[xmlsec] Xmlsec preferring KeyValue element over explicitly loaded public keys?

Greg Vishnepolsky greg at adallom.com
Mon Nov 11 09:49:49 PST 2013

- Previous message: [xmlsec] decrypt document after encryption failed
- Next message: [xmlsec] Xmlsec preferring KeyValue element over explicitly loaded public keys?
- **Messages sorted by:** [date] [thread] [subject] [author]

Hello,

I've encountered very odd behavior of xmlsec which could be dangerous / considered a security issue.

Take a look at the following XML (from the URL below). It contains a KeyValue tag to be filled in during signature. It can be signed as follows:

```
$ curl https://gist.github.com/gregvish/7362993/raw/6979439b13056d9622a404be40fd49d56381d7cb/xmlsign2.xml > test.xml

$ openssl req -x509 -newkey rsa:2048 -keyout key.pem -out cert.pem -days 1000 -nodes

$ xmlsec1 --sign --privkey-pem key.pem test.xml > signed.xml
```

Naturally, it verifies:

```
$ xmlsec1 --verify --pubkey-cert-pem cert.pem signed.xml

OK
```

However, it also verifies with a completely unrelated public key! (when it certainly should not!)

```
$ openssl req -x509 -newkey rsa:2048 -keyout other-key.pem -out other-cert.pem -days 1000 -nodes

$ xmlsec1 --verify --pubkey-cert-pem other-cert.pem signed.xml

OK
```

And in fact, it would verify even if the verification is supposed to be against an X509 CA! There aren't any certificates in this XML at all!

```
$ xmlsec1 --verify --trusted-pem other-cert.pem signed.xml

OK
SignedInfo References (ok/all): 1/1
Manifests References (ok/all): 0/0
```

Apparently, xmlsec "prefers" the value in the KeyValue element over any other public key material that is explicitly provided to a KeysMngr!

In order to fix (work around) this behavior, apparently, the "--enabled-key-data" flag should be passed. This way, doing the first 2 examples works as expected:

```
$ xmlsec1 --verify --pubkey-cert-pem cert.pem --enabled-key-data rsa signed.xml

OK
$ xmlsec1 --verify --pubkey-cert-pem other-cert.pem --enabled-key-data rsa signed.xml

func=xmlSecOpenSSLEvpSignatureVerify:file=signatures.c:line=346:obj=rsa-sha1:subj=EVP_VerifyFinal:error=18:data do not match:signature do not match

FAIL
```

The same is true for the example with the x509 certificates. If "--enabled-key-data x509" is provided, a valid X509Data element is expected in the XML, and the KeyValue element is properly ignored.

I don't think this is reasonable default behavior. I'd even consider this is a security issue, at least in the documentation. I haven't found any documentation stating the importance of this flag. People using the library could be prone to making this mistake.

For instance, here is a guide (#2 on google for "using xmlsec1"): http://users.dcc.uchile.cl/~pcamacho/tutorial/web/xmlsec/xmlsec.html . There, as you can see, the given usage examples exhibit this vulnerable behavior.

This of course is not limited to the xmlsec1 command line utility, but is a problem in the xmlsec library itself. Looking at the C code examples, only 2 out of 4 verification examples are not vulnerable to this.

The first one is called "Verifying a signature with a single key". In that example, an xmlSecKeysMngr is not used. The following code loads the key directly into the DSigCtx:

```
    /* load public key */

    dsigCtx->signKey = xmlSecCryptoAppKeyLoad(...);
```

This code is not vulnerable because of this. The example "Verifying a signature with additional restrictions" is also not vulnerable, since it explicitly sets the allowed key data (as part of the additional restrictions).

All the other examples, that use a KeysMngr (but do not limit the key data) are vulnerable. For instance, you may test the above (signed) XML file with the example called "Verifying a signature with keys manager" . This is the example called "verify2.c" on the xmlsec website (http://www.aleksey.com/xmlsec/api/xmlsec-verify-with-keys-mngr.html).

```
$ openssl x509 -in cert.pem -pubkey -noout > pubkey.pem
```

```
$ openssl x509 -in other-cert.pem -pubkey -noout > other-pubkey.pem

$ ./verify2 signed.xml pubkey.pem

Signature is OK


So far so good...


$ ./verify2 signed.xml other-pubkey.pem

Signature is OK


Unexpected!


In order to fix this, the following patch needs to be applied to the
"verify2.c" example:


--- ex2.c       2013-11-11 18:16:36.136024683 +0200

+++ ex3.c       2013-11-11 19:28:26.825831754 +0200

@@ -254,6 +254,11 @@

         goto done;

     }


+    if(xmlSecPtrListAdd(&(dsigCtx->keyInfoReadCtx.enabledKeyData), BAD_CAST
xmlSecKeyDataRsaId) < 0) {

+        fprintf(stderr,"Error: failed to limit allowed key data\n");

+        goto done;

+    }

+

     /* Verify signature */

     if(xmlSecDSigCtxVerify(dsigCtx, node) < 0) {

         fprintf(stderr,"Error: signature verify\n");



With this patch, this is what happens:

$ ./verify2-fix signed.xml other-pubkey.pem

func=xmlSecOpenSSLEvpSignatureVerify:file=signatures.c:line=346:obj=rsa-sha1
:subj=EVP_VerifyFinal:error=18:data do not match:signature do not match

Signature is INVALID


Now it behaves as expected.

The code of this patch was taken from the "verify4.c" example, where the
limitation of the key data is described. I've looked around a bit, and in
fact, this is the only place where this feature is documented!


To sum up, I think that this issue needs to be addressed somehow. Either
this is better documented, or the default behavior is changed to be far less
"permissive".

Please tell me your thoughts on the matter, and what can/should/shot not be
done about it.


Thanks,

Greg


-------------- next part --------------
A non-text attachment was scrubbed...
Name: winmail.dat
Type: application/ms-tnef
Size: 9178 bytes
Desc: not available
URL: <http://www.aleksey.com/pipermail/xmlsec/attachments/20131111/317a1f94/attachment.bin>
```

- Previous message: [xmlsec] decrypt document after encryption failed
- Next message: [xmlsec] Xmlsec preferring KeyValue element over explicitly loaded public keys?
- **Messages sorted by:** [date] [thread] [subject] [author]
