strongSwan - Bug #509

"ipsec listcerts" misinterpretes UTCTime information from x509 certificates

05.02.2014 04:34 - Karsten Hohmeier

Status:	Closed	**Start date:**	05.02.2014
Priority:	Normal	**Due date:**	
Assignee:	Tobias Brunner	**Estimated time:**	0.00 hour
Category:	libstrongswan		
Target version:	5.1.2		
Affected version:	5.1.1	**Resolution:**	Fixed

Description

Hello everyone. I used Openssl 1.0.1f to generate x509 certificates to use with strongSwan 5.1.1 on debian.
I created the certs with the greatest validity period possible using UTCTime (see below).

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 4096 (0x1000)
    Signature Algorithm: sha512WithRSAEncryption
        Issuer: (...)
        Validity
            Not Before: Jan  1 00:00:01 1950 GMT
            Not After : Dec 31 23:59:59 2049 GMT
    (...)
```

The cert works fine but "ipsec listcerts" diplays the validity perid wrong.

```
# ipsec listcerts

List of X.509 End Entity Certificates:

  altNames:  (...)
  subject:  (...)
  issuer:   (...)
  serial:    10:00
  validity:  not before Jan 19 04:14:07 2038, not valid yet (valid in 8749 days)
             not after  Jan 01 00:59:59 2050, ok
```

It might just be a cosmetic flaw but if the same logic applies to the validation of client certificates there could be a real bug or compatibility issue lurking.

FYI:

```
RFC 5280            PKIX Certificate and CRL Profile           May 2008

4.1.2.5.1.  UTCTime

   The universal time type, UTCTime, is a standard ASN.1 type intended
   for representation of dates and time.  UTCTime specifies the year
   through the two low-order digits and time is specified to the
   precision of one minute or one second.  UTCTime includes either Z
   (for Zulu, or Greenwich Mean Time) or a time differential.

   For the purposes of this profile, UTCTime values MUST be expressed in
   Greenwich Mean Time (Zulu) and MUST include seconds (i.e., times are
   YYMMDDHHMMSSZ), even where the number of seconds is zero.  Conforming
   systems MUST interpret the year field (YY) as follows:

      Where YY is greater than or equal to 50, the year SHALL be
      interpreted as 19YY; and
      Where YY is less than 50, the year SHALL be interpreted as 20YY.
```

Associated revisions

Revision ebc665be - 12.02.2014 13:54 - Tobias Brunner

asn1: Support dates before 1970-01-01 (i.e. when time_t gets negative)

On x86 we allow "overflows" around 1969/1970 but not for other dates.

Fixes #509.

History

#1 - 05.02.2014 18:23 - Tobias Brunner

- Category set to libstrongswan

- Status changed from New to Feedback

- Assignee set to Tobias Brunner

The first question, of course, is why would anybody do something like this. I mean, honestly, what's the advantage of having certificates that are valid so long in the past? (Other than perhaps time travel ;-)

The ASN.1 date parser previously interpreted all negative time_t values, that is, even legitimate dates before 1970, as overflows on account of time_t being too small on x86 systems to handle large dates (i.e. after Tue, 19 Jan 2038 03:14:07 UTC = 0x7fffffff). It simply returned that maximum date in such a case.

I pushed a fix (9e847bc8) to the *asn1-time* branch of our repository, which adds support for dates before 1970, but still catches overflows on x86 (e.g. for dates before Fri, 13 Dec 1901 20:45:52 UTC = 0x80000000 or after the date listed above).

Additionally, we could define and return TIME_32_BIT_SIGNED_MIN for dates earlier than the minimum and handle that value separately when listing certificates with ipsec listcerts, but since such dates are rather uncommon I'm not sure if it is worth the effort.

#2 - 05.02.2014 19:04 - Karsten Hohmeier

- File cert.jpg added

Hello,

Why anybody would do this?

1) Because I can ;)
2) Push the limits!
3) I once deployed certificates onto embedded systems that had no reliable RTC or any means of time synchronisation via NTP. So I just didn't care for validity just for correct signatures. I even generated certs for client authentication to connect to those systems. To avoid any validity issues I just set it to the biggest period possible (with -startdate -enddate options in OpenSSL), effectively creating "everlasting" certificates (in terms of UTCTime).

Why you should fix this?

1) Even Microsoft can handle those dates correctly (see attachment). You don't want to be worse than them, do you?
2) RFC compliance
3) not confusing users (like me)

Best regards

Karsten Hohmeier

#3 - 12.02.2014 16:09 - Tobias Brunner

- Status changed from Feedback to Resolved

- Target version set to 5.1.2

- Resolution set to Fixed

#4 - 28.02.2014 08:43 - Tobias Brunner

- Status changed from Resolved to Closed

Files

cert.jpg	61.4 KB	05.02.2014	Karsten Hohmeier