied Cryptography and Network Security* (Lecture Notes in Computer Science), vol. 6715, J. Lopez and G. Tsudik Eds. Berlin, Germany: Springer, 2011,

[27] S. Gerić and T. Vidačić, "XML digital signature and its role in information system security," in *Proc. 35th Int. Conv. (MIPRO)*, Opatija, Croatia, May 2012, pp. 21–25.

[28] C. Mainka, M. Jensen, I. L. Lo, and J. Schwenk, "Making xml signatures immune to xml signature wrapping attacks," in *Cloud Computing and Services Science* (Communications in Computer and Information Science), vol. 367, I. I. Ivanov, M. van Sinderen, F. Leymann, and T. Shan, Eds. Cham, Switzerland: Springer, 2013.

[29] S. Li and F. You, "Research on XML digital signature application in the log protection of typesetting on Web," in *Proc. Int. Conf. Control Eng. Commun. Technol.*, Dec. 2012, pp. 7–9, doi: 10.1109/ICCECT.2012.15.

[30] (Apr. 2012). *W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures, W3C Recommendation*. Accessed: Dec. 1, 2019. [Online]. Available: http://www.w3.org/TR/xmlschema11–1/

[31] (Apr. 2012). *W3C XML Schema Definition Language (XSD) 1.1 Part 2: Datatypes, W3C Recommendation 5*. Accessed: Dec. 1, 2019. [Online]. Available: http://www.w3.org/TR/xmlschema11–2/

[32] Y. Chen, W. Guo, and X. Zhao, "Study of XML digital signature for resource document fragment," in *Proc. 2nd Int. Conf. Inf.Sci. Eng.*, Dec. 2010, pp. 1541–1544, doi: 10.1109/ICISE.2010.5690206.

[33] A. A. A. El-Aziz and A. Kannan, "A comprehensive presentation to XML signature and encryption," in *Proc. Int. Conf. Recent Trends Inf. Technol. (ICRTIT)*, Jul. 2013, doi: 10.1109/ICRTIT.2013.6844276.

[34] H. R. Kouchaksaraei and A. Chefranov, "Countering wrapping attack on XML signature in SOAP message for cloud computing," *Int. J. Comput. Sci. Inf. Secur.*, vol. 11, no. 9, pp. 1–6, Sep. 2013.

[35] R. A. Saravanaguru, G. Abraham, K. Ventakasubramanian, and K. Borasia, "Securing Web services using XML signature and XML encryption," pp. 1–6, Mar. 2013, *arXiv:1303.0910*. [Online]. Available: https://arxiv.org/abs/1303.0910

[36] R. K. Saravanaguru, G. Abraham, K. Ventakasubramanian, and K. Borasia, "Securing Web services using XML signature and XML encryption," 2013, *arXiv:1303.0910*. [Online]. Available: http://arxiv.org/abs/1303.0910

[37] (Apr. 2013). *XML Signature Syntax and Processing Version 1.1, W3C Recommendation*. Accessed: Dec. 1, 2019. [Online]. Available: https://www.w3.org/TR/xmldsig-core1/

[38] *Electronic Signatures an Infractructures (ESI); CMS Advanced Electronic Signatures (CAdES)*, document ETSI TS 101 733 v2.2.1, 2013.

[39] F. Buccafurri, L. Fotia, and G. Lax, ''Implementing advanced electronic signature by public digital identity system (SPID),'' in *Electronic Government and the Information Systems Perspective* (Lecture Notes in Computer Science), vol. 9831, A. Kö and E. Francesconi, Eds. Cham, Switzerland: Springer, 2016, pp. 289–303, doi: 10.1007/978-3-319-44159-7_21.

[40] A. N. Gupta and P. S. Thilagam, ''Detection of XML signature wrapping attack using node counting,'' in *Proc. 3rd Int. Symp. Big Data Cloud Comput. Challenges*, vol. 49, V. Vijayakumar and V. Neelanarayanan, Eds. Cham, Switzerland: Springer, 2016, pp. 57–63, doi: 10.1007/978-3-319-30348-2_5.

[41] J. Ravi and B. Balusamy, ''Provision of XML security in E-Commerce applications with XML digital signatures using virtual smart card,'' in *Proc. 1st Int. Conf. Inf. Commun. Technol. Intell. Syst.* (Smart Innovation, Systems and Technologies), vol. 2, S. C. Satapathy and S. Das, Eds. Cham, Switzerland: Springer, 2016, pp. 411–421, doi: 10.1007/978-3-319-30927-9_40.

[42] A. Razaque, A. Trivedi, M. Joshi, M. Kurakula, and S. Srungaram, ''Novel signature verification process,'' in *Proc. IEEE Long Island Syst., Appl. Technol. Conf. (LISAT)*, Apr. 2016, pp. 1–5.

[43] A. Alshammari, S. Alhaidari, A. Alharbi, and M. Zohdy, ''Security threats and challenges in cloud computing,'' in *Proc. IEEE 4th Int. Conf. Cyber Secur. Cloud Comput. (CSCloud)*, Jun. 2017, pp. 26–28, doi: 10.1109/CSCloud.2017.59.

[44] R. Seetha, ''An enhanced digital signature scheme,'' *Int. J. Appl. Eng. Res.*, vol. 12, no. 22, pp. 11878–11884, 2017. [Online]. Available: http://www.ripublication.com

[45] (Mar. 2017). *XML Path Language (XPath) 3.1, W3C Recommendation*. Accessed: Dec. 1, 2019. [Online]. Available: https://www.w3.org/TR/xpath-31/

[46] J. Kumar, B. Rajendran, B. S. Bindhumadhava, and N. S. Chandra Babu, ''XML wrapping attack mitigation using positional token,'' in *Proc. Int. Conf. Public Key Infrastruct. Appl. (PKIA)*, Nov. 2017, pp. 36–42, doi: 10.1109/PKIA.2017.8278958.

[47] S. Pawar and N. N. Chiplunkar, ''Open source apis for processing the XML result of Web services,'' in *Proc. Int. Conf. Adv. Comput., Commun. Inform. (ICACCI)*, Sep. 2017, pp. 13–16, doi: 10.1109/ICACCI.2017.8126114.

[48] M. Liu and Y. Li, ''An Implementation of a XML Documents Secure Circulation System,'' in *Proc. J. Phys., Conf.*, vol. 1060, 2018, Art. no. 012004, doi: 10.1088/1742–6596/1060/1/012004.

[49] K. Seol, Y.-G. Kim, E. Lee, Y.-D. Seo, and D.-K. Baik, ''Privacy-preserving attribute-based access control model for XML-based electronic health record system,'' *IEEE Access*, vol. 6, pp. 9114–9128, Feb. 2018.

[50] P. Thanalakshmi and R. Anitha, ''A new code-based designated verifier signature scheme,'' *Int. J. Commun. Syst.*, vol. 31, no. 17, 2018, Art. no. e3803.

[51] A. Buldas, D. Firsov, R. Laanoja, H. Lakk, and A. Truu, ''A new approach to constructing digital signature schemes,'' in *Advances in Information and Computer Security* (Lecture Notes in Computer Science), vol. 11689, N. Attrapadung and T. Yagi, Eds. Cham, Switzerland: Springer, 2019. [Online]. Available: https://iopscience.iop.org/article/10.1088/1742-6596/1060/1/012004

[52] *Jednolity Plik Kontrolny*. Accessed: Jan. 1, 2020. [Online]. Available: https://www.podatki.gov.pl/jednolity-plik-kontrolny/

GERARD WAWRZYNIAK was born in Poland, in 1963. He received the M.Sc.E. degree in naval architecture and marine engineering from the Shipbuilding Institute, Technical University of Szczecin, Poland, in 1989, and the M.Sc. degree in applied informatics from the School of Computer Science, De Montfort University, Leicester, U.K., in 1995.

He is currently an Assistant with the Faculty of Computer Science and Information Technology, West Pomeranian University of Technology, Szczecin. His main areas of research interests are in document formats, XML documents, document processing, digital signatures, the secure Internet transactions, and digital support for transactions.

IMED EL FRAY received the M.Sc. degree in naval automation from the Szczecin University of Technology, the Ph.D. degree from the Faculty of Maritime Technology, and the D.Sc. degree from the Faculty of Computer Science. He is currently a Professor with the West Pomeranian University of Technology (ZUT), Szczecin, Poland. He is the author or coauthor of four book chapters and over 80 research articles on risk assessment and risk management, information systems audit, security information and event management, and common criteria.