Alternate Representations of the Public Key Cryptography Standards (PKCS) Using S-Expressions, S-PKCS

Matthew D. Wood
Carl M. Ellison
Intel Security Technology Lab

Abstract

Most of the existing RSA Data Security* Inc. Public Key Cryptography Standards (PKCS) documents describe structured data that must be represented inside encrypted blobs or transferred to a second party. In many cases the two parties are processes running on machines of different platform architectures. To accommodate this, ASN.1 syntax and BER/DER encoding is used. While BER/DER is acceptable for platform-independent data transfer, the burden imposed on the application writer is large, either in the form of hand-written code or obtaining an ASN.1 compiler.

This paper proposes an alternate representation of the PKCS data structures that possesses the same cross-platform properties. The data structures are represented as S-expressions. Using S-expressions allows any application developer with a basic knowledge of a programming language and data structures to efficiently interpret and manipulate the data, and provides the application with a more explicit way of discovering the identity of a data structure.

We do not propose replacing existing uses of PKCS with S-PKCS, but propose this version as an alternative for applications that are not using ASN.1. This version was inspired by SPKI/SDSI, which had already adopted PKCS formats in S-expression form. There will undoubtedly be other applications that adopt these formats in the future.

Contents

Introduction

Most of the existing RSA Data Security* Inc. Public Key Cryptography Standards (PKCS) documents describe structured data that must be represented inside encrypted blobs or transferred to a second party. In many cases the two parties are processes running on machines of different platform architectures. To accommodate this, ASN.1 syntax and BER/DER encoding is used. While BER/DER is acceptable for platform-independent data transfer, the burden imposed on the application writer is large, either in the form of hand-written code or obtaining an ASN.1 compiler.

This paper proposes an alternate representation of the PKCS data structures using S-expressions [SEXP] that possess the same cross-platform properties as BER/DER encoding. Since S-expressions are simple structures, they lend themselves to efficient manipulation and interpretation by any application developer with basic knowledge of a programming language and data structures. The code to manipulate S-expression structures is often smaller than the corresponding code to manipulate ASN.1 structures.

A key benefit of S-expressions over ASN.1 structures is that S-expressions provide the applications with an explicit method of discovering the identity of a data structure. This is in great contrast to ASN.1 structures where the identity of some structures is assumed based on the context within the data structure. S-expressions provide a way for the application to quickly and accurately verify the identity of an S-expression and its expected structure.

S-Expressions

We have chosen a simplified form of S-expression (the canonical form) as the format for S-PKCS objects. An S-expression is a list enclosed in matching "(" and ")". We assume the S-expression technology of [SEXP] with the restrictions that no empty lists are allowed and that each list must have a byte string as its first element. That first element is the "type" or "name" of the object represented by the list. We also assume that the values in each S-expression are listed in a fixed order specified by the syntax of the expression.

S-PKCS objects are defined below in a familiar extension of BNF -- with "|" meaning logical OR, "*" meaning closure (0 or more occurrences), "?" meaning optional (0 or 1 occurrence) and "+" meaning non-empty closure (1 or more occurrences). A quoted string represents those characters. First we define the canonical S-expression form in that BNF.

For the sake of readability, all examples in this document specify advanced rather than canonical S-expressions. That is, single word strings that start with alphabetic characters are used without quotes and strings can be in hex, base64 or double-quoted ASCII. See [SEXP] for a complete description of S-expression formats.

Canonical S-expression Definition

All S-expressions are to be communicated in their canonical form. The following BNF defines the grammar for all S-expressions used in this document. The same definitions as [SPKI] are used whenever appropriate (display hints are not used).

```
<s-expr>:: "(" <byte-string> <s-part>* ")" ;

<s-part>:: <byte-string> | <s-expr> ;

<byte-string>:: <decimal> ":" {binary byte string of that length} ;
```

```
<decimal>:: <nzddigit> <ddigit>* | "0" ;
<nzddigit>:: "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" ;
<ddigit>:: "0" | <nzddigit> ;
<integer> :: <byte-string> ;
```

> <integer> is a representation of a signed integer value using the minimum number of bytes, with the most significant byte first. The integers are twos-compliment signed values, which requires unsigned integers with the most significant bit set to have a zero byte prepended.

Useful S-Expression Definitions

The following definitions are used throughout this document.

```
<algorithms> :: <digest-alg> |
                <signature-alg> |
                <encrypt-alg> |
                <derive-alg> |
                <mac-alg> |
                <asymmetric-alg> ;
<digest-alg> :: "md5" | "sha1" | …;
<signature-alg> :: "rsa-pkcs1-sha1" | "dsa-sha1" | … ;
<encrypt-alg> :: "rsa-pkcs1-oaep" | "rsa-pkcs1-v15" |
                 "pkcs5-pbes1-des-cbc" | "pkcs5-pbes1-rc2-cbc" |
                 "pkcs5-pbes2" | … ;
<derive-alg> :: "pkcs5-pbkdf1" | "pkcs5-pbkdf2" ;
<mac-alg> :: "hmac-sha1" | "hmac-md5" | … ;
<asymmetric-alg> :: "rsa" | "dsa" | "dh-pkcs3" | "ecdsa" | … ;
<version> :: "(" "version" <integer> ")" ;
```

Representations of Non-PKCS Structures

<algorithm-identifier> is the S-expression representation of the ASN.1 structure `AlgorithmIdentifier` defined in [X509]. The <algorithm-params> field is optional, and its value is dictated by the value of the <algorithms> field.

```
<algorithm-identifier> :: "(" <algorithms> <algorithm-params>? ")" ;
```

PKCS #1: RSA Encryption Standard

PKCS #1 [PKCS1] defines the mathematical operations to perform the RSA algorithm, as well as multiple encryption and signature with appendix schemes, encryption scheme parameters, and key blob representations. This section presents the S-expression representation of the ASN.1 data structures found in [PKCS1].

<digest-info>

The <digest-info> structures are embedded in digital signatures using the RSASSA-PKCS1-v1_5 signature scheme.

```
<digest-info> :: "(" <digest-alg> <digest-value> ")" ;
<digest-value> :: <byte-string> ;
```

<rsa-public-key>

<rsa-public-key> represents a public key generated according to [PKCS1]. The format is compatible with the RSA public key representation in [SPKI], except that the algorithm name does not include a hash name.

```
<rsa-public-key> :: "(" "public-key"
                        "(" "rsa"
                            "(" "n" <integer> ")"
                            "(" "e" <integer> ")"
                        ")"
                    ")" ;
```

<rsa-private-key>

<rsa-private-key> represents a private key generated according to [PKCS1]. The values "a" and "b" correspond to the "exponent1" and "exponent2" fields in `RSAPrivateKey` respectively.

```
<rsa-private-key> :: "(" "private-key"
                         "(" "rsa"
                             <version>?
                             "(" "n" <integer> ")"
                             "(" "e" <integer> ")"
                             "(" "d" <integer> ")"
                             "(" "p" <integer> ")"
                             "(" "q" <integer> ")"
                             "(" "a" <integer> ")"
                             "(" "b" <integer> ")"
                             "(" "c" <integer> ")"
                         ")"
                     ")" ;
```

The value of <version> for this specification is: (version 1)

<rsaes-oaep-params>

<rsaes-oaep-params> represents the configuration used with the RSAES-OAEP encryption scheme. All fields are optional. If not present, a default value is assumed by the application. Table 1 lists the default values for each field. If all fields in the parameter structure are omitted, the entire S-expression may be omitted and the defaults are assumed by the application.

```
<rsaes-oaep-params> :: "(" "rsaes-oaep-params"
                           <digest-alg>?
                           <rsaes-oaep-mgf>?
                           <rsaes-oaep-data-source>?
                       ")" ;
<rsaes-oaep-mgf> :: <rsaes-oaep-mgf1> ;
<rsaes-oaep-mgf1> :: "(" "mgf1" <digest-alg> ")" ;
<rsaes-oaep-data-source> :: <rsaes-oaep-data-specified> ;
<rsaes-oaep-data-specified> :: "(" "specified" <byte-string> ")" ;
```

Table 1, Default values for omitted <rsaes-oaep-params> fields

Field	Default Value
<digest-alg>	sha1
<rsea-oaep-mgf>	<rsaes-oaep-mgf1> with its <digest-alg> value set to "sha1"
<rsaes-oaep-data-source>	empty

The following are all valid <rsaes-oaep-params> values.

```
(rsaes-oaep-params sha1 (mgf1 sha1) (specified #0123456789ABCDEF#))
(rsaes-oaep-params (mgf1 md5))
(rsaes-oaep-params sha1 (specified #0123456789ABCDEF#))
(rsaes-oaep-params)
```

PKCS #3: Diffie-Hellman Key-Agreement Standard

PKCS #3 [PKCS3] defines the mathematical operations to perform conventional Diffie-Hellman key agreement, and algorithm parameters. This section presents the S-expression representation of the ASN.1 data structures found in [PKCS3].

<dh-params>

```
<dh-params> :: "(" "dh-params"
                   "(" "p" <integer> ")"
                   "(" "g" <integer> ")"
               ")" ;
```

PKCS #5: Password-Based Cryptography Standard

PKCS #5 [PKCS5] defines the multiple methods for deriving an encryption key from a password and salt value, encryption and MAC methods, and key derivation parameters. This section presents the S-expression representation of the ASN.1 data structures found in [PKCS5].

<pbes1-params>

<pbes1-params> is used to represent the key derivation parameters when using the PBES1 encryption scheme. The <pbes1-salt-source> field must be 8 bytes long, and <iteration-count> must be at least 1000.

```
<pbes1-params> :: "(" "pbes1-params"
                      <pbes1-salt-source>
                      <iteration-count>
                  ")" ;
<pbes1-salt-source> :: <byte-string> ;
<iteration-count> :: <integer> ;
```

<pbkdf2-params>

<pbkdf2-params> is used to represent the key derivation parameters when using the PBKDF2 method. PBES2 and PBMAC1 use this method.

```
<pbkdf2-params> :: "(" "pbkdf2-params"
                       <pbkdf2-salt-source>
                       <iteration-count>
                       <derived-key-length>?
                       <pbkdf2-prng>?
                    ")" ;
<pbkdf2-salt-source> :: "(" "specified" <byte-string> ")" ;
<derived-key-length> :: <integer> ;
<pbkdf2-prng> :: <algorithm-identifier> ;
```

<pbes2-params>

<pbes2-params> is used to represent all operations required to decrypt a message encrypted with PBES2, including key derivation and encryption.

```
<pbes2-params> :: "(" "pbes2-params"
                      <pbes2-key-derive-alg>
                      <pbes2-encrypt-alg>
                   ")" ;
<pbes2-key-derive-alg> :: <algorithm-identifier> ;
<pbes2-encrypt-alg> :: <algorithm-identifier> ;
```

For PKCS #5 v2.0, the <pbes2-key-derive-alg> is restricted to the following expression. Terms enclosed in ## symbols indicate values chosen by the application.

```
(pkcs5-pbkdf2
    (pbkdf2-params
        (specified #XX#)
        #Iterations#
        #KeyLength#?
        (hmac-sha1)
   )
)
```

<pbmac1-params>

<pbmac1-params> is used to represent all operations required to verify a MAC generated using PBMAC1, including key derivation and underlying MAC algorithm.

```
<pbmac1-params> :: "(" "pbmac1-params"
                       <pbmac1-key-derive-alg>
                       <pbmac1-mac-alg>
                    ")" ;
<pbmac1-key-derive-alg> :: <algorithm-identifier> ;
<pbmac1-mac-alg> :: <algorithm-identifier> ;
```

For PKCS #5 v2.0, the <pbmac1-key-derive-alg> is restricted to the following expression. Terms enclosed in ## symbols indicate values chosen by the application.

```
(pkcs5-pbkdf2
    (pbkdf2-params
        (specified #XX#)
        #Iterations#
        #KeyLength#?
        (hmac-sha1)
   )
)
```

PKCS #8: Private-Key Information Syntax Standard

PKCS #8 [PKCS8] defines a standardized encoding format for plain text and encrypted private keys. This section presents the S-expression representation of the ASN.1 data structures found in [PKCS8].

<private-key-info>

The <private-key-info> structure is used to represent any arbitrary private key. This structure is encrypted with the application's choice of algorithm and placed into the <encrypted-private-key> field of an <encrypted-private-key-info> structure defined below.

```
<private-key-info> :: "(" "private-key-info"
                          <version>?
                          <algorithm-identifier>
                          <private-key-value>
                      ")" ;
```

The value of <version> for this specification is: (version 1)

When encoding a DSA private key, the *p*, *q*, and *g* values are encoded into the <algorithm-identifier> field of <private-key-info> using the <dsa-param> structure. When encoding PKCS #3 [PKCS3] private keys, the *p* and *g* values are encoded using the <dh-params> structure defined above. In both cases, the <private-key-value> field consists of the private *x* value.

```
<dsa-param> :: "(" "dsa-param"
                   "(" "p" <integer> ")"
                   "(" "q" <integer> ")"
                   "(" "g" <integer> ")"
               ")" ;

<private-key-value> :: <rsa-private-key> |
                       <dsa-private-key-value> |
                       <dh-private-key-value> |
                       … ;

<dsa-private-key-value> :: "(" "x" <integer> ")" ;

<dh-private-key-value> :: "(" "x" <integer> ")" ;
```

The following expression is an example of a <private-key-info> structure for an RSA key, followed by an example DSA key.

```
(private-key-info
    (rsa)
    (private-key
        (rsa
            (n …)(e …)(d …)(p …)(q …)(a …)(b …)(c …)
        )
```

```
    )
)
```

<version> omitted; application assumes default: (version 1)

```
(private-key-info
    (version 1)
    (dsa
        (dsa-param
            (p …)(q …)(g …)
        )
    )
    (x …)
)
```

The <encrypted-private-key-info> structure contains the parameters required to decrypt an encrypted <private-key-info> structure located in the <encrypted-private-key> field.

```
<encrypted-private-key-info> :: "(" "encrypted-private-key-info"
                                    <encrypt-alg>
                                    <encrypted-private-key>
                                ")" ;

<encrypt-alg> :: <algorithm-identifier> ;

<encrypted-private-key> :: <byte-string> ;
```

The following expression is an example <encrypted-public-key-info> structure. The <private-key-info> structure was encrypted using PKCS #5 scheme PBES2 with DES in CBC mode.

```
(encrypted-private-key-info
    (pkcs5-pbes2
        (pbes2-params
            (pkcs5-pbkdf2
                (pbkdf2-params
                    (specified #0123456789ABCDEF#)
                    #03E8#
                    (hmac-sha1)
                )
            )
            (des-cbc
                (iv #0123456789ABCDEF#)
            )
        )
    )
    #A236D974B … #
)
```

References

[PKCS1] PKCS #1: RSA Encryption Standard, RSA Data Security, Inc., October 1, 1998, Version 2.0.

[PKCS3] PKCS #3: Diffie-Hellman Key-Agreement Standard, RSA Data Security, Inc., November 1, 1993, Version 1.4.

[PKCS5] PKCS #5: Password-Based Cryptography Standard, RSA Data Security, Inc., March 25, 1999, Version 2.0.

[PKCS8] PKCS #8: Private-Key Information Syntax Standard, RSA Data Security, Inc., November 1, 1993, Version 1.2.

[SEXP] Ron Rivest, code and description of S-expressions, http://theory.lcs.mit.edu/~rivest/sexp.html.

[SPKI] Carl M. Ellison, et. al., Simple Public Key Certificate, http://www.pobox.com/~cme/spki.html.

[X509] ITU-T Recommendation X.509, The Directory: Authentication Framework, June 1997.