strongSwan - Bug #1076

Multiple policies between strongswan and racoon.

19.08.2015 12:58 - Alexander Velkov

Status:	Closed	**Start date:**	18.08.2015
Priority:	Normal	**Due date:**	
Assignee:	Tobias Brunner	**Estimated time:**	0.00 hour
Category:	interoperability		
Target version:	5.3.3		
Affected version:	5.3.2	**Resolution:**	Fixed

Description

Hello,

I try to configure an IPsec tunnel between two peers in the same LAN - one is running StrongSwan 5.3.2 (Peer1) the other racoon (ipsec-tools v0.8.2) (Peer2).
I configure two policies for the tunnel, the policies come UP and everything seems fine. BUT traffic flows only through one of the policies.

IPsec tunnel:

```
10.0.1.1/32 | ---  192.168.3.8 (Peer1) <-> (Peer2) 192.168.3.4  --- | 10.0.1.2/32
10.0.2.1/32 |                                                       | 10.0.2.2/32
```

I produce traffic by simply pinging the other side of the policy (e.g. *ping -I 10.0.1.1 10.0.1.2*).
If I remove any one of the two policies, then the remaining policy is successfully established and traffic flows without errors.

I am not sure if this behavior is a bug either on the racoon or on the StrongSwan side, or the configuration is simply wrong. Any help is appreciated.

Peer1 configuration and status:

```
--- strongswan.conf

charon {
        i_dont_care_about_security_and_use_aggressive_mode_psk=yes
        cisco_unity=no
        install_routes=no
        interfaces_use=eth0
        retransmit_tries=2
        threads=32
        reauth=yes
        forceencaps=no
        mobike=no
        rekey=yes
        installpolicy=yes
        fragmentation=yes
        closeaction=none
        syslog {
                identifier=charon
                daemon {
                        default=1
                }
        }
}

--- ipsec.conf

conn ipsectest
        left=192.168.3.8
        right=192.168.3.4
        leftauth=psk
        rightauth=psk
        aggressive=no
        auto=ignore
        keyexchange=ikev1
        compress=no
        type=tunnel
        margintime=540s
        ike=aes256-sha1-modp1536!
        ikelifetime=4200s
        esp=aes256-sha1-modp1536!
        lifetime=3600s

conn ipsectest$0
        leftsubnet=10.0.1.1/32[%any/%any]
        rightsubnet=10.0.1.2/32[%any/%any]
        auto=route
        keyingtries=1
        also=ipsectest

conn ipsectest$1
        leftsubnet=10.0.2.1/32[%any/%any]
        rightsubnet=10.0.2.2/32[%any/%any]
        auto=route
        keyingtries=1
        also=ipsectest

--- ip xfrm policy

src 10.0.2.2/32 dst 10.0.2.1/32 
        dir fwd priority 2819 
        tmpl src 192.168.3.4 dst 192.168.3.8
                proto esp reqid 2 mode tunnel
src 10.0.2.2/32 dst 10.0.2.1/32 
        dir in priority 2819 
        tmpl src 192.168.3.4 dst 192.168.3.8
                proto esp reqid 2 mode tunnel
src 10.0.2.1/32 dst 10.0.2.2/32 
        dir out priority 2819 
        tmpl src 192.168.3.8 dst 192.168.3.4
                proto esp reqid 2 mode tunnel
src 10.0.1.2/32 dst 10.0.1.1/32 
        dir fwd priority 2819 
        tmpl src 192.168.3.4 dst 192.168.3.8
                proto esp reqid 1 mode tunnel
src 10.0.1.2/32 dst 10.0.1.1/32 
        dir in priority 2819 
        tmpl src 192.168.3.4 dst 192.168.3.8
                proto esp reqid 1 mode tunnel
src 10.0.1.1/32 dst 10.0.1.2/32 
        dir out priority 2819 
        tmpl src 192.168.3.8 dst 192.168.3.4
                proto esp reqid 1 mode tunnel

--- ipsec statusall

ipsec stroke statusall-nb
Status of IKE charon daemon (weakSwan 5.3.2, Linux 3.10.45, i686):
  uptime: 14 minutes, since Aug 19 09:29:20 2015
  malloc: sbrk 176128, mmap 0, used 158496, free 17632
  worker threads: 27 of 32 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 2
  loaded plugins: charon aes des sha1 sha2 md5 random nonce x509 revocation constraints pubkey pkc
s1 pkcs7 pkcs8 pkcs12 pgp dnskey pem openssl fips-prf gmp xcbc hmac attr kernel-netlink resolve so
cket-default stroke updown xauth-generic unity
Listening IP addresses:
  192.168.3.8
  fc00:9999::8
Connections:
 ipsectest$0:  192.168.3.8...192.168.3.4  IKEv1
 ipsectest$0:   local:  [192.168.3.8] uses pre-shared key authentication
 ipsectest$0:   remote: [192.168.3.4] uses pre-shared key authentication
 ipsectest$0:   child:  10.0.1.1/32 === 10.0.1.2/32 TUNNEL
 ipsectest$1:   child:  10.0.2.1/32 === 10.0.2.2/32 TUNNEL
Routed Connections:
 ipsectest$1{2}:  ROUTED, TUNNEL, reqid 2
 ipsectest$1{2}:   10.0.2.1/32 === 10.0.2.2/32
 ipsectest$0{1}:  ROUTED, TUNNEL, reqid 1
 ipsectest$0{1}:   10.0.1.1/32 === 10.0.1.2/32
Security Associations (1 up, 0 connecting):
 ipsectest$0[1]: ESTABLISHED 14 minutes ago, 192.168.3.8[192.168.3.8]...192.168.3.4[192.168.3.4]
 ipsectest$0[1]: IKEv1 SPIs: 7ef3604b59f7f110_i* ab2ce7324524e95c_r, pre-shared key reauthenticati
on in 46 minutes
 ipsectest$0[1]: IKE proposal: AES_CBC_256/HMAC_SHA1_96/PRF_HMAC_SHA1/MODP_1536
 ipsectest$0{3}:  INSTALLED, TUNNEL, reqid 1, ESP SPIs: c8c8d5bb_i 02d3e7a7_o
 ipsectest$0{3}:  AES_CBC_256/HMAC_SHA1_96, 0 bytes_i, 252 bytes_o (3 pkts, 859s ago), rekeying in
 36 minutes
 ipsectest$0{3}:   10.0.1.1/32 === 10.0.1.2/32
 ipsectest$1{4}:  INSTALLED, TUNNEL, reqid 2, ESP SPIs: cc8d3c06_i 04c3d7b0_o
 ipsectest$1{4}:  AES_CBC_256/HMAC_SHA1_96, 336 bytes_i (4 pkts, 853s ago), 84 bytes_o (1 pkt, 853
s ago), rekeying in 36 minutes
 ipsectest$1{4}:   10.0.2.1/32 === 10.0.2.2/32
```

Peer2 configuration and status:

```
--- setkey.conf

flush;
spdflush;

spdadd 10.0.1.2/32 10.0.1.1/32 any -P out ipsec
        esp/tunnel/192.168.3.4-192.168.3.8/require;
spdadd 10.0.1.1/32 10.0.1.2/32 any -P in ipsec
        esp/tunnel/192.168.3.8-192.168.3.4/require;

spdadd 10.0.2.2/32 10.0.2.1/32 any -P out ipsec
        esp/tunnel/192.168.3.4-192.168.3.8/require;
spdadd 10.0.2.1/32 10.0.2.2/32 any -P in ipsec
        esp/tunnel/192.168.3.8-192.168.3.4/require;

--- racoon.conf

listen {
        strict_address;
        isakmp 192.168.3.4 [500];
        isakmp_natt 192.168.3.4 [4500];
}

remote "ipsectest"  {
        remote_address 192.168.3.8;
        my_identifier address;
        peers_identifier address;
        nat_traversal on;
        script "phase1.sh" phase1_down;
        script "phase1.sh" phase1_up;
        proposal_check obey;
        exchange_mode main;
        proposal {
                lifetime time 4200 sec;
                authentication_method pre_shared_key;
                encryption_algorithm aes256;
                hash_algorithm sha1;
                dh_group modp1536;
        }
}
```

```
sainfo address 10.0.1.2/32 any address 10.0.1.1/32 any {
        pfs_group modp1536;
        lifetime time 3600 sec;
        encryption_algorithm aes256;
        authentication_algorithm hmac_sha1;
        compression_algorithm deflate;
}

sainfo address 10.0.2.2/32 any address 10.0.2.1/32 any {
        pfs_group modp1536;
        lifetime time 3600 sec;
        encryption_algorithm aes256;
        authentication_algorithm hmac_sha1;
        compression_algorithm deflate;
}

--- ip xfrm policy

src 10.0.2.1/32 dst 10.0.2.2/32
        dir fwd priority 2147483648
        tmpl src 192.168.3.8 dst 192.168.3.4
                proto esp reqid 0 mode tunnel
src 10.0.2.1/32 dst 10.0.2.2/32
        dir in priority 2147483648
        tmpl src 192.168.3.8 dst 192.168.3.4
                proto esp reqid 0 mode tunnel
src 10.0.2.2/32 dst 10.0.2.1/32
        dir out priority 2147483648
        tmpl src 192.168.3.4 dst 192.168.3.8
                proto esp reqid 0 mode tunnel
src 10.0.1.1/32 dst 10.0.1.2/32
        dir fwd priority 2147483648
        tmpl src 192.168.3.8 dst 192.168.3.4
                proto esp reqid 0 mode tunnel
src 10.0.1.1/32 dst 10.0.1.2/32
        dir in priority 2147483648
        tmpl src 192.168.3.8 dst 192.168.3.4
                proto esp reqid 0 mode tunnel
src 10.0.1.2/32 dst 10.0.1.1/32
        dir out priority 2147483648
        tmpl src 192.168.3.4 dst 192.168.3.8
                proto esp reqid 0 mode tunnel

--- racoonctl show-sa esp

192.168.3.4 192.168.3.8
        esp mode=tunnel spi=3431808006(0xcc8d3c06) reqid=0(0x00000000)
        E: aes-cbc  2415c97b 0e80d4a1 301f379c d5967fee cd65c16c af6b1d1d 19ce3746 69c9bc5e
        A: hmac-sha1  86a9638b 11829553 8c22fbf3 7e0342b9 539c17bb
        seq=0x00000000 replay=4 flags=0x00000000 state=mature
        created: Aug 19 19:33:09 2015   current: Aug 19 19:49:38 2015
        diff: 989(s)    hard: 3600(s)   soft: 2880(s)
        last: Aug 19 19:33:15 2015      hard: 0(s)      soft: 0(s)
        current: 336(bytes)     hard: 0(bytes)  soft: 0(bytes)
        allocated: 4    hard: 0 soft: 0
        sadb_seq=1 pid=2284 refcnt=0
192.168.3.8 192.168.3.4
        esp mode=tunnel spi=79943600(0x04c3d7b0) reqid=0(0x00000000)
        E: aes-cbc  b0ab1e95 55e68fde ae9bf5b4 eb5577b1 615bb524 4b05c2bf ea3999e8 9388589f
        A: hmac-sha1  9d4b86b7 ce225e04 98dfab0e f04a5cb4 1035e14c
        seq=0x00000000 replay=4 flags=0x00000000 state=mature
        created: Aug 19 19:33:09 2015   current: Aug 19 19:49:38 2015
        diff: 989(s)    hard: 3600(s)   soft: 2880(s)
        last: Aug 19 19:33:23 2015      hard: 0(s)      soft: 0(s)
        current: 84(bytes)      hard: 0(bytes)  soft: 0(bytes)
        allocated: 1    hard: 0 soft: 0
        sadb_seq=2 pid=2284 refcnt=0
192.168.3.4 192.168.3.8
        esp mode=tunnel spi=3368605115(0xc8c8d5bb) reqid=0(0x00000000)
        E: aes-cbc  78c82a56 895ea1b5 a5d3e116 6b17c7e9 20a2e77a 8c029351 5d5b019e fc8bd909
        A: hmac-sha1  594a4679 62083e3a 3fa29117 981ed9c4 4295db64
        seq=0x00000000 replay=4 flags=0x00000000 state=mature
        created: Aug 19 19:33:07 2015   current: Aug 19 19:49:38 2015
        diff: 991(s)    hard: 3600(s)   soft: 2880(s)
        last:                           hard: 0(s)      soft: 0(s)
        current: 0(bytes)       hard: 0(bytes)  soft: 0(bytes)
        allocated: 0    hard: 0 soft: 0
        sadb_seq=3 pid=2284 refcnt=0
192.168.3.8 192.168.3.4
        esp mode=tunnel spi=47441831(0x02d3e7a7) reqid=0(0x00000000)
        E: aes-cbc  3266df28 d51a7c40 53ff84fd 270079c6 bf8c49d1 96a008be b0740cf6 1d5d9514
        A: hmac-sha1  4a3bea13 8fff5eba f00a5fd8 e2b3b1d6 8ea509c8
        seq=0x00000000 replay=4 flags=0x00000000 state=mature
        created: Aug 19 19:33:07 2015   current: Aug 19 19:49:38 2015
        diff: 991(s)    hard: 3600(s)   soft: 2880(s)
        last: Aug 19 19:33:15 2015      hard: 0(s)      soft: 0(s)
        current: 252(bytes)     hard: 0(bytes)  soft: 0(bytes)
        allocated: 3    hard: 0 soft: 0
        sadb_seq=0 pid=2284 refcnt=0
```

I also noticed that it matters which peer initiates the tunnel:

- StrongSwan initiates the tunnel - both policies successfully come UP, traffic only through one of the policies
- racoon initiates the tunnel - one policy comes UP but the second policy ends stuck with the following log on StrongSwan side:

```
Aug 19 10:32:55 Peer1 info  charon: [  ENC] parsed INFORMATIONAL_V1 request 2460648804 [ HASH N(INITIAL_CONTACT) ]
Aug 19 10:32:55 Peer1 info  charon: [  CFG] received stroke: loglevel 1 for any
Aug 19 10:32:55 Peer1 info  charon: [  NET] received packet: from 192.168.3.4[500] to 192.168.3.8[500] (364 bytes)
Aug 19 10:32:55 Peer1 info  charon: [  ENC] parsed QUICK_MODE request 3246901756 [ HASH SA No KE ID ID ]
Aug 19 10:32:56 Peer1 info  charon: [  ENC] generating QUICK_MODE response 3246901756 [ HASH SA No KE ID ID ]
Aug 19 10:32:56 Peer1 info  charon: [  NET] sending packet: from 192.168.3.8[500] to 192.168.3.4[500] (380 bytes)
Aug 19 10:32:56 Peer1 info  charon: [  NET] received packet: from 192.168.3.4[500] to 192.168.3.8[500] (364 bytes)
Aug 19 10:32:56 Peer1 info  charon: [  ENC] parsed QUICK_MODE request 2849100021 [ HASH SA No KE ID ID ]
Aug 19 10:32:56 Peer1 info  charon: [  IKE] CHILD_SA ipsectest$0{3} established with SPIs c26049cf_i 0d8c5ab7_o and TS 10.0.1.1/32 === 10.0.1.2/32
Aug 19 10:32:56 Peer1 info  charon: [  NET] received packet: from 192.168.3.4[500] to 192.168.3.8[500] (60 bytes)
Aug 19 10:32:56 Peer1 info  charon: [  ENC] parsed QUICK_MODE request 3246901756 [ HASH ]
Aug 19 10:32:56 Peer1 info  charon: [  IKE] sa payload missing
Aug 19 10:32:56 Peer1 info  charon: [  ENC] generating INFORMATIONAL_V1 request 984206471 [ HASH N(CRIT) ]
Aug 19 10:32:56 Peer1 info  charon: [  NET] sending packet: from 192.168.3.8[500] to 192.168.3.4[500] (76 bytes)
Aug 19 10:33:06 Peer1 info  charon: [  NET] received packet: from 192.168.3.4[500] to 192.168.3.8[500] (364 bytes)
Aug 19 10:33:06 Peer1 info  charon: [  ENC] invalid HASH_V1 payload length, decryption failed?
Aug 19 10:33:06 Peer1 info  charon: [  ENC] could not decrypt payloads
Aug 19 10:33:06 Peer1 info  charon: [  IKE] message parsing failed
Aug 19 10:33:06 Peer1 info  charon: [  ENC] generating INFORMATIONAL_V1 request 1236998600 [ HASH N(PLD_MAL) ]
Aug 19 10:33:06 Peer1 info  charon: [  NET] sending packet: from 192.168.3.8[500] to 192.168.3.4[500] (76 bytes)
Aug 19 10:33:06 Peer1 info  charon: [  IKE] QUICK_MODE request with message ID 2849100021 processing failed
```

Associated revisions

Revision 4de361d9 - 20.08.2015 19:13 - Tobias Brunner

ikev1: Fix handling of overlapping Quick Mode exchanges

In some cases the third message of a Quick Mode exchange might arrive
after the first message of a subsequent Quick Mode exchange. Previously
these messages were handled incorrectly and the second Quick Mode
exchange failed.

Some implementations might even try to establish multiple Quick Modes
simultaneously, which is explicitly allowed in RFC 2409. We don't fully
support that, though, in particular in case of retransmits.

Fixes #1076.

Revision 37a22a16 - 29.10.2015 16:03 - Tobias Brunner

ikev1: Avoid fourth QM message if third QM messages of multiple exchanges are handled delayed

If we haven't received the third QM message for multiple exchanges the
return value of NEED_MORE for passive tasks that are not responsible for
a specific exchange would trigger a fourth empty QM message.

Fixes: 4de361d92c54 ("ikev1: Fix handling of overlapping Quick Mode exchanges")

References #1076.

History

#1 - 19.08.2015 15:34 - Tobias Brunner

- File 0001-ikev1-Fix-handling-of-overlapping-Quick-Mode-exchang.patch added

- Status changed from New to Feedback

While racoon installs all the SAs and policies it assigns the same reqid to all of them. So if you look at the outbound policies on Peer2 you see that all the traffic will be sent to the same IPsec SA because the associated template lists the same IPs, same protocol, same mode, same reqid, so the first SA will get used for all the traffic. That won't work together with strongSwan on the other end. Because there the kernel will check the decrypted packets against the inbound policies and that won't result in a match for the second set of traffic selectors as the inbound policy points to a different IPsec SA (different reqid). Is it possible to use distinct reqids for each SA in racoon?

> racoon initiates the tunnel - one policy comes UP but the second policy ends stuck with the following log on StrongSwan side:

It looks like the first message of the second Quick Mode exchange overtaking the third message of the first Quick Mode exchange trips up strongSwan. The exchanged messages are as follows (with the respective message ID):

```
racoon                                        charon
Quick Mode req (3246901756) ----------->
                            <-----------   Quick Mode resp (3246901756)
Quick Mode req (3246901756) ---\
Quick Mode req (2849100021) ----\------>   Interpreted as last QM message of (3246901756)
                                 \----->   Interpreted as new QM exchange, but it misses all required payloads
```

This is definitely a bug. The attached patch tried to address this, could you give it a try? (It won't fix the policy/SA problem mentioned above, but it should allow racoon to initiate the SAs).

#2 - 19.08.2015 18:08 - Alexander Velkov

Hi Tobias,

> Is it possible to use distinct reqids for each SA in racoon?

I tried setting distinct reqids in the policies. The StrongSwan configuration was changed by setting *reqid=0* in the *ipsectest$0* and *reqid=1* in the *ipsectest$1* blocks. This change fixed the issue with traffic between both policies, great!

> This is definitely a bug. The attached patch tried to address this, could you give it a try? (It won't fix the policy/SA problem mentioned above, but it should allow racoon to initiate the SAs).

I applied your patch 0001-ikev1-Fix-handling-of-overlapping-Quick-Mode-exchang.patch:

- with reqid changes - both policies came UP and traffic flows successfully through both of the policies (irrespective of which peer initiates the tunnel).

- without reqid changes - both policies came UP but traffic flows only through the last established policy as expected (irrespective of which peer initiates the tunnel).

Both changes solved my issues.

Thank you very much for the great support!

#3 - 19.08.2015 18:30 - Tobias Brunner

- Tracker changed from Issue to Bug

- Assignee set to Tobias Brunner

- Target version set to 5.3.3

- % Done set to 0

> > Is it possible to use distinct reqids for each SA in racoon?
>
> I tried setting distinct reqids in the policies. The StrongSwan configuration was changed by setting *reqid=0* in the *ipsectest$0* and *reqid=1* in the *ipsectest$1* blocks. This change fixed the issue with traffic between both policies, great!

That's actually kind of an ugly hack. *reqid=0* will allocate 1 as reqid (0 means no static reqid and the first one allocated is 1 - but if there are other SAs that might not be the case) and *reqid=1* sets the reqid for the second SA to the same value. You should rather check if you can use distinct reqids on the racoon side (e.g. use *unique* instead of *require*).

> > This is definitely a bug. The attached patch tried to address this, could you give it a try? (It won't fix the policy/SA problem mentioned above, but it should allow racoon to initiate the SAs).
>
> I applied your patch 0001-ikev1-Fix-handling-of-overlapping-Quick-Mode-exchang.patch:
>
> - with reqid changes - both policies came UP and traffic flows successfully through both of the policies (irrespective of which peer initiates the tunnel).
> - without reqid changes - both policies came UP but traffic flows only through the last established policy as expected (irrespective of which peer initiates the tunnel).

OK, thanks for testing. I'll line that up for the next release.

#4 - 19.08.2015 18:54 - Alexander Velkov

Hi,

> That's actually kind of an ugly hack. *reqid=0* will allocate 1 as reqid (0 means no static reqid and the first one allocated is 1 - but if there are other SAs that might not be the case) and *reqid=1* sets the reqid for the second SA to the same value. You should rather check if you can use distinct reqids on the racoon side (e.g. use *unique* instead of *require*).

Good point!

So I changed back the StrongSwan configuration to not set *reqids*. Additionally, the setkey.conf configuration was changed by replacing *require* by *unique* for all lines.

Traffic flows without any errors through both policies!

Thanks once again!

#5 - 20.08.2015 19:14 - Tobias Brunner

- Status changed from Feedback to Closed

- Resolution set to Fixed

Files

0001-ikev1-Fix-handling-of-overlapping-Quick-Mode-exchang.patch	5.74 KB	19.08.2015	Tobias Brunner