strongSwan - Bug #2847

Duplicate CHILD_SA in case of poor network connectivity and make-before-break reauthentication

03.12.2018 14:58 - Alexandre Martins

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Tobias Brunner	**Estimated time:**	0.00 hour
Category:	libcharon		
Target version:	5.7.2		
Affected version:	5.6.3	**Resolution:**	Fixed

Description

Hi,

I'm currently trying to reproduce an issue when CHILD_SA are rekey with a poor internet connectivity (packets lost).

To stress the issue, I have set 180 seconds of IKE SA lifetime and 60 seconds of IPSEC SA lifetime.

I have truncated the verbose to be more readable.

- The story start with a needed rekey on the IPSEC SA. The SPI should change from c85f22aa to c3da69c0
Nov 29 15:12:28 16[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> activating CHILD_REKEY task
Nov 29 15:12:28 16[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> establishing CHILD_SA (site_to_site)(9dce1767c36666bc2cce033726356b14){7355} reqid 2
Nov 29 15:12:28 16[KNL] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> got SPI c3da69c0
Nov 29 15:12:28 16[CHD] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> CHILD_SA (site_to_site)(9dce1767c36666bc2cce033726356b14){7074} state change: INSTALLED => REKEYING
Nov 29 15:12:28 16[ENC] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> generating CREATE_CHILD_SA request 2 [N(REKEY_SA) N(ESP_TFC_PAD_N) SA No KE TSi TSr]

- However, the connection is unstable and retransmit task start:
Nov 29 15:12:33 15[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> retransmit 1 of request with message ID 2
Nov 29 15:12:40 04[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> retransmit 2 of request with message ID 2
Nov 29 15:12:55 03[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> retransmit 3 of request with message ID 2

- I suspect now the peer to not receive our message and start the rekey on this own:
Nov 29 15:12:55 03[NET] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> received packet: from XXX[500] to XXX[500] (1228 bytes)
Nov 29 15:12:55 03[ENC] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> parsed CREATE_CHILD_SA request 2 [N(REKEY_SA) N(ESP_TFC_PAD_N) SA No KE TSi TSr]
Nov 29 15:12:55 03[KNL] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> got SPI c9d05cd4
Nov 29 15:12:55 03[CHD] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> CHILD_SA (site_to_site)(9dce1767c36666bc2cce033726356b14){7406} state change: CREATED => INSTALLING
Nov 29 15:12:55 03[KNL] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> adding SAD entry with SPI c9d05cd4 and reqid {2}

Please note that the packet is not a CREATE_CHILD_SA RESPONSE, but a REQUEST. And the receive request don't have the same SPI: c3da69c0 VS c9d05cd4

- And now, here start the strange things. Charon mark the rekeying IPSEC SA as rekeyed, and detect the collision, but not delete the first SA. I think that's a bug.
Nov 29 15:12:56 03[CHD] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> CHILD_SA (site_to_site)(9dce1767c36666bc2cce033726356b14){7074} state change: REKEYING => REKEYED
Nov 29 15:12:56 03[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> detected CHILD_REKEY collision with CHILD_REKEY

- The IKE SA is now rekeyed actively (lost message ?)
Nov 29 15:13:04 07[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> initiator did not reauthenticate as requested
Nov 29 15:13:04 07[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> reauthenticating IKE_SA (site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)[1311] actively
Nov 29 15:13:04 07[MGR] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> created IKE_SA (unnamed)[1434]

At this point, I suspect the kernel to have two IPSEC SA for the same tunnel. Can the new IKE SA inherit the two CHILD SA

and rekey them again and again ?

- Then a lost response (to the 15:12:28 message) comes finaly and the rekey of the SPI c3da69c0 complete
Nov 29 15:13:04 15[NET] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> received packet: from XXX[500] to XXX[500] (1212 bytes)
Nov 29 15:13:04 15[ENC] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> parsed CREATE_CHILD_SA response 2 [N(ESP_TFC_PAD_N) SA No KE TSi TSr]
Nov 29 15:13:04 15[KNL] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> unable to delete SAD entry with SPI c3da69c0: No such file or directory (2)
Nov 29 15:13:04 15[KNL] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> adding SAD entry with SPI c3da69c0 and reqid {2}

- due to the collision, the IPSEC SA is deleted
Nov 29 15:13:04 15[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> CHILD_SA rekey collision lost, deleting redundant child (site_to_site)(9dce1767c36666bc2cce033726356b14){7355}
Nov 29 15:13:04 15[APP] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> CHILD_SA rekey collision lost, deleting redundant child c3da69c0
Nov 29 15:13:04 15[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> sending DELETE for ESP CHILD_SA with SPI c3da69c0
Nov 29 15:13:04 15[ENC] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> added payload of type DELETE to message
Nov 29 15:13:04 15[ENC] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> generating INFORMATIONAL request 3 [D]

- a rekey task is still in queue (From the duplicate ?)
Nov 29 15:13:04 15[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> reinitiating already active tasks
Nov 29 15:13:04 15[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> CHILD_REKEY task

- And the delete task never complete
Nov 29 15:13:09 08[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> retransmit 1 of request with message ID 3
Nov 29 15:13:16 13[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> retransmit 2 of request with message ID 3
Nov 29 15:13:30 01[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> retransmit 3 of request with message ID 3
Nov 29 15:13:31 08[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> queueing CHILD_REKEY task
Nov 29 15:13:31 08[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> delaying task initiation, INFORMATIONAL exchange in progress
Nov 29 15:13:31 08[MGR] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> checkin IKE_SA (site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)[1311]
Nov 29 15:13:31 08[MGR] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> checkin of IKE_SA successful
Nov 29 15:13:43 03[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> queueing CHILD_REKEY task
Nov 29 15:13:43 03[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> delaying task initiation, INFORMATIONAL exchange in progress
Nov 29 15:13:43 03[MGR] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> checkin IKE_SA (site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)[1311]
Nov 29 15:13:43 03[MGR] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> checkin of IKE_SA successful
Nov 29 15:13:44 00[APP] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> IKE SA is expired, closing it
Nov 29 15:13:44 00[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> queueing IKE_DELETE task
Nov 29 15:13:44 00[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> delaying task initiation, INFORMATIONAL exchange in progress

Associated revisions

Revision 10f8834b - 07.12.2018 10:28 - Tobias Brunner

ikev2: Don't recreate IKE_SA if deletion fails after make-before-break reauth

Fixes: 745714307256 ("During reauthentication reestablish IKE_SA even if deleting the old one fails.")
Fixes #2847.

History

#1 - 03.12.2018 15:46 - Tobias Brunner

- Status changed from New to Feedback

> - And now, here start the strange things. Charon mark the rekeying IPSEC SA as rekeyed, and detect the collision, but not delete the first SA. I think that's a bug.

The outcome of the rekey collision depends on the nonces used in the CREATE_CHILD_SA exchanges. And it will only be resolved once the active task (i.e. the rekeying that was initiated locally) is concluded, so there will be several retransmits until a response is received.

> - The IKE SA is now rekeyed actively (lost message ?)

```
Nov 29 15:13:04 07[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> initiator did not reauthen
ticate as requested
```

This means the peer did not reauthenticate at the time requested via AUTH_LIFETIME notify. So a reauthentication is initiated actively. However, that should not actually be possible until the rekeying above is done, since we don't have the complete logs it's unclear what exactly happens. See ExpiryRekey for details.

> At this point, I suspect the kernel to have two IPSEC SA for the same tunnel. Can the new IKE SA inherit the two CHILD SA and rekey them again and again ?

Yes, there will be the newly created CHILD_SA (only the inbound SA is installed, the outbound SA won't be installed until the old SA is deleted). And no, CHILD_SAs are not inherited at all for IKEv2 reauthentications (again see ExpiryRekey for details and the difference to proper rekeyings and the two different reauth methods).

> - Then a lost response (to the 15:12:28 message) comes finaly and the rekey of the SPI c3da69c0 complete

The log messages here are a bit weird. The delete is unique to *kernel-pfkey*, but if it failed, there should be another log message that's not seen here (deleting SPI allocation SA failed) and the adding SAD entry ... message is logged on level 2 but that should result in more messages when attempting to delete the SAD entry.

```
Nov 29 15:13:04 15[NET] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> received packet: from XXX[
500] to XXX[500] (1212 bytes)
Nov 29 15:13:04 15[ENC] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> parsed CREATE_CHILD_SA res
ponse 2 [ N(ESP_TFC_PAD_N) SA No KE TSi TSr ]
Nov 29 15:13:04 15[KNL] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> unable to delete SAD entry
 with SPI c3da69c0: No such file or directory (2)
Nov 29 15:13:04 15[KNL] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> adding SAD entry with SPI
c3da69c0 and reqid {2}
```

> - due to the collision, the IPSEC SA is deleted

Yes, the peer won the rekey collision, so the redundant CHILD_SA created by this host is deleted.

> - a rekey task is still in queue (From the duplicate ?)

```
Nov 29 15:13:04 15[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> reinitiating already activ
e tasks
Nov 29 15:13:04 15[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311>   CHILD_REKEY task
```

That's the still active rekeying task that now is handling the deletion of the redundant CHILD_SA (it internally uses a child-delete task). And the peer should delete the original CHILD_SA (with SPI c85f22aa, I guess).

> - And the delete task never complete

Yeah, that's obviously a problem with low rekeying settings (i.e. if the soft limit is lower than a full retransmission cycle).

#2 - 03.12.2018 16:48 - Alexandre Martins

Tobias Brunner wrote:

> > - And now, here start the strange things. Charon mark the rekeying IPSEC SA as rekeyed, and detect the collision, but not delete the first SA. I think that's a bug.
>
> The outcome of the rekey collision depends on the nonces used in the CREATE_CHILD_SA exchanges. And it will only be resolved once the active task (i.e. the rekeying that was initiated locally) is concluded, so there will be several retransmits until a response is received.

Yes, there is three retransmissions:
Nov 29 15:12:33 15[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> retransmit 1 of request with message ID 2

Nov 29 15:12:40 04[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> retransmit 2 of request with message ID 2
Nov 29 15:12:55 03[IKE] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> retransmit 3 of request with message ID 2

And the received request prevent the deletion of this CHILD SA

> - The IKE SA is now rekeyed actively (lost message ?)
> [...]

This means the peer did not reauthenticate at the time requested via AUTH_LIFETIME notify. So a reauthentication is initiated actively. However, that should not actually be possible until the rekeying above is done, since we don't have the complete logs it's unclear what exactly happens. See ExpiryRekey for details.

But there is no more rekeying in progress as charon show us :
Nov 29 15:12:56 03[CHD] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> CHILD_SA (site_to_site)(9dce1767c36666bc2cce033726356b14){7074} state change: REKEYING => REKEYED

The new CHILD_SA mark the old one REKEYED. Is it maybe the problem ?

> At this point, I suspect the kernel to have two IPSEC SA for the same tunnel. Can the new IKE SA inherit the two CHILD SA and rekey them again and again ?

Yes, there will be the newly created CHILD_SA (only the inbound SA is installed, the outbound SA won't be installed until the old SA is deleted). And no, CHILD_SAs are not inherited at all for IKEv2 reauthentications (again see ExpiryRekey for details and the difference to proper rekeyings and the two different reauth methods).

OK, got it.

> - Then a lost response (to the 15:12:28 message) comes finaly and the rekey of the SPI c3da69c0 complete

The log messages here are a bit weird. The delete is unique to *kernel-pfkey*, but if it failed, there should be another log message that's not seen here (deleting SPI allocation SA failed) and the adding SAD entry ... message is logged on level 2 but that should result in more messages when attempting to delete the SAD entry.

I have the creation one, but no deletion at all (some other CHILD_SA are deleted with "deleted SAD entry with SPI " message):
Nov 29 15:13:04 15[KNL] <(site_to_site)(d58c0730fd881cc82ba47eb1eaa19e99)|1311> adding SAD entry with SPI c3da69c0 and reqid {2}

> [...]
>
> - due to the collision, the IPSEC SA is deleted

Yes, the peer won the rekey collision, so the redundant CHILD_SA created by this host is deleted.

OK

> - a rekey task is still in queue (From the duplicate ?)
> [...]

That's the still active rekeying task that now is handling the deletion of the redundant CHILD_SA (it internally uses a child-delete task). And the peer should delete the original CHILD_SA (with SPI c85f22aa, I guess).

> - And the delete task never complete

Yeah, that's obviously a problem with low rekeying settings (i.e. if the soft limit is lower than a full retransmission cycle).

I have the same problem with bigger lifetime (12 hours IKE, 1 hour IPSEC typically). But the duplication take longer to reproduce, that why i have reduced the lifetime.
I'll augment the values to 15 minutes IKE and 5 minutes IPSEC.

#3 - 04.12.2018 10:04 - Tobias Brunner

> And the received request prevent the deletion of this CHILD SA

What do you mean?

> The new CHILD_SA mark the old one REKEYED. Is it maybe the problem ?

No, that one is rekeyed, so that's fine.

> I have the creation one, but no deletion at all (some other CHILD_SA are deleted with "deleted SAD entry with SPI " message):

Not sure what that means. Do you have any patches applied to your installation (in particular that plugin)?

> I have the same problem with bigger lifetime (12 hours IKE, 1 hour IPSEC typically). But the duplication take longer to reproduce, that why i have reduced the lifetime.

What problem are you referring to exactly? I don't see a problem in the output above (except for the retransmissions, which delay stuff and should just trigger DPD actions in the worst case). Unless the other end does weird stuff, this is normal rekey collision handling.

#4 - 05.12.2018 10:54 - Alexandre Martins

Sorry for the late response, your questions made me rethink the problem. I was focused on a duplicate of CHILD_SA, but the problem is a duplication of IKE_SA, resulting the duplication of CHILD_SA.

I keep the lifetimes very short to produce the problem quickly and have the smallest verbose file possible.
lifetimes -> IKE: 300 IPSEC: 60

When the connection is unstable, it seems that we are unable to send "big" packets (over 500 bytes ?)

Here the new log that I can see.

- NEW IKE_SA(3) due to reauthentication. This IKE_SA is normal, I just put it to give you the creation time.
  Dec 5 08:08:42 16[IKE] <(site_A_to_B)|2> reauthenticating IKE_SA (site_A_to_B)[2]
  Dec 5 08:08:42 16[MGR] <(site_A_to_B)|2> created IKE_SA (unnamed)[3]

- The IKE_SA is going to expire, a re-auth is started
  Dec 5 08:10:47 15[IKE] <(site_A_to_B)|3> reauthenticating IKE_SA (site_A_to_B)[3]
  Dec 5 08:10:47 15[MGR] <(site_A_to_B)|3> created IKE_SA (unnamed)[4]
  Dec 5 08:10:47 15[ENC] <(site_A_to_B)|3> generating IKE_SA_INIT request 0 [SA KE No N(NATD_S_IP) N(NATD_D_IP) N(HASH_ALG) N(REDIR_SUP)]
  Dec 5 08:10:47 15[NET] <(site_A_to_B)|3> sending packet (412 bytes)

- The packet is received then replied, so the auth continue
  Dec 5 08:10:47 04[NET] <(site_A_to_B)|4> received packet
  Dec 5 08:10:47 04[ENC] <(site_A_to_B)|4> parsed IKE_SA_INIT response 0 [SA KE No N(NATD_S_IP) N(NATD_D_IP) N(HASH_ALG)]
  Dec 5 08:10:47 04[ENC] <(site_A_to_B)|4> generating IKE_AUTH request 1 [IDi IDr AUTH N(ESP_TFC_PAD_N) SA TSi TSr N(EAP_ONLY) N(MSG_ID_SYN_SUP)]
  Dec 5 08:10:47 04[NET] <(site_A_to_B)|4> sending packet (972 bytes)

- Now the connection seems to be unstable, the retransmit is starting
  Dec 5 08:10:51 04[IKE] <(site_A_to_B)|4> retransmit 1 of request with message ID 1
  Dec 5 08:10:58 04[IKE] <(site_A_to_B)|4> retransmit 2 of request with message ID 1

- A CHILD_SA rekey start now
  Dec 5 08:11:03 04[NET] <(site_A_to_B)|3> received packet (1228 bytes)
  Dec 5 08:11:03 04[ENC] <(site_A_to_B)|3> parsed CREATE_CHILD_SA request 4 [N(REKEY_SA) N(ESP_TFC_PAD_N) SA No KE TSi TSr]
  Dec 5 08:11:03 04[CHD] <(site_A_to_B)|3> CHILD_SA (site_A_to_B){13} state change: CREATED => INSTALLING
  Dec 5 08:11:03 04[CHD] <(site_A_to_B)|3> adding inbound ESP SA
  Dec 5 08:11:03 04[CHD] <(site_A_to_B)|3> CHILD_SA (site_A_to_B){13} state change: INSTALLING => INSTALLED
  Dec 5 08:11:03 04[CHD] <(site_A_to_B)|3> CHILD_SA (site_A_to_B){11} state change: INSTALLED => REKEYING
  Dec 5 08:11:03 04[CHD] <(site_A_to_B)|3> CHILD_SA (site_A_to_B){11} state change: REKEYING => REKEYED
  Dec 5 08:11:03 04[ENC] <(site_A_to_B)|3> generating CREATE_CHILD_SA response 4 [N(ESP_TFC_PAD_N) SA No KE TSi TSr]
  Dec 5 08:11:03 04[NET] <(site_A_to_B)|3> sending packet (1212 bytes)

- The connection is still unstable, a retransmission of the CREATE_CHILD_SA is received
  Dec 5 08:11:07 04[ENC] <(site_A_to_B)|3> parsed CREATE_CHILD_SA request 4 [N(REKEY_SA) N(ESP_TFC_PAD_N) SA No KE TSi TSr]
  Dec 5 08:11:07 04[IKE] <(site_A_to_B)|3> received retransmit of request with ID 4, retransmitting response

- Same thing with the IKE_SA
  Dec 5 08:11:11 04[IKE] <(site_A_to_B)|4> retransmit 3 of request with message ID 1

- And again for the CREATE_CHILD_SA, two times
  Dec 5 08:11:14 04[ENC] <(site_A_to_B)|3> parsed CREATE_CHILD_SA request 4 [N(REKEY_SA) N(ESP_TFC_PAD_N) SA No KE TSi TSr]
  Dec 5 08:11:14 04[IKE] <(site_A_to_B)|3> received retransmit of request with ID 4, retransmitting response
  Dec 5 08:11:27 04[ENC] <(site_A_to_B)|3> parsed CREATE_CHILD_SA request 4 [N(REKEY_SA) N(ESP_TFC_PAD_N) SA No KE TSi TSr]
  Dec 5 08:11:27 04[IKE] <(site_A_to_B)|3> received retransmit of request with ID 4, retransmitting response

- The IKE_SA is now giving up, so it restart. The connection is stable for few seconds
  Dec 5 08:11:34 04[IKE] <(site_A_to_B)|4> giving up after 3 retransmits
  Dec 5 08:11:34 04[APP] <(site_A_to_B)|4> Remote seems to be dead
  Dec 5 08:11:34 04[IKE] <(site_A_to_B)|4> peer not responding, trying again (2/3)
  Dec 5 08:11:34 04[MGR] <(site_A_to_B)|4> change initiator SPI of IKE_SA (site_A_to_B)[4] from 924847b8cdb2982a to 92e4137eaad78b94
  Dec 5 08:11:34 04[ENC] <(site_A_to_B)|4> generating IKE_SA_INIT request 0 [SA KE No N(NATD_S_IP) N(NATD_D_IP) N(HASH_ALG) N(REDIR_SUP)]
  Dec 5 08:11:34 04[NET] <(site_A_to_B)|4> sending packet (412 bytes)
  Dec 5 08:11:34 04[NET] <(site_A_to_B)|4> received packet (320 bytes)
  Dec 5 08:11:34 04[ENC] <(site_A_to_B)|4> parsed IKE_SA_INIT response 0 [SA KE No N(NATD_S_IP) N(NATD_D_IP) N(HASH_ALG)]
  Dec 5 08:11:34 04[ENC] <(site_A_to_B)|4> generating IKE_AUTH request 1 [IDi IDr AUTH N(ESP_TFC_PAD_N) SA TSi TSr N(EAP_ONLY) N(MSG_ID_SYN_SUP)]
  Dec 5 08:11:34 04[NET] <(site_A_to_B)|4> sending packet (972 bytes)
  Dec 5 08:11:34 04[NET] <(site_A_to_B)|4> received packet (956 bytes)
  Dec 5 08:11:34 04[ENC] <(site_A_to_B)|4> parsed IKE_AUTH response 1 [IDr AUTH N(ESP_TFC_PAD_N) SA TSi TSr N(AUTH_LFT)]
  Dec 5 08:11:34 04[IKE] <(site_A_to_B)|4> IKE_SA (site_A_to_B)[4] state change: CONNECTING => ESTABLISHED
  Dec 5 08:11:34 04[IKE] <(site_A_to_B)|4> activating IKE_REAUTH_COMPLETE task

- The deletion of the previous IKE_SA start, but it hang due to the CREATE_CHILD_SA that does not complete
  Dec 5 08:11:34 09[IKE] <(site_A_to_B)|3> IKE_SA (site_A_to_B)[3] state change: ESTABLISHED => DELETING
  Dec 5 08:11:34 09[IKE] <(site_A_to_B)|3> sending DELETE for IKE_SA (site_A_to_B)[3]
  Dec 5 08:11:34 09[ENC] <(site_A_to_B)|3> generating INFORMATIONAL request 2 [D]
  Dec 5 08:11:34 09[NET] <(site_A_to_B)|3> sending packet (76 bytes)

- So retransmission start, not because the network, but because the CREATE_CHILD_SA message
  Dec 5 08:11:38 09[IKE] <(site_A_to_B)|3> retransmit 1 of request with message ID 2
  Dec 5 08:11:46 09[IKE] <(site_A_to_B)|3> retransmit 2 of request with message ID 2
  Dec 5 08:11:59 04[IKE] <(site_A_to_B)|3> retransmit 3 of request with message ID 2

- And finally, it give up and start the creation of another IKE_SA !
  Dec 5 08:12:22 04[IKE] <(site_A_to_B)|3> giving up after 3 retransmits
  Dec 5 08:12:22 04[APP] <(site_A_to_B)|3> Remote seems to be dead
  Dec 5 08:12:22 04[IKE] <(site_A_to_B)|3> proper IKE_SA delete failed, peer not responding
  Dec 5 08:12:22 04[IKE] <(site_A_to_B)|3> delete during reauthentication failed, trying to reestablish IKE_SA anyway
  Dec 5 08:12:22 04[MGR] <(site_A_to_B)|3> created IKE_SA (unnamed)[6]
  Dec 5 08:12:22 04[IKE] <(site_A_to_B)|3> restarting CHILD_SA (site_A_to_B)
  Dec 5 08:12:22 04[CHD] <(site_A_to_B)|3> CHILD_SA (site_A_to_B){13} state change: INSTALLED => DESTROYING

At this point, we have split in the IKE_SA [3] two IKE_SA: the [4] and the [6].

Hope it's more clear this time.

#5 - 05.12.2018 12:31 - Tobias Brunner

> When the connection is unstable, it seems that we are unable to send "big" packets (over 500 bytes ?)

If the peer supports IKEv2 fragmentation, use that and maybe reduce the fragment size. You should definitely try to avoid these problems somehow. Increasing the retransmission settings might also be an option.

Regarding reauthentication in general, let me refer you to ExpiryRekey once more. If you want to avoid it, disable it and use rekeying instead (*reauth=no* in ipsec.conf, swanctl.conf doesn't enable reauthentication, by default).

> The deletion of the previous IKE_SA start, but it hang due to the CREATE_CHILD_SA that does not complete

What do you mean? It clearly sends a DELETE payload. And it is in state IKE_DELETING (seen by the log messages later).

> And finally, it give up and start the creation of another IKE_SA !

Yeah, that should probably be avoided for make-before-break reauthentication as the new SA is established already in that case.

> Dec 5 08:12:22 04[IKE] <(site_A_to_B)|3> proper IKE_SA delete failed, peer not responding
> Dec 5 08:12:22 04[IKE] <(site_A_to_B)|3> delete during reauthentication failed, trying to reestablish IKE_SA anyway

These are logged if an SA marked for reauthentication could not be deleted, which comes first in the break-before-make reauthentication, so recreating the SA makes sense then, but not if make-before-break reauthentication is used. I pushed a fix for that to the *2847-reauth-reestablish* branch.

#6 - 05.12.2018 13:57 - Alexandre Martins

Tobias Brunner wrote:

> If the peer supports IKEv2 fragmentation, use that and maybe reduce the fragment size. You should definitely try to avoid these problems somehow. Increasing the retransmission settings might also be an option.
>
> Regarding reauthentication in general, let me refer you to ExpiryRekey once more. If you want to avoid it, disable it and use rekeying instead (*reauth=no* in ipsec.conf, swanctl.conf doesn't enable reauthentication, by default).

OK, I'll check that with our customer.

> What do you mean? It clearly sends a DELETE payload. And it is in state IKE_DELETING (seen by the log messages later).

I have seen that in another log, I think it's related, but I'm not sure.
queueing CHILD_DELETE task
delaying task initiation, CREATE_CHILD_SA exchange in progress

> Yeah, that should probably be avoided for make-before-break reauthentication as the new SA is established already in that case.

Nice ! This bug start to make me crazy :D

> > Dec 5 08:12:22 04[IKE] <(site_A_to_B)|3> proper IKE_SA delete failed, peer not responding
> > Dec 5 08:12:22 04[IKE] <(site_A_to_B)|3> delete during reauthentication failed, trying to reestablish IKE_SA anyway
>
> These are logged if an SA marked for reauthentication could not be deleted, which comes first in the break-before-make reauthentication, so recreating the SA makes sense then, but not if make-before-break reauthentication is used. I pushed a fix for that to the *2847-reauth-reestablish* branch.

Thank you ! I'll test that now !

#7 - 05.12.2018 16:48 - Alexandre Martins

The test is running since 1 hour now. No duplicate found. \o/

I let the test run all the night. I'll confirm the fix tomorrow

#8 - 06.12.2018 09:09 - Alexandre Martins

After one night, I confirm that the problem is gone.

Thank you very much !

#9 - 06.12.2018 11:06 - Tobias Brunner

- Tracker changed from Issue to Bug

- Subject changed from Duplicate CHILD_SA in cas of poor network connectivity to Duplicate CHILD_SA in case of poor network connectivity and make-before-break reauthentication

- Category set to libcharon

- Assignee set to Tobias Brunner

- Target version set to 5.7.2

Great, I'll line up the fix for the next release.

#10 - 07.12.2018 10:42 - Tobias Brunner

- Status changed from Feedback to Closed

- Resolution set to Fixed