strongSwan - Bug #688

Reported key length is incorrect for ECDSA keys using ECC Brainpool curves

26.08.2014 13:07 - F DCG

Status:	Closed	**Start date:**	26.08.2014
Priority:	Normal	**Due date:**	
Assignee:	Tobias Brunner	**Estimated time:**	0.00 hour
Category:	libstrongswan		
Target version:	5.2.1		
Affected version:	5.2.0	**Resolution:**	Fixed

Description

Hi,

It seems like the openssl 1.0.2b2 brainpool curves are not handled by ss 5.2, due to a ECDSA 0 bit size problem.
openssl ecparam -genkey -name secp521r1 -out signCAKey.key -noout -no_seed -rand .rnd
openssl ecparam -genkey -name brainpoolP512t1 -out signCAKey.key -noout -no_seed -rand .rnd

The normal line concerns a nist curve, which goes fine. The bold line concerns the bp curve and that produces unusable results. Is this to be expected or not?
Compile ss 5.2 from source including the openssl plugin.

The second question concerns using ssh keys opposed to the openssl versions. It there a snip somewhere outlining the way to use those?

Tia,
Fermin DCG

Associated revisions

Revision 58184676 - 02.09.2014 08:15 - Tobias Brunner

openssl: Report correct key length for EC keys when not using NIST curves

Fixes #688.

History

#1 - 26.08.2014 17:03 - F DCG

- File strongswan.log added

Below are the nist and bp creation and verification results.
The bold text shows the bp curve. The private key size is also shown correctly (according to openssl).

moon@base:~/myCA1L$ openssl ecparam -genkey -name secp521r1 -out nistkey
moon@base:~/myCA1L$ openssl ec -text -noout -in nistkey
read EC key
Private-Key: (521 bit)
priv:
36:b6:5a:25:d9:10:30:60:31:08:d1:27:18:9d:90:
1f:a2:ae:6a:a9:fa:d8:1a:0a:5e:ad:08:18:1b:05:
ec:a1:2d:d3:71:6e:28:7d:52:8c:ad:f7:ae:79:63:
b2:e9:f3:f0:63:b4:73:02:e8:8d:f5:7e:18:0e:0c:
a0:b7:84:b0:23
pub:
04:00:03:8a:37:1b:12:dd:d9:e4:0f:93:47:af:48:
d1:0d:ac:25:59:06:58:6f:b1:57:df:5f:03:99:40:
08:6d:49:08:37:e0:66:e8:f7:07:05:20:64:ca:fd:
df:aa:89:d5:8d:cc:3f:b1:80:de:58:1a:e2:c1:07:
87:e5:72:4a:f9:96:71:01:71:f6:e0:73:bc:df:e4:
a9:01:e3:63:85:b4:5a:bd:33:b8:7c:3f:ba:6b:05:
ca:be:41:41:be:91:6c:96:ff:f4:11:1c:a2:8e:70:
fe:09:a7:55:82:9d:1a:1c:d2:a7:70:70:31:f5:d2:
25:0f:2d:a6:c5:78:a3:dc:21:40:a3:9f:45
ASN1 OID: secp521r1
NIST CURVE: P-521

*moon@base:~/myCA1L$ openssl ecparam -genkey -name brainpoolP512t1 -out bpkey
moon@base:~/myCA1L$ openssl ec -text -noout -in bpkey
read EC key
Private-Key: (512 bit)
priv:
00:a2:21:72:21:c6:b5:b3:a9:6a:26:57:42:3f:97:
fe:64:d7:a5:0a:06:08:e8:17:02:2b:5f:d7:83:c4:
e2:a0:7e:8a:58:0e:3f:e3:52:f5:44:2b:86:bd:1c:
2c:0a:23:31:1c:d9:0d:63:3c:38:30:81:79:eb:d4:
56:3d:ec:12:21
pub:
04:7a:41:39:b5:06:83:27:59:20:54:4c:5d:17:90:
41:96:99:a3:63:da:41:83:97:fb:10:fe:d8:a1:4e:
9f:a3:82:f9:6b:21:a5:af:52:65:55:12:4f:a1:69:
9d:9f:15:8d:15:1b:36:1c:e3:56:21:42:9b:6a:f9:
42:6d:4e:eb:d8:34:88:b1:a7:99:0b:f9:c9:09:d9:
a0:59:48:08:3b:4a:df:ab:87:2d:6e:64:b1:71:6e:
77:6b:47:b4:22:6a:27:ea:3a:06:68:9c:5e:a0:63:
9a:0e:55:10:22:58:bd:6b:96:94:c9:5e:f3:53:35:
0d:9d:0e:33:68:af:06:42:22
ASN1 OID: brainpoolP512t1 *

Attached is a C&P of the client en server logs, incl some comments.
As noted in the log as well. All the code comes from the same source. The only difference is the bp vs the nist curve.
This **also** applies to the conf files. Meaning that the working conf for the nist curve is equal to that of the bp curve.

#2 - 26.08.2014 18:45 - Tobias Brunner

- File 0001-openssl-Report-correct-key-length-for-EC-keys-when-n.patch added

- Tracker changed from Issue to Bug

- Subject changed from Question re. openssl brainpool curves to Reported key length is incorrect for ECDSA keys using ECC Brainpool curves

- Status changed from New to Feedback

- Assignee set to Tobias Brunner

- Target version set to 5.2.1

> It seems like the openssl 1.0.2b2 brainpool curves are not handled by ss 5.2, due to a ECDSA 0 bit size problem.

You definitely can't use certificates/keys with ECC Brainpool curves for IKE authentication (i.e. as end-entity certificates), only three NIST curves (256, 384 and 521 bits) can be used for that (RFC 4754). Until draft-kivinen-ipsecme-signature-auth makes it into an RFC we can't do much about it.

There could be other problems as we haven't focused much on other curves (for ECDSA that is, you can use the ECC Brainpool curves for ECDH since 5.1.1). In particular, the get_keysize() function was hard-coded to the three NIST curves. The attached patch fixes this. With it applied I can use EC keys with *brainpoolP512t1* generated by OpenSSL to build and verify a PKI using the ipsec pki utility. So regarding IKE it should be possible to use ECC Brainpool curves for the CA certificates and use one of the NIST curves for the end-entity certificates.

> The second question concerns using ssh keys opposed to the openssl versions. It there a snip somewhere outlining the way to use those?

The private keys can be configured in ipsec.secrets with the RSA and ECDSA keywords.

The public keys are loaded with the *left|rightsigkey* keyword in ipsec.conf. Either directly configure the base64 encoding of the key (using the *ssh:* prefix) or the path to the file.

#3 - 26.08.2014 21:59 - F DCG

Thank you Tobias for your swift reply.

Just to make sure i understand correctly i'll try to summarize your reply:
- Applying the patch to the 5.2.0 code will allow me to use the bp curves for the CA certificates.
- The endpoints (the actual server and the user-clients) will have to make do with one of the 3 nist curves
- The private ssh keys are clear and public versions 'go into' the conf. Should they be signed with a SSH CA key or just be plain vanilla?

One question re. the ssh keys. Are they allowed to be of the ed25519 nature?
And, am i correct in assuming that if i use the SSH keys i can do without the openssl ones?

Thank you for the patch. Unfortunately i have no experience in that area so below is the result of what i did.
*moon@base:~/strongswan-5.2.0/src/libstrongswan/plugins/openssl$ patch openssl_ec_private_key.c <
0001-openssl-Report-correct-key-length-for-EC-keys-when-n.patch
patching file openssl_ec_private_key.c*

patching file openssl_ec_private_key.c
Reversed (or previously applied) patch detected! Assume -R? [n] y
Hunk #1 succeeded at 181 with fuzz 2 (offset 2 lines).
moon@base:~/strongswan-5.2.0/src/libstrongswan/plugins/openssl$ patch openssl_ec_public_key.c <
0001-openssl-Report-correct-key-length-for-EC-keys-when-n.patch
patching file openssl_ec_public_key.c
Reversed (or previously applied) patch detected! Assume -R? [n] y
Hunk #1 succeeded at 179 with fuzz 2 (offset -2 lines).
patching file openssl_ec_public_key.c

Would the above be the correct way?
Assuming a correct patch should the x509 CA certificates then show the correct key sizes (after re-compile etc off course)?

Tia,
Fermin DCG

#4 - 27.08.2014 09:43 - Tobias Brunner

- Category changed from pki to libstrongswan

> Just to make sure i understand correctly i'll try to summarize your reply:
> - Applying the patch to the 5.2.0 code will allow me to use the bp curves for the CA certificates.

Theoretically yes, but I only tested with pki --verify.

> - The endpoints (the actual server and the user-clients) will have to make do with one of the 3 nist curves

Correct.

> - The private ssh keys are clear and public versions 'go into' the conf. Should they be signed with a SSH CA key or just be plain vanilla?

No, these are plain SSH keys, as generated by ssh-keygen. Not sure what you mean with "private ssh keys are clear". The public keys are either copied into the config file (e.g. *rightsigkey=ssh:AAAAB3NzaC1y...*) or you just refer to the file that contains the key (e.g. *rightsigkey=/path/to/id_rsa.pub* - the path can also be relative to ipsec.d/certs).

> One question re. the ssh keys. Are they allowed to be of the ed25519 nature?

No, IKEv2 does currently not support that.

> And, am i correct in assuming that if i use the SSH keys i can do without the openssl ones?

Yes. But you will have to configure all the keys manually. If you have many peers it's easier to work with a PKI as you then just have to install the CA certificate to accept all certificates issued by that CA.

```
moon@base:~/strongswan-5.2.0/src/libstrongswan/plugins/openssl$ patch openssl_ec_private_key.c < 0001-open
ssl-Report-correct-key-length-for-EC-keys-when-n.patch
patching file openssl_ec_private_key.c
patching file openssl_ec_private_key.c
Reversed (or previously applied) patch detected!  Assume -R? [n] y
Hunk #1 succeeded at 181 with fuzz 2 (offset 2 lines).
```

> Would the above be the correct way?

No. The simplest way to apply the patch is this:

```
moon@base:~/strongswan-5.2.0$ patch -p1 < 0001-openssl-Report-correct-key-length-for-EC-keys-when-n.patch
```

It should apply cleanly, if you get errors you might have to extract the tarball again.

> Assuming a correct patch should the x509 CA certificates then show the correct key sizes (after re-compile etc off course)?

Yes.

#5 - 02.09.2014 08:16 - Tobias Brunner

- Status changed from Feedback to Closed

- Resolution set to Fixed

Files

strongswan.log	16.2 KB	26.08.2014	F DCG
0001-openssl-Report-correct-key-length-for-EC-keys-when-n.patch	1.91 KB	26.08.2014	Tobias Brunner