strongSwan - Bug #3249

x509 plugin creates CA certificates with invalid Key Usage flags when compiling with GCC 9+ or clang 4+

06.11.2019 21:45 - Paul Wouters

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Tobias Brunner	**Estimated time:**	0.00 hour
Category:	pki		
Target version:	5.8.3		
Affected version:	5.8.1	**Resolution:**	Fixed

Description

Specifically, the NSS library using RFC 4945 IPsec profiles will reject the certificate validation because of the empty yet critical key usage section.

the commands used to generate the ca:

strongswan pki --gen --type ecdsa --size 384 > strongCAkey.der
strongswan pki --self --in strongCAkey.der --dn "C=CH, O=strongSwan, CN=strongSwan CA" --ca > strongCAcert.der

openssl x509 shows:

```
X509v3 extensions:
            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Key Usage: critical
                ....
            X509v3 Subject Key Identifier:
                D9:B1:43:EE:F9:A0:F9:D9:9C:D7:BC:0F:73:72:96:5B:5B:19:1D:3D
    Signature Algorithm: ecdsa-with-SHA384
```

Related issues:

Has duplicate Issue #3434: Windows client error 13801 with Ubuntu 20.04 stron...	**Closed**

Associated revisions

Revision 8ea13bbc - 27.01.2020 18:31 - Tobias Brunner

lgtm: Add query to detect problematic uses of chunk_from_chars()

GCC 9+ and clang 4+ (partially) optimize out usages of
chunk_from_chars() if the value is read outside of the block where the
macro is used. For instance:

```
chunk_t chunk = chunk_empty;
if (...) {
chunk = chunk_from_chars(0x01, 0x06);
}
/* do something with chunk */
```

The chunk_from_chars() macro expands to a chunk_t declaration, which is
technically only defined inside that block.

Still, with older GCC versions the fourth line was compiled to something
like this:

```
mov     WORD PTR [rsp+14], 1537 # 0x0106 in little-endian
lea     rdx, [rsp+14]
mov     ecx, 2
```

However, with GCC 9.1 and -O2 the first instruction might be omitted
(strangely the others usually were not, so the chunk pointed to whatever

was stored on the stack). It's not easily reproducible, so there are situations where the seemingly identical code is not optimized in this way.

This query should detect such problematic uses of the macro (definition and usage in different blocks).

References #3249.

Revision d5cf2d1f - 30.01.2020 18:18 - Tobias Brunner

tls-crypto: Fix usage of chunk_from_chars()

See 8ea13bbc5ccd for details.

References #3249.

Revision d16e8107 - 30.01.2020 18:18 - Tobias Brunner

pki: Remove unnecessary and problematic chunk_from_chars() usage in --signcrl

If the serial is not yet set, the same default value is set just below.

See 8ea13bbc5ccd for details on chunk_from_chars().

References #3249.

Revision 77643350 - 30.01.2020 18:18 - Tobias Brunner

x509: Replace problematic calls of chunk_from_chars() for keyUsage extension

As noted in 8ea13bbc5ccd newer compilers might optimize out the assignment leading to invalid values in the keyUsage extension (as the length was still set, the extension was encoded, just not with the intended values).

Fixes #3249.

Revision ef4113a4 - 30.01.2020 18:18 - Tobias Brunner

libtpmtss: Fix problematic usage of chunk_from_chars() in TSS2 implementations

See 8ea13bbc5ccd for details.

References #3249.

History

#1 - 07.11.2019 08:23 - Tobias Brunner

- Category set to pki

- Status changed from New to Feedback

CA certificates generated by the *x509* plugin have the CertificateSign and CRLsign flags set in the keyUsage extension. Since that's the only plugin able to generate certificates, I don't know what went wrong here. Can't reproduce it either; with the commands you gave I get this output:

```
       X509v3 extensions:
            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Key Usage: critical
                Certificate Sign, CRL Sign
            X509v3 Subject Key Identifier:
                5E:31:9C:71:4F:D7:67:85:C9:54:26:20:F6:C8:50:C8:A4:67:0E:E6
    Signature Algorithm: ecdsa-with-SHA384
```

#2 - 07.11.2019 18:22 - Paul Wouters

This problem seems to not happen on fedora29 or ubuntu 18, but happens on fedora30. Perhaps a different openssl version causing this? fedora30 uses 1.1.1c (while fedora29 uses 1.1.1d?). Ubuntu has 1.1.1-1ubuntu2.1~18.04.4 which i dont know if that is c or d.

#3 - 08.11.2019 08:39 - Tobias Brunner

> Perhaps a different openssl version causing this?

Hm, don't see how, as the *openssl* plugin can't encode certificates. So the encoding has to be the same on all platforms. Could you attach the generated certificate?

> fedora30 uses 1.1.1c (while fedora29 uses 1.1.1d?)

According to this page the version is 1.1.1d on both.

#4 - 23.01.2020 01:26 - G. G.

Paul Wouters wrote:

> Specifically, the NSS library using RFC 4945 IPsec profiles will reject the certificate validation because of the empty yet critical key usage section.
>
> the commands used to generate the ca:
>
> strongswan pki --gen --type ecdsa --size 384 > strongCAkey.der
> strongswan pki --self --in strongCAkey.der --dn "C=CH, O=strongSwan, CN=strongSwan CA" --ca > strongCAcert.der
>
> openssl x509 shows:
>
> X509v3 extensions:
> X509v3 Basic Constraints: critical
> CA:TRUE
> X509v3 Key Usage: critical
> ....
> X509v3 Subject Key Identifier:
> D9:B1:43:EE:F9:A0:F9:D9:9C:D7:BC:0F:73:72:96:5B:5B:19:1D:3D
> Signature Algorithm: ecdsa-with-SHA384

There is really something even ""uglier"" happening when setting output to PEM.

I also get an empty Key Usage:

```
            X509v3 Key Usage: critical
                 ....
```

BUT when repeating the pki --self command I get sometimes:

```
            X509v3 Key Usage: critical
                 ...K
```

OR

```
             X509v3 Key Usage: critical
                 ...q
```

OR

```
            X509v3 Key Usage: critical
                 Encipher Only
```

appears from time to time (but less often than ....)

This happens on x86-64 (F31) and on ARMv7 with OpenSSL 1.1.1d (on both).

Validation of certs fail on Windows10 because of the Key Usage badly set.

#5 - 23.01.2020 07:54 - Florian Friedrich

G. G. wrote:

> Paul Wouters wrote:
>
> > Specifically, the NSS library using RFC 4945 IPsec profiles will reject the certificate validation because of the empty yet critical key usage section.
> >
> > the commands used to generate the ca:

> strongswan pki --gen --type ecdsa --size 384 > strongCAkey.der
> strongswan pki --self --in strongCAkey.der --dn "C=CH, O=strongSwan, CN=strongSwan CA" --ca > strongCAcert.der
>
> openssl x509 shows:
>
> X509v3 extensions:
> X509v3 Basic Constraints: critical
> CA:TRUE
> X509v3 Key Usage: critical
> ....
> X509v3 Subject Key Identifier:
> D9:B1:43:EE:F9:A0:F9:D9:9C:D7:BC:0F:73:72:96:5B:5B:19:1D:3D
> Signature Algorithm: ecdsa-with-SHA384

> There is really something even ""uglier"" happening when setting output to PEM.
>
> I also get an empty Key Usage:
> [...]
> BUT when repeating the pki --self command I get sometimes:
> [...]
> OR
> [...]
> OR
> [...]
> appears from time to time (but less often than ....)
>
> This happens on x86-64 (F31) and on ARMv7 with OpenSSL 1.1.1d (on both).
>
> Validation of certs fail on Windows10 because of the Key Usage badly set.

Yes, I've seen similar output. I can add "+" and "[" to the list of characters after "...":

```
X509v3 Key Usage: critical
                  ...+

X509v3 Key Usage: critical
                  ...[
```

Happened (on first try) on a RasPi 4 running a current build of Ubuntu 20.04, which also has OpenSSL 1.1.1d.
On Ubuntu 19.10 (current stable version) I've had the "...." output as well. Ubuntu 19.10 uses OpenSSL 1.1.1c.

BTW. this also happens when issuing sub CA's using pki --issue --ca.

#6 - 23.01.2020 10:29 - Tobias Brunner

> There is really something even ""uglier"" happening when setting output to PEM.

The PEM format should really not make a difference, because that is just the Base64 encoding of the DER-encoded certificate (unless the Base64 encoder that has been the same for 12 years suddenly produces incorrect output).

> I also get an empty Key Usage:

Could you please attach some of these certificates.

> Yes, I've seen similar output. I can add "+" and "[" to the list of characters after "...":

Florian, please attach one or more of these certificates.

> Happened (on first try) on a RasPi 4 running a current build of Ubuntu 20.04, which also has OpenSSL 1.1.1d.
> On Ubuntu 19.10 (current stable version) I've had the "...." output as well. Ubuntu 19.10 uses OpenSSL 1.1.1c.

Again, the OpenSSL version is completely irrelevant to this process.

#7 - 23.01.2020 10:46 - Florian Friedrich

- File root-cert.pem added

- File sub-cert.pem added

> Florian, please attach one or more of these certificates.

Sure!

Btw, on macOS you can't even install the certificates in the keychain, because macOS declares them invalid due to this issue.

#8 - 23.01.2020 11:50 - Tobias Brunner

> > Florian, please attach one or more of these certificates.
>
> Sure!

Thanks. Yep, that encoding is wrong. Instead of 0106 the value is set to FF5B (in both certificates).

No idea why. The code sets the value directly before it is encoded (source:src/libstrongswan/plugins/x509/x509_cert.c#L2320). asn1_wrap simply copies the given value.

#9 - 23.01.2020 12:39 - Tobias Brunner

- Tracker changed from Issue to Bug

- Subject changed from pki tool creates root CA with empty yet critical Key Usage - breaking RFC 4945 processing to x509 plugin creates CA certificates with invalid Key Usage flags when compiling with GCC 9+ or clang 4+

- Assignee set to Tobias Brunner

- Target version set to 5.8.3

I did some experiments with different compilers (via https://godbolt.org).

While GCC 8.3 and before compiles the line keyUsageBits = chunk_from_chars(0x01, 0x06); to this:

```
        mov     WORD PTR [rsp+14], 1537 # 0x0106 in little-endian
        lea     rdx, [rsp+14]
        mov     ecx, 2
```

Starting with GCC 9.1 and when using at least -O1 the compiler doesn't assign a value, i.e. the first instruction above is missing. So the encoded value depends on whatever is stored on the stack at that location.

There actually was a similar issue I fixed in August (a4279fcc38). Unfortunately, in this case no warning was issued because the chunk is initialized at the start of the function.

The problem apparently is that the temporary chunk created with the chunk_from_chars() macro is technically not valid anymore after the if-block. So the compiler optimizes out the assignment (clang already started doing this with 4.0). The same is also true for chunk_from_thing() if it points to a variable inside a block if the chunk is used outside of it (no warning is issued in this case either).

#10 - 23.01.2020 20:22 - G. G.

I did put some

```
#pragma GCC optimize ("O0")
```

in src/libstrongswan/utils/chunk.h and recompiled -> no luck.

However, when compiled with CFLAGS="-O0" I got :

```
            X509v3 Key Usage: critical
                Certificate Sign, CRL Sign
```

Thanks for your fast reply!

#11 - 29.01.2020 13:47 - Tobias Brunner

Alright, I did some further analysis on this. There are a total of 1121 uses of this macro in our codebase. Most of them are in a static context (test

vectors, DH params, global constants etc.), which are not problematic. Of the remaining about 70 cases most are fine either because the result of the macro is used directly in a function call (e.g. asn1_wrap(..., chunk_from_chars(...));) or within the same or a nested block (e.g. if the assignment happens at the start of a function).

According to a CodeQL query I wrote (see here), there are only four problematic locations where the result of the macro is used in a parent or sibling block. One of them affects TLS (signature algorithm fallback if the client doesn't provide a list of supported algorithms), two affect the keyUsage extension in CA and CRL-signer certificates (the problem reported here) and one the default CRL serial number in pki --signcrl. There are actually two more, which are not found on lgtm.com because the affected component is currently not built there (only compiled code is analyzed). They are in the TSS2 adapters to set a default RSA exponent if the TPM doesn't provide one.

I've pushed fixes for these issues to the *3249-chunk-from-chars* branch. The query is also included and should be applied to new code by LGTM (triggered via Travis), which hopefully prevents this from happening in the future (at least in code that is built and analyzed).

#12 - 30.01.2020 18:21 - Tobias Brunner

- Status changed from Feedback to Closed

- Resolution set to Fixed

The fixes are now in master. I also changed the build on LGTM so that two additional plugins (botan, wolfssl) and libtmptss (TSS2 issue mentioned above) are also analyzed.

#13 - 21.04.2020 03:22 - Paul Wouters

Note I just installed 5.8.4 and still see the same issue on fedora rawhide and fedora 32, so the fix that went into 5.8.3 is not complete

#14 - 21.04.2020 03:42 - Paul Wouters

Same for centos7 with gcc-4.8.5 testing with 5.8.4

using both rpmbuild with rpm build macros, and by using manual compile using configure without any kind of directives for cflags/ldflags or anything.

simple test case:

```
PKI=/usr/local/strongswan/bin/pki
$PKI --gen --type ecdsa --size 384 > strongCAkey.der
$PKI --self --in strongCAkey.der --dn "C=CH, O=strongSwan, CN=strongSwan CA" --ca > strongCAcert.der
$PKI --gen --type ecdsa --size 384 > strongWestKey.der
$PKI --pub --in strongWestKey.der | $PKI --issue --cacert strongCAcert.der --cakey strongCAkey.der --dn "C=CH, O=strongSwan, CN=strongWest"
--flag serverAuth --san west.testing.libreswan.org > strongWestCert.der

openssl x509 -inform der -outform pem -in strongWestCert.der -out strongWestCert.pem
openssl x509 -in strongWestCert.pem -noout -text | grep "Key Usage"
```

This should show Key Usage and Extended Key Usage, but will not show Key Usage because the entire section is missing

#15 - 21.04.2020 09:41 - Tobias Brunner

> This should show Key Usage and Extended Key Usage, but will not show Key Usage because the entire section is missing

strongSwan only encodes a *keyUsage* extension in CA certificates (CertificateSign/CRLsign) and CRLs (only CRLsign), not in end-entity certificates.

#16 - 21.04.2020 15:24 - Paul Wouters

Tobias Brunner wrote:

> > This should show Key Usage and Extended Key Usage, but will not show Key Usage because the entire section is missing
>
> strongSwan only encodes a *keyUsage* extension in CA certificates (CertificateSign/CRLsign) and CRLs (only CRLsign), not in end-entity certificates.

ah yes. I see. And it is allowed by RFC-4945 (https://tools.ietf.org/html/rfc4945#section-5.1.3.2)

So that means indeed the bug is fixed. sorry for the noise

#17 - 04.05.2020 15:35 - Tobias Brunner

- Has duplicate Issue #3434: Windows client error 13801 with Ubuntu 20.04 strongSwan server added

Files

root-cert.pem	1.74 KB	23.01.2020	Florian Friedrich
sub-cert.pem	1.78 KB	23.01.2020	Florian Friedrich