SECURITY IN SENDING AND STORAGE OF PETRI NETS BY SIGNING AND ENCRIPTION

Íñigo León Samaniego[(a)], Mercedes Pérez de la Parte[(b)], Eduardo Martínez Cámara[(b)], Juan Carlos Sáenz-Díez Muro[(a)]

[(a)] University of La Rioja. Industrial Engineering Technical School. Department of Electrical Engineering. Logroño, Spain
[(b)] University of La Rioja. Industrial Engineering Technical School. Department of Mechanical Engineering. Logroño, Spain

inigo.leon@gmail.com, mercedes.perez@unirioja.es, eduardo.martinezc@unirioja.es, juan-carlos.saenz-diez@unirioja.es

ABSTRACT

The aim of this paper is double. On the one hand, to provide a standard way to hide all or part of a Petri net that could contain sensitive information, such as a company that represents a secret production process through Petri nets (privacy). On the other hand also as standard ensure that Petri net has not been altered (integrity) and that who sends or firm that Petri net is who he say he is (non-repudiation).

To ensure the privacy of an entire Petri net (or a part of it) the best solution is not to prevent access to such information, such as hiding in a safe or behind a firewall, but encrypt that information, even being to view. Today it is easier to open a safe or circumvent a firewall than to break an encryption standard algorithm (which, incidentally, is impossible nowadays).

As for the integrity and non-repudiation, the solution again is not to deliver the Petri net 'in hand' to avoid disruptions and to know who delivers it (since we are in the Internet age). The solution is to digitally sign all or part of a Petri net so that reliably to know who has performed the firm, and be able to detect any unauthorized modification of any of the signed data.

The aim of this paper is to show how to encrypt the selected part of the graph and to sign the Petri net, so that the obtained file compliances with the desired signature and encryption. So, in this final file, all the information (and only that) referred to the shaded part is encrypted and will not be interpretable. In particular, anything will be know about the nodes p1 and p2 or transitions t1 and t3: their constitute a secret process. In addition, this file will contain additional information that will verify the integrity of the file to prevent anyone to modify and information about who has signed this Petri net. The solution we propose is to use PNML representation of Petri nets and XMLEncryption standards for encryption and for signing XMLSignature.

Keywords: Petri nets, Encryption, Digital signature, Privacy, Integrity, Authentication, non-repudiatability

1. INTRODUCTION

This paper consits on the application on Petri Nets of some of the latest standard technologies used in computer security. The idea is to provide security and protection of information in data storage and sharing. In particular, we will achieve privacy, authentication, integrity and non-repudiatability data. To achieve this, we introduce some concepts such as XML, digital signature, encryption and PNML (Petri Net Marked Language).

Throughout the whole paper standard Technologies are used, but, in order to implement them, in some cases it is necessary to introduce a transformation to the data (without loss of information).

1.1. Privacy

This term is related with the prevention of unauthorized access to information. The solution is not to prevent physically access to such information, eg in a safe hiding or behind a firewall, but to encrypt the information. Nowadays it is easier to open a safe or to circumvent a firewall than to break an encryption standard algorithm (which today is impossible).

1.2. Integrity

The integrity of the data will be obtained if we can avoid or at least detect unauthorized modification of information.

1.3. Authentication

Authentication ensures that people assuring that they say or sign the data, are actually who they say they are. This avoids receiving data from a person posing as another.

1.4. Non-repudiatability

It will be obtaided if we can prevent anyone saying that has not sent or modify something done. It should be possible to assure that a preson have done something.

The solution for authentication, integrity and non repudiatability fails to deliver the information 'in hand'

to avoid disruptions and to know who gives it, as we are in the era of Internet and technology. The solution is then to digitally sign all or part of the data so that we know who has made the signing of a reliable and be able to detect unauthorized modification of any of the signed data.

1.5. XML

XML is a metalanguage for defining other languages. XML is not really a particular language, but a way of defining languages for different needs. XML is also a standard way to exchange structured information. It is based on distributed hierarchical labels containing data. The XML files are text files. The work is based on this format for implementing information security.

1.6. PNML

Marked Petri Net Language (PNML) is an XML language designed to represent Petri nets. With this language a Petri net can be stored in a text file (XML), without loss of information.

1.7. Digital certificate

A digital certificate is a digital file non-transferable and non-modifiable generated by a trusted third party called Certificate Authority (CA), that associates a public key to a person or entity. For a certificate to perform its tasks need to use a private key that only the owner possesses.

1.8. Digital signature

It is equivalent to the conventional signature. It is an addition to the document you signed and indicates that it agrees with what is said in it. The digital signature provides authentication features, integrity, and non repudiation. Computationally speaking, it is a process thst transforms the original message using the private key, and anyone with the signer's public key can verify this.

1.9. Encrypt and Decrypt

Encryption is the process to convertí in unreadable some information considered as important. Decoding is the reverse: from the encrypted content becomes legible original content. Keys are used to encrypt and decrypt. In the case that the key to encrypt and decrypt is the same, it is called symmetric encryption. If encryption is made with a key but decryption is made with a different key, it is called asymmetric encryption.

2. APPROACH

The goal of this work is to hide all or part of a Petri net that may contain sensitive information, such as a company that represents a secret production process through Petri nets (privacy). On the other hand, another goal is to ensure with standard resources that a Petri net has not been altered (integrity), and that the sender or firmer of the Petri net is who sais to be (authentication) and furdermore it may not have been another (non-repudiation).

Let us suppose we have the following Petri net.

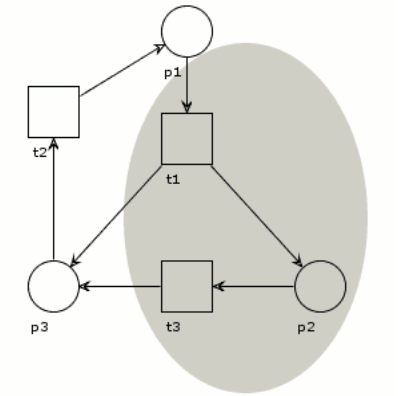

Figure 1: Petri net with a part that want to be hidden

It will be proposed how to encrypt the selected part of the graph, and to sign the Petri net, so that the obtained file meets the desired signature and encryption. So in this final file, all the shaded information (and only that) is encrypted and will not be interpretable. In particular, we will not know anything about the node p2 or transitions t1 and t3: it is a secret process. In addition, this file will contain additional data that will verify the integrity of the file to prevent that anyone modify it, as well as data about who has signed this Petri net.

3. TECHNOLOGIES

3.1. XMLEncryption

Encryption is a standard of XML files. It can be used symmetric or asymmetric encryption, but in this case, it is preferable to use symmetric encryption, because it is computationally less demanding.

The idea behind this encryption is to replace the XML elements that want to be encrypted by another piece of XML that contains encrypted information and data about the algorithms used for encryption.

When a file non-XML is encrypted, the only option is to encrypt it completely. When we apply this technology to XML, it permits to define specific fragments of the document that want to be encrypted or even to transform the document before applying encryption.

Whatever the origin of data, the result is always an XML element. Typically, this document has all the information needed to be deciphered. The information that can be found is:

• Encryption algorithm: is the name of a method for encrypting information. It may not be included, being necessary to be know by both the part that encriptes the file and the part that decryptes it.

• Encrypted information: this part must always be present.

• Name of the password used: it is optional. It is used when there is a set of keys, and have to be also

known by both the part that encriptes the file and the part that decryptes it..

• Encrypted password: it is optional. The part that encrypts the document must have a public or a private key. With this key it can encrypt the password used to encrypt the content. The part that decrypts the document must have the other key.

Below is an example of XMLEncryption. This is the original document:

```
<?xml version='1.0'?>
<PaymentInfo xmlns='http://example.org/paymentv2'>
        <Name>John Smith</Name>
        <CreditCard Limit='5,000' Currency='USD'>
                <Number>4019       2445       0277
                5567</Number>
                <Issuer>Example Bank</Issuer>
                <Expiration>04/02</Expiration>
        </CreditCard>
</PaymentInfo>
```

Figure 2: Original document with XMLEncryption

This is the document after encrypt the credit card (Figure 3). In this example we have only the encrypted information and we have no information about the key or the encryption algorithm.

```
<?xml version='1.0'?>
<PaymentInfo xmlns='http://example.org/paymentv2'>
        <Name>John Smith</Name>
        <CreditCard Limit='5,000' Currency='USD'>
                <Number>
                        <EncryptedData
                        xmlns='http://www.w3.org/2
                        001/04/xmlenc#'
                        Type='http://www.w3.org/20
                        01/04/xmlenc#Content'>
                        <CipherData>
                        <CipherValue>A23B45C56</Ci
                        pherValue>
                        </CipherData>
                        </EncryptedData>
                </Number>
        <Issuer>Example Bank</Issuer>
        <Expiration>04/02</Expiration>
        </CreditCard>
</PaymentInfo>
```

Figure 3: Encrypted document with XMLEncryption

3.2. XMLSignature

It is a standard of digital signature of files, not necessarily XML files. However, the final file is always an XML document. It requires digital certificates and public and private keys for its operation. There are three alternatives:

• Envelope: The result is the original XML file to which a signature element is added in the XML file itself.

• Enveloping: The result is an XML file with the signature, and within it, there are the original elements of the original XML file.

• Detached: The result is the original file and, separately, an XML file with the signature of that file.

It really does not matter which one to use. They are simply different ways of organizing the generated signature.

```
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
  <SignedInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
    <CanonicalizationMethod
      Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"
      xmlns="http://www.w3.org/2000/09/xmldsig#" />
    <SignatureMethod
      Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1"
      xmlns="http://www.w3.org/2000/09/xmldsig#" />
    <Reference URI=""
      xmlns="http://www.w3.org/2000/09/xmldsig#">
      <Transforms
        xmlns="http://www.w3.org/2000/09/xmldsig#">
        <Transform
          Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"
          xmlns="http://www.w3.org/2000/09/xmldsig#" />
      </Transforms>
      <DigestMethod
        Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"
        xmlns="http://www.w3.org/2000/09/xmldsig#" />
      <DigestValue
        xmlns="http://www.w3.org/2000/09/xmldsig#">
        Oyyx+K28+cp7kuUgcnANtTBdUwg=
      </DigestValue>
    </Reference>
  </SignedInfo>
  <SignatureValue xmlns="http://www.w3.org/2000/09/xmldsig#">
    ZVzRud7G4mEZsDnBavbnZoFUmm5J2OBDkQ+looDLn95ndGYdrq6uPQ==
  </SignatureValue>
  <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
    <X509Data xmlns="http://www.w3.org/2000/09/xmldsig#">
      <X509Certificate
        xmlns="http://www.w3.org/2000/09/xmldsig#">
        IICmDCCAlYCBEfrim8wCwYHKoZIzjgEAwUAMDIxCzAJBgNVBAYTAkVTMREwDwYDVQQKEwhBdXRI
        pYTEQMA4GA1UEAxMHVXN1YXJpbzAeFw0wODAzMjcxMTUyMTVaFw0wOTAzMjcxMTUyMTVaMDIx
        BgNVBAYTAkVTMREwDwYDVQQKEwhBdXRIbnRpYTEQMA4GA1UEAxMHVXN1YXJpbzCCAbcwggEs
        BgcqhkjOOAQBMIIBHwKBgQD9f1OBHXUSKVLfSpwu7OTn9hG3UjzvRADDHj+AtlEmaUVdQCJR+1k9
        jVj6v8X1ujD2y5tVbNeBO4AdNG/yZmC3a5lQpaSfn+gEexAiwk+7qdf+t8Yb+DtX58aophUPBPuD
        9tPFHsMCNVQTWhaRMvZ1864rYdcq7/IiAxmd0UlgBxwlVAJdgUI8VIwvMspK5gqLrhAvwWBz1AoGB
        APfhoIXWmz3ey7yrXDa4V7l5lK+7+jrqgvlXTAs9B4JnUVIXjrrUWU/mcQcQgYC0SRZxl+hMKBYT
        t88JMozIpuE8FnqLVHyNkOCjrh4rs6Z1kW6jfwv6ITVi8ftiegEkO8yk8b6oUZCJqlPf4VrInwaS
        i2ZegHtVJWQBTDv+z0kqA4GEAAKBgDUPDwxDZFXMrZha74VNmgyFsILM01wKw17nbt9UFTJALk71
        iPpozeZMP2u0SoYst2nbxkCs1hziiuaNJnykzcjVf3+PmL3sQES8SxwJBRUME2UTA2006WD3xNy9
        iZ9yibcWQimB8eKIJyBBxSk5TueAzvTA8HN2+Rvgh8RMaOzhMAsGByqGSM44BAMFAAMvADAsAhQH
        4+nQZdFvlvsfyOfq1t02h9MJEgIUEvYDfxeygKCmrIlA0sQLtaCs0Qo=
      </X509Certificate>
    </X509Data>
    <KeyValue xmlns="http://www.w3.org/2000/09/xmldsig#">
      <DSAKeyValue
        xmlns="http://www.w3.org/2000/09/xmldsig#">
        <P xmlns="http://www.w3.org/2000/09/xmldsig#">
          /X9TgR11EilS30qcLuzk5/YRt1I870QAwx4/gLZRJmlFXUAiUftZPY1Y+r/F9bow9subVWzXgTuA
          HTRv8mZgt2uZUKWkn5/oBHsQIsJPu6nX/rfGG/g7V+fGqKYVDwT7g/bTxR7DAjVUE1oWkTL2dfOu
          K2HXKu/ylgMZndFIAcc=
        </P>
        <Q xmlns="http://www.w3.org/2000/09/xmldsig#">
          l2BQjxUjC8yykrmCouuEC/BYHPU=
        </Q>
        <G xmlns="http://www.w3.org/2000/09/xmldsig#">
          9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdRWVeOutRZT+ZxBxCBgLRJFnEj6EwoFhO3
          zwKyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx+2J6ASQ7zKTxvqhRkImog9/hVuWfBpKL
          Zl6Ae1UlZAFMO/7PSSo=
        </G>
        <Y xmlns="http://www.w3.org/2000/09/xmldsig#">
          NQ8PDENkVcytmFrvhU2aDlWyUszTXArDXudu31QVMkAuTvWI+mjN5kw/a7RKhiy3advGQKzWHOKK
          5o0mfKTNyNV/f4+YvexARLxLHAkFFQwTZRMDbTTpYPfE3L2Jn3KJtxZCKYHx4ognlEHFKTIO54DO
          9MDwc3b5G+CHxExo7OE=
        </Y>
      </DSAKeyValue>
    </KeyValue>
  </KeyInfo>
</Signature>
```

Figure 4: XML obtained after applying XMLSignature

A signature as must contain, accordingly with XML Signature:

• Canonicalization method: Two XML documents are equivalent if they represent the same information. A method of canonicalization transforms an XML document into another equivalent one. All XML documents equivalent, since they are canonicalized using the same method, result in the same XML. It is applyed before signing. If a method is not specified, one of them is assigned by default.

• Reference: There may be several references within a single firm. In each reference the part of the document that is signed and the hash algorithm used are indicated. A summary algorithm generates a sequence of bytes of fixed length from contents of arbitrary length. This sequence of bytes is different for each content.

• Information on the key signature can optionally include the data necessary for validation. This part can indicate the public key directly, through a sequence of characters that identifies it or through a URL. Additionally it can also have more information about who has signed it: name, organization, country...

• Transformations: It is possible that what want to be signed is not the complete document, but some information of it. With the changes you can do almost anything, from selecting only certain parts, to change the structure of XML, or to include other XML fragments. If it is not necessary to apply any transformation before signing you can skip this part.

The end result of applying XML Signature is an XML element of the form shown in Figure 4.

4. PROPOSAL

4.1. Encription

Here are presented 4 equivalent representations of a PN: as graph, code, matrix, and PNML.

Graph:
(see figure 1)

PNML:

```
<?xml version="1.0" encoding="UTF-8"?>
<pnml>
 <net type="http://www.informatik.hu-
berlin.de/top/pntd/ptNetb" id="noID">
  <place id="p3">
   <name>
    <text>p3</text>
   </name>
  </place>
  <place id="p2">
   <name>
    <text>p2</text>
   </name>
  </place>
  <place id="p1">
   <name>
    <text>p1</text>
   </name>
  </place>
  <transition id="t3">
   <name>
    <text>t3</text>
   </name>
  </transition>
  <transition id="t2">
   <name>
    <text>t2</text>
   </name>
  </transition>
  <transition id="t1">
   <name>
    <text>t1</text>
   </name>
  </transition>
  <arc id="a9" source="t3" target="p3">
   <inscription>
    <text>1</text>
   </inscription>
  </arc>
  <arc id="a11" source="p2" target="t3">
   <inscription>
    <text>1</text>
   </inscription>
  </arc>
  <arc id="a1" source="p1" target="t1">
   <inscription>
    <text>1</text>
   </inscription>
  </arc>
  <arc id="a2" source="t1" target="p2">
   <inscription>
    <text>1</text>
   </inscription>
  </arc>
  <arc id="a3" source="t1" target="p3">
   <inscription>
    <text>1</text>
   </inscription>
  </arc>
  <arc id="a4" source="p3" target="t2">
   <inscription>
    <text>1</text>
   </inscription>
  </arc>
  <arc id="a7" source="t2" target="p1">
   <inscription>
    <text>1</text>
   </inscription>
  </arc>
 </net>
</pnml>
```

Figure 7: Example of a Petri net defined by PNML.

Matrix:

```
      p1  p2  p3
t1  - [ 1   1  1 ]
t2    [ 1   0 -1 ]
t3    [ 0  -1  1 ]
```

Figure 6: Example of a Petri net defined by its incidence matrix.

Code:

```
if (p1>0) then
    p1 <- p1 – 1
    p2 <- p2 + 1
    p3 <- p3 + 1
if (p2>0) then
    p2 <- p2 - 1
    p3 <- p3 + 1
if (p3>0) then
    p3 <- p3 – 1
    p1 <- p1 + 1
```

Figure 5: Example of a Petri net defined by code.

The proposed solution is to use the PNML representation of Petri nets, and from it to use XMLEncryption standards for encryption and XMLSignature for the signature.

A little example will be developed to show that it reduces to perform the signature and / or encrypted operations on a Petri subnet of the original network, and that also the matrix associated to the network more appropriated can be selected.

A Petri subnet is a submatrix of the matrix associated to the network. In the matrix, the rows are associated to transitions, and the columns to nodes. This way we can easily show that there exists a single matrix associated to a PN, but if we exchange two rows or two columns of the matrix, the result also defines the same Petri net (or more precisely, it defines an equivalent Petri net).

```
      p1  p2  p3          p1  p2  p3          p2  p1  p3
t1  [ -1   1   1 ]   t1  [ -1   1   1 ]   t1  [  1  -1   1 ]
t2  [  1   0  -1 ]   t3  [  0  -1   1 ]   t3  [ -1   0   1 ]
t3  [  0  -1   1 ]   t2  [  1   0  -1 ]   t2  [  0   1  -1 ]
```

Figure 8: Example of equivalent Petri nets

All these would be equivalent representations. It can be shown that indeed an equivalence relation is between matrices M and M'. The equivalence relation is M r M 'if we can get from M to M' by swaping rows and/or columns. To test compliance reflexive relations (reflexive), symmetric (symmetric) and transitive (transitive).

Let T be the set of transformations of a matrix of order nxm (n rows and m columns). Let be Tfij, with i <= n, and j <= n, the transformation that exchanges the row i with row j. Obviously Tfij = Tfji. Similarly, we define Tckl with k<=m and l<=m, as the transformation that exchanges the column k to column l. Similarly, Tckl = Tclk. Let T = {Tabs, s> = 1 | = Tfij Tabs, a = i, b = j, i <= n, j <= n or = Tckl Tabs, a = k, b = l, k <= m, l <= m} where s is a sequence of consecutive natural numbers beginning with 1. Thus Tab1 is the first transformation, Tab2 the second, ..., and Tabn is the nth. Thus we have an ordered set of transformations applied to one mxn matrix in a particular order. It can be shown that the order in which transformations are

applied does not influence the final result, but it is not necessary for our purpose.

Let r be the following relationship we want to study: A matrix M is related to an N, M r N, if you can get from M to N with a finite number of transformations. A transformation will be an exchange of two rows or two columns. Let show that this relation is of equivalence:

• Reflexive: M r M. Obviously, since you do not need any exchange of rows or columns. In this case T = Ø.

• Symmetric: if M r M 'then M' r M. If from M p exchange operations are made from in rows and / or columns to arrive at M', from M' the same operations are performed in reverse order in order to arrive at M. Thus if T = {Tabs, s = 1 .. p} and the number of transformations is p> 0, then T = {Tab (n-s +1), s = 1 .. p} is the set of transformations leading from M 'to M.

• Transitive: if M r M' and M' r M r M * then M r M *. Let T1 = { Tabs, s = 1 .. p} of size p the set of transformations that lead from M to M' and let T2 = {Tabr, s = 1 .. q} q-sized, the set of transformations that lead from M' to M*. Then T = { Tabt with t = s if t <= p and t = p + r if t> p} is a sequence of transformations that lead from M to M*.

Therefore it is shown that r is an equivalence relation. Thus, we can say that a Petri net corresponds to an equivalence class of the relation r. Thus, we can choose on what representative of the class to make the transformations.

With this in mind, given a matrix representing a Petri net, if we eliminate some row and / or a column, a valid subset results. This subnet is what we want to process. As a row is associated to a place and a transition to a column, any subset of places and transitions can be selected as a valid subnet.

Following the selection of places and transitions that are to be processed, a matrix having first nodes and transitions th eones that we want to encrypt can be chosen as representative of the Petri net. In our case:

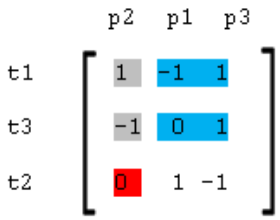

Figure 9: Petri net representing the equivalence class

Interpreting the Figure 9, differnt parts can be seen:

• The gray part corresponds to the parts that are completely into the process. In this case p2, t1, t3, the arc from t1 to p2, and the arc from p2 to t3. It is denoted as hidden subnet.

• The blue arcs indicate arcs (> 0) that part from a hidden transition but become to a visible place or from a visible place to a hidden transition (<0). It is denoted as hidden transitions subnet.

• The red part corresponds to the arcs starting from a hidden node to a visible transition (<0) or from a visible transition to a hidden node (> 0). It is denoted as hidden nodes subnet.

• The uncoloured part are nodes, transitions, and arcs that are not hidden. It is denoted as visible subnet.

Thus, a Petri net that wants to be encrypted can be represented by a matrix as follows:

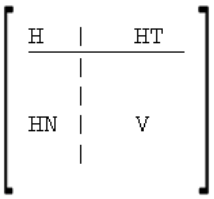

Figure 10: Parts in an ordered encrypted Petri net

Being H the hidden subnet, HT the hidden transitions subnet, HN the hidden nodes subnet, and V the visible subnet.

What will be encrypted, in order to not to give information about the structure, corresponds to the matrices H, HT and HN, which are those affected by any hidden element.

Let us now see how would be the PNML representation associated with this subnet. Within the document PNML three main elements appears:

• place: defines a place with an id and a name (a column of the matrix).

```
<place id="p1">
  <name>
    <text>nodo 1</text>
  </name>
</place>
```

• transition: defines a transition, also with an id and a name (a row of the matrix).

```
<transition id="t2">
  <name>
    <text>transicion 2</text>
  </name>
</transition>
```

• arc: defines an arrow with an ID, from one node to a transition or from a transition to a node, defined by its id (one matrix element different from 0).

```
<arc id="a2" source="t1" target="p2">
  <inscription>
    <text>1</text>
  </inscription>
</arc>
```

Note that no matter the order in which these elements are in the PNLM file. Just as there are several matrices that represent the same net, the same goes for files PNML. The order in the file provides no information. It is similar that appear first all places, then all the transitions and finally all the arcs, that all of them

interspersed with each other. Thus, once we have defined what is the subset of nodes and transitions that we want to encrypt, what will be done in the PNML file is to join all these nodes, transitions and arcs that contain as the origin or the end any of these nodes and transitions, and include a new XML element called 'subnet' within the PNLM document, with a concrete and unique id. Later it would be indicated what does this id is used for.

Therefore, in the example, the subset would be {p2, t1 and t3}; the new file applying these changes to the original PNML would result this way, grouping these three elements together with the arcs that have one of them as a source or destination; that is, everything associated with the matrices H, HT and HN. The visible subnet, V, remains out of this item.

```
<?xml version="1.0" encoding="UTF-8"?>
<pnml>
  <net type="http://www.informatik.hu-
berlin.de/top/pntd/ptNetb" id="noID">
   <subnet id="subnet1">
    <place id="p2">
     <name>
      <text>p2</text>
     </name>
    </place>
    <transition id="t1">
     <name>
      <text>t1</text>
     </name>
    </transition>
    <transition id="t3">
     <name>
      <text>t3</text>
     </name>
    </transition>
    <arc id="a9" source="t3" target="p3">
     <inscription>
      <text>1</text>
     </inscription>
    </arc>
    <arc id="a11" source="p2" target="t3">
     <inscription>
      <text>1</text>
     </inscription>
    </arc>
    <arc id="a1" source="p1" target="t1">
     <inscription>
      <text>1</text>
     </inscription>
    </arc>
    <arc id="a2" source="t1" target="p2">
     <inscription>
      <text>1</text>
     </inscription>
    </arc>
    <arc id="a3" source="t1" target="p3">
     <inscription>
      <text>1</text>
     </inscription>
    </arc>
   </subnet>    <place id="p1">
     <name>
      <text>p1</text>
     </name>
    </place>
    <place id="p3">
     <name>
      <text>p3</text>
     </name>
    </place>
    <transition id="t2">
     <name>
      <text>t2</text>
     </name>
    </transition>
    <arc id="a4" source="p3" target="t2">
     <inscription>
      <text>1</text>
     </inscription>
    </arc>
    <arc id="a7" source="t2" target="p1">
     <inscription>
      <text>1</text>
     </inscription>
    </arc>
   </net>
  </pnml>
```

Figure 11: PNML of the example, grouping the elements of H, HT y HN.

Notice that this PNML file does not meet the official PNML grammar, but these changes can always be undone to leave the original PNML file (that is, the transformation is reversible). The goal is not thatthe encrypted and/or signed document meets the grammar, but that, alter decrypt and/or check the signature, the original file can be obtained. To obtain the original file it is Orly neccssary to take the elements from the 'subnet' labels.

Note that multiple subnets can be encrypted just considering all of them as a single subnet with the union of the nodes and transitions of them.

Once we got the PNML file in this format, we can apply the encryption via XMLEncryption. The final file is shown in Figure 12.

Notice that the content of element 'subnet' no longer exists and has been replaced by an element 'EncryptedData'. The subnet has already been encrypted. A possible graph representation would be Figure 13, where the subnet is the visible subnet V and the subnets H, HT, and HN are hidden in the black box.

```
<?xml version="1.0" encoding="UTF-8"?>
<pnml>
 <subnet id="subnet1">
  <net type="http://www.informatik.hu-berlin.de/top/pntd/ptNetb" id="noID">
   <xenc:EncryptedData xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
     Type="http://www.w3.org/2001/04/xmlenc#Element">
     <xenc:EncryptionMethod
       Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc"
       xmlns:xenc="http://www.w3.org/2001/04/xmlenc#" />
     <xenc:CipherData
       xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
       <xenc:CipherValue
         xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
          Wr1njyJlYYOM9IAYqcwGCWkw2L4pUjQD2GGVoU
          9lVZ0wKqHY8y3l3GY8FY4i5K3G8grle1HRFqe
          7RtkFiXZgGMeYnQp6oB6ckKp3KFkHVqtucc9
          rAVzOgC7XAw/oe61HRFqe6RRVzXjNM GY8FY4i5K
          dId8NVPQmUSDX7NRtnRXZgGMeYnQp6oB6ckKp3
          SWr1njyJlYYOM9IAYqcwGCWkw2L4pU6ckKp3KFK
          9hlVZ0wKqHY8y3l3GY8FY4i5K3G8grle2xN4u7x
          7fRtkFiXZgGMeYnQp6oB6ckKp3KFRRVzXjN
          AtVzOgC7XAw/oe61HRFqe6RRVzXjNMLU5ZgGMeYn
          Iy8NVPQmUSDX7NRtnR6YnQp6oB6GY8F
          dWr1njyJlYYOM9IAYqcwGCWkw2L4pUjQ3l3GY8FY
          v9lVZ0wKqHY8y3l3GY8FY4i5K3G8grle2xN4u7x
          b7RtkFiXZgGMeYnQp6oB6ckKp3KFKG8grle2
          nAVzOgC7XAw/oe61HRFqe6RRVzXjNMLU5T1HRFqe
          Ll8NVPQmUSDX7NRtnR68=
       </xenc:CipherValue>
     </xenc:CipherData>
   </xenc:EncryptedData>
  </subnet>
    <place id="p1">
     <name>
      <text>p1</text>
     </name>
    </place>
    <place id="p3">
     <name>
      <text>p3</text>
     </name>
    </place>
    <transition id="t2">
     <name>
      <text>t2</text>
     </name>
    </transition>
    <arc id="a4" source="p3" target="t2">
     <inscription>
      <text>1</text>
     </inscription>
    </arc>
    <arc id="a7" source="t2" target="p1">
     <inscription>
      <text>1</text>
     </inscription>
    </arc>
   </net>
  </pnml>
```

Figure 12: PNML after applying XMLEncription.

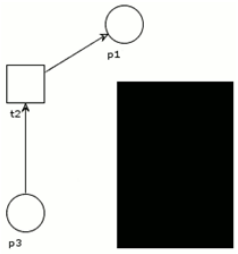

Figure 13: Grafic representation of the Petri net after applying XMLEncription.

Once encrypted information, even the number of nodes, transitions, and arcs that are contained in that black box is unknown. The final matrix associated is represented in figure 14. Black areas are those for which we have no information, and even the size of the matrix is unknown.

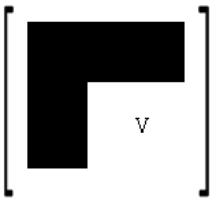

Figure 14: Matrix representation of the Petri net after applying XMLEncription.

Note that in the hidden network no arc comes in or comes out. This is a security decision. However, we could define the arcs that go from inside out or from outsider in, hiding the place or the transition of destination in the hidden subnet, replacing the node / transition id of origin or destination inside the hidden net by the own id of the net. Thus, the final file would be as follows:

Figure 15: Alternative encryption allowing arc knowledge in the hidden Petri net.

Thus, the final graph representation would be similar to Figure 16. Note that although arcs to / from the hidden subnet exist, they do not indicate which is the transition or node of origin or destination within the hidden subnet.

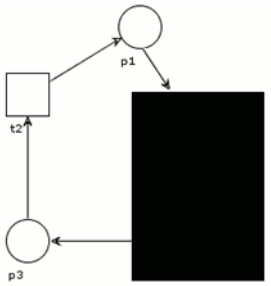

Figure 16: Graphic representation of the alternative encryption of Figure 15..

4.2. Signature

In this case what is followed is that the Petri net, once completed, can not be modified without being detected, and that is is possible to know exactly who firms it.

I could do A development similar to the one carried out with encryption could be made, by signing only part of the network, but in this case it will be simplifyed and the entire network will be signed. For this, XMLSignature is used, and together with a certificate and a public key, the file generated above is signed. Thus, the final file is as shown in Figure 17.

Figure 17: Example of the Petri net after applying XMLSignature.

REFERENCES

Fan YS, Zhou M, et al., 2010, Data-Driven Service Composition in Enterprise SOA Solutions: A Petri Net Approach. Tan W, IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING Volume: 7 Issue: 3 Pages: 686-694

Fahmy Hma, ANALYSIS OF PETRI NETS BY PARTITIONING - SPLITTING PLACES OR TRANSITIONS. 1993. INTERNATIONAL JOURNAL OF COMPUTER MATHEMATICS Volume: 48 Issue: 3-4 Pages: 127-148

Nordbotten NA. XML and Web Services Security Standards. 2009, IEEE COMMUNICATIONS SURVEYS AND TUTORIALS Volume: 11 Issue: 3 Pages: 4-21 Published: 2009

Ekelhart A, Fenz S, Goluch G, et al. XML security - A comparative literature review. 2008. JOURNAL OF SYSTEMS AND SOFTWARE Volume: 81 Issue: 10 Pages: 1715-1724

Meadors K. Secure electronic data interchange over the Internet. 2005 IEEE INTERNET COMPUTING Volume: 9 Issue: 3 Pages: 82-89

Selkirk A. XML and security. 2001 BT TECHNOLOGY JOURNAL Volume: 19 Issue: 3 Pages: 23-34

Selkirk A. Using XML security mechanisms. 2001 BT TECHNOLOGY JOURNAL Volume: 19 Issue: 3 Pages: 35-43

Nordbotten NA. XML and Web Services Security Standards. 2009 IEEE COMMUNICATIONS SURVEYS AND TUTORIALS Volume: 11 Issue: 3 Pages: 4-21 Published: 2009

Brooke PJ, Paige RF, Power C. Document-centric XML workflows with fragment digital signatures. 2010 SOFTWARE-PRACTICE & EXPERIENCE Volume: 40 Issue: 8 Pages: 655-672