strongSwan - Bug #2837

IKE_SA is inadvertently reset when a delayed COOKIE payload is received with the same value already in use

28.11.2018 10:00 - Alexandre Martins

Status:	Closed	**Start date:**	
Priority:	Low	**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,

Our compagny is trying to figure out some troubles with strongswan for a customer, who have an unstable internet connection.

During my investigations into charon's verbose file, I can see the following behavior (here followow the truncated log to be more readable):

```
- The IKE_SA comes to timeout, and charon try to restart it:
Nov 21 11:33:29 13[IKE] <(site_to_site)(A_to_B)|1315> received DELETE for IKE_SA (site_to_site)(A_
to_B)[1315]
Nov 21 11:33:29 13[IKE] <(site_to_site)(A_to_B)|1315> IKE_SA (site_to_site)(A_to_B)[1315] state ch
ange: ESTABLISHED => DELETING
Nov 21 11:33:29 13[MGR] <(site_to_site)(A_to_B)|1315> created IKE_SA (unnamed)[1859]
Nov 21 11:33:29 13[IKE] <(site_to_site)(A_to_B)|1315> restarting CHILD_SA (site_to_site)(old_A_to_
B)
Nov 21 11:33:29 13[IKE] <(site_to_site)(A_to_B)|1315> IKE_SA (site_to_site)(A_to_B)[1859] state ch
ange: CREATED => CONNECTING

- The IKE packets is lost 2 times, but was retransmit (OK)
Nov 21 11:33:33 12[IKE] <(site_to_site)(A_to_B)|1859> retransmit 1 of request with message ID 0
Nov 21 11:33:40 14[IKE] <(site_to_site)(A_to_B)|1859> retransmit 2 of request with message ID 0

- The third retransmission get replied, and the process continue
Nov 21 11:33:53 13[IKE] <(site_to_site)(A_to_B)|1859> retransmit 3 of request with message ID 0
Nov 21 11:33:53 13[ENC] <(site_to_site)(A_to_B)|1859> parsed IKE_SA_INIT response 0 [ N(COOKIE) ]
Nov 21 11:33:53 13[IKE] <(site_to_site)(A_to_B)|1859> IKE_SA (site_to_site)(A_to_B)[1859] state ch
ange: CONNECTING => CREATED
Nov 21 11:33:53 13[IKE] <(site_to_site)(A_to_B)|1859> IKE_SA (site_to_site)(A_to_B)[1859] state ch
ange: CREATED => CONNECTING
Nov 21 11:33:53 13[ENC] <(site_to_site)(A_to_B)|1859> generating IKE_SA_INIT request 0 [ N(COOKIE)
 SA KE No N(NATD_S_IP) N(NATD_D_IP) N(HASH_ALG) N(REDIR_SUP) ]

- However, the connection still unstable and retransmission start again
Nov 21 11:33:57 13[IKE] <(site_to_site)(A_to_B)|1859> retransmit 1 of request with message ID 0
Nov 21 11:34:05 01[IKE] <(site_to_site)(A_to_B)|1859> retransmit 2 of request with message ID 0

- And we receive a retransmission of the IKE_SA_INIT response. This retransmission restart the sta
te machine but don't stop the retransmissions
Nov 21 11:34:06 08[ENC] <(site_to_site)(A_to_B)|1859> parsed IKE_SA_INIT response 0 [ N(COOKIE) ]
Nov 21 11:34:06 08[IKE] <(site_to_site)(A_to_B)|1859> IKE_SA (site_to_site)(A_to_B)[1859] state ch
ange: CONNECTING => CREATED
Nov 21 11:34:06 08[IKE] <(site_to_site)(A_to_B)|1859> IKE_SA (site_to_site)(A_to_B)[1859] state ch
ange: CREATED => CONNECTING
Nov 21 11:34:06 08[ENC] <(site_to_site)(A_to_B)|1859> generating IKE_SA_INIT request 0 [ N(COOKIE)
 SA KE No N(NATD_S_IP) N(NATD_D_IP) N(HASH_ALG) N(REDIR_SUP) ]

- The connection is unstable again
Nov 21 11:34:10 12[IKE] <(site_to_site)(A_to_B)|1859> retransmit 1 of request with message ID 0
Nov 21 11:34:17 13[IKE] <(site_to_site)(A_to_B)|1859> retransmit 2 of request with message ID 0
```

```
- Now, I can see the retransmission of the first attempt (the timing of retransmissions is 4,7,13
seconds)
Nov 21 11:34:18 04[IKE] <(site_to_site)(A_to_B)|1859> retransmit 3 of request with message ID 0
Nov 21 11:34:18 04[IKE] <(site_to_site)(A_to_B)|1859> giving up after 3 retransmits
Nov 21 11:34:18 04[APP] <(site_to_site)(A_to_B)|1859> Remote seems to be dead
Nov 21 11:34:18 04[MGR] <(site_to_site)(A_to_B)|1859> change initiator SPI of IKE_SA (site_to_site
)(A_to_B)[1859] from 9f0583f59b60932e to a1fd137d2154c533
Nov 21 11:34:18 04[IKE] <(site_to_site)(A_to_B)|1859> IKE_SA (site_to_site)(A_to_B)[1859] state ch
ange: CONNECTING => CREATED
Nov 21 11:34:18 04[IKE] <(site_to_site)(A_to_B)|1859> IKE_SA (site_to_site)(A_to_B)[1859] state ch
ange: CREATED => CONNECTING
Nov 21 11:34:18 04[ENC] <(site_to_site)(A_to_B)|1859> generating IKE_SA_INIT request 0 [ N(COOKIE)
 SA KE No N(NATD_S_IP) N(NATD_D_IP) N(HASH_ALG) N(REDIR_SUP) ]
```

Should we reject the retransmission of the response, like we ignore it in the request part of process_message function of task_manager_v2.c ?

```
/* reject initial messages if not received in specific states */
type = msg->get_exchange_type(msg);
state = this->ike_sa->get_state(this->ike_sa);
if ((type == IKE_SA_INIT && state != IKE_CREATED)
(type == IKE_AUTH && state != IKE_CONNECTING))  {
DBG1(DBG_IKE, "ignoring %N in IKE_SA state %N",
exchange_type_names, type, ike_sa_state_names, state);
return FAILED;
}
```

Or maybe reset the retransmission counter ?

Best regards,

Alexandre

Associated revisions

Revision 01f462f0 - 07.12.2018 10:25 - Tobias Brunner

ikev2: Ignore COOKIE notifies we already received

This could be due to a delayed response to an IKE_SA_INIT retransmit.

Fixes #2837.

History

#1 - 28.11.2018 12:18 - Tobias Brunner

- Tracker changed from Bug to Issue

- Category changed from charon to libcharon

- Status changed from New to Feedback

- Start date deleted (28.11.2018)

What exactly is the problem?

#2 - 28.11.2018 14:19 - Alexandre Martins

Tobias Brunner wrote:

> What exactly is the problem?

It's not a real issue for that example (that's why I put it as low priority), but it show us that a transmitted IKE_SA_INIT message can reset the state of the handshake.

I do known how charon works, but if the state is "ESTABLISHED", can a such message reset the status of the IKE_SA ?

#3 - 28.11.2018 14:22 - Tobias Brunner

> but it show us that a transmitted IKE_SA_INIT message can reset the state of the handshake.

What are you referring to?

> I do known how charon works, but if the state is "ESTABLISHED", can a such message reset the status of the IKE_SA ?

What message would that be?

#4 - 28.11.2018 15:33 - Alexandre Martins

Tobias Brunner wrote:

> > but it show us that a transmitted IKE_SA_INIT message can reset the state of the handshake.
>
> What are you referring to?

To the following line:

- Nov 21 11:34:06 08[ENC] <(site_to_site)(A_to_B)|1859> parsed IKE_SA_INIT response 0 [N(COOKIE)]
- Nov 21 11:34:06 08[IKE] <(site_to_site)(A_to_B)|1859> IKE_SA (site_to_site)(A_to_B)[1859] state change: CONNECTING => CREATED

We receive the retransmission of the IKE_SA_INIT so we reset the state to CREATED

> > I do known how charon works, but if the state is "ESTABLISHED", can a such message reset the status of the IKE_SA ?
>
> What message would that be?

Our customer have a very poor internet connection. Sometimes, we can see that some packet may take around 10 seconds to be received.
My suspicious is:
- The IKE_SA_INIT is send (but take time to arrive)
- A retransmission of the IKE_SA_INIT is receive and the handshake complete CREATED => CONNECTING => ESTABLISHED
- The first IKE_SA_INIT finaly comes and ...

And the question is, in that case, can the IKE_SA go from ESTABLISHED => CREATED ?

Our customer suffer of never deleted incoming CHILD_SA, resulting hundered of incoming IPSEC SA installed into the kernel for the same tunnel.
We already found, into the charon verbose, that the IPSEC SA is still referenced.
And we are able to reproduce the problem by randomly dropping some IKE packets.
I currently suspect "something" into the rekey mecanism that goes wrong when there is such lost.
To confirm that, I put the token charon.delete_rekeyed_delay to "0", and there is no more dead IPSEC SA into the kernel.

(And sorry for my poor English)

#5 - 28.11.2018 16:27 - Tobias Brunner

- Tracker changed from Issue to Bug

- Target version set to 5.7.2

> - Nov 21 11:34:06 08[ENC] <(site_to_site)(A_to_B)|1859> parsed IKE_SA_INIT response 0 [N(COOKIE)]
> - Nov 21 11:34:06 08[IKE] <(site_to_site)(A_to_B)|1859> IKE_SA (site_to_site)(A_to_B)[1859] state change: CONNECTING => CREATED
>
> We receive the retransmission of the IKE_SA_INIT so we reset the state to CREATED

I see, we could probably just ignore such a COOKIE notify if it contains the same value we already are using (I pushed a patch to that effect to the *2837-ignore-cookie* branch). But this doesn't really reset much because there was not much state created yet.

> My suspicious is:
> - The IKE_SA_INIT is send (but take time to arrive)
> - A retransmission of the IKE_SA_INIT is receive and the handshake complete CREATED => CONNECTING => ESTABLISHED
> - The first IKE_SA_INIT finaly comes and ...
>
> And the question is, in that case, can the IKE_SA go from ESTABLISHED => CREATED ?

The IKE_SA state won't change to ESTABLISHED until the last IKE_AUTH response has been received. But as soon as the IKE_AUTH request has

been sent, any response to the IKE_SA_INIT request with MID 0 will be ignored (actually because the responder SPI is probably different, the IKE_SA_INIT might not even result in the check out of that IKE_SA).

> We already found, into the charon verbose, that the IPSEC SA is still referenced.

Referenced by whom?

> I currently suspect "something" into the rekey mecanism that goes wrong when there is such lost.
> To confirm that, I put the token charon.delete_rekeyed_delay to "0", and there is no more dead IPSEC SA into the kernel.

Hm, that doesn't really make sense. Not sure what you mean with "dead", but the point of that option is to keep rekeyed inbound CHILD_SAs in the kernel a while (5 seconds by default, the outbound SA is removed immediately) to process delayed messages. So if that option is not 0, instead of uninstalling the SA immediately, it causes a job to get queued that deletes the CHILD_SA with a delay. And while that job requires that the CHILD_SA is still registered with the IKE_SA, if the IKE_SA was actually destroyed earlier (e.g. due to retransmits), all associated CHILD_SAs would get destroyed and uninstalled with it (if that does not happen, check the log for details). The job would then later just not find the CHILD_SA anymore and do nothing.

#6 - 28.11.2018 17:24 - Alexandre Martins

Tobias Brunner wrote:

> I see, we could probably just ignore such a COOKIE notify if it contains the same value we already are using (I pushed a patch to that effect to the *2837-ignore-cookie* branch). But this doesn't really reset much because there was not much state created yet.

OK, as expected, it's not a big issue. The peer is just declared dead too early.
I'll test that, thank you.

> The IKE_SA state won't change to ESTABLISHED until the last IKE_AUTH response has been received. But as soon as the IKE_AUTH request has been sent, any response to the IKE_SA_INIT request with MID 0 will be ignored (actually because the responder SPI is probably different, the IKE_SA_INIT might not even result in the check out of that IKE_SA).

Good, I can strike out that possibility.

> Referenced by whom?

By charon when it update the CHILD_SA byte counter.

> Hm, that doesn't really make sense. Not sure what you mean with "dead", but the point of that option is to keep rekeyed inbound CHILD_SAs in the kernel a while (5 seconds by default, the outbound SA is removed immediately) to process delayed messages. So if that option is not 0, instead of uninstalling the SA immediately, it causes a job to get queued that deletes the CHILD_SA with a delay. And while that job requires that the CHILD_SA is still registered with the IKE_SA, if the IKE_SA was actually destroyed earlier (e.g. due to retransmits), all associated CHILD_SAs would get destroyed and uninstalled with it (if that does not happen, check the log for details). The job would then later just not find the CHILD_SA anymore and do nothing.

When I say "dead", I mean "no more used for a while".

I'm currently searching the "thing" that make that job don't work (not started or ineffective). Some of the inbound CHILD_SA remain in kernel for hours but the config say 30 minutes max.
It's related to IKE messages lost, and I'm unable for the moment to tell what is the relation.

#7 - 28.11.2018 18:19 - Tobias Brunner

> The peer is just declared dead too early.

Not really dead, the COOKIE notify just triggered a reset of the IKE_SA (usually when a COOKIE notify is received, a restart is required in order to send that COOKIE in the request, but that obviously isn't really necessary when we already are sending that COOKIE, so fixing that definitely makes sense).

> > Referenced by whom?
>
> By charon when it update the CHILD_SA byte counter.

Hm, but that would mean it is still associated with an active IKE_SA. But which byte counter do you mean exactly? Via RADIUS accounting?

> When I say "dead", I mean "no more used for a while".

They should only be around for 5 seconds (in the default configuration) so not sure what might go wrong here.

> I'm currently searching the "thing" that make that job don't work (not started or ineffective). Some of the inbound CHILD_SA remain in kernel for hours but the config say 30 minutes max.

Yeah, that sounds weird. Because at some point they should expire anyway (depending on the lifetime settings you configured).

> It's related to IKE messages lost, and I'm unable for the moment to tell what is the relation.

Let me know if you find an easy way to reproduce it.

#8 - 29.11.2018 10:59 - Alexandre Martins

Tobias Brunner wrote:

> Not really dead, the COOKIE notify just triggered a reset of the IKE_SA (usually when a COOKIE notify is received, a restart is required in order to send that COOKIE in the request, but that obviously isn't really necessary when we already are sending that COOKIE, so fixing that definitely makes sense).

I agree. I'll do the test of the "2837-ignore-cookie" branch today. I'll post the result thereafter.

> Hm, but that would mean it is still associated with an active IKE_SA. But which byte counter do you mean exactly? Via RADIUS accounting?

For the "lifebytes" setting.

> They should only be around for 5 seconds (in the default configuration) so not sure what might go wrong here.
>
> > I'm currently searching the "thing" that make that job don't work (not started or ineffective). Some of the inbound CHILD_SA remain in kernel for hours but the config say 30 minutes max.
>
> Yeah, that sounds weird. Because at some point they should expire anyway (depending on the lifetime settings you configured).
>
> > It's related to IKE messages lost, and I'm unable for the moment to tell what is the relation.
>
> Let me know if you find an easy way to reproduce it.

OK, I'll open another issue when I get a clear scenario of the problem.

#9 - 29.11.2018 11:24 - Tobias Brunner

> Hm, but that would mean it is still associated with an active IKE_SA. But which byte counter do you mean exactly? Via RADIUS accounting?
>
> For the "lifebytes" setting.

strongSwan does not use that setting itself, these are configured on the IPsec SAs in the kernel, which is responsible for acting upon it (e.g. triggering a soft expire message so the IKE daemon can initiate a rekeying, or enforcing the hard limit and deleting the SA eventually).

#10 - 29.11.2018 11:57 - Alexandre Martins

Tobias Brunner wrote:

> > Hm, but that would mean it is still associated with an active IKE_SA. But which byte counter do you mean exactly? Via RADIUS accounting?
> >
> > For the "lifebytes" setting.
>
> strongSwan does not use that setting itself, these are configured on the IPsec SAs in the kernel, which is responsible for acting upon it (e.g. triggering a soft expire message so the IKE daemon can initiate a rekeying, or enforcing the hard limit and deleting the SA eventually).

In the code, it's the "get_usestats" function (via sub calls) that trigger some verbose about the "dead" child_sa. I don't know the reason why it's called.

#11 - 29.11.2018 12:02 - Tobias Brunner

> In the code, it's the "get_usestats" function (via sub calls) that trigger some verbose about the "dead" child_sa. I don't know the reason why it's called.

That method is used to determine if a DPD (inbound)/keep alive (outbound) should be sent. Anyway, it indicates that these SAs are still associated with an active IKE_SA.

#12 - 29.11.2018 16:15 - Alexandre Martins

For the retransmitted IKE_SA_INIT message, I confirm that your fix works properly.

For the SAs, I confirm that they are keep rekeyed despite the fact it should be deleted.

You can close this ticket, I'll open a new one when I'll have more details about that trouble.

Thank you for your help.

#13 - 29.11.2018 16:41 - Tobias Brunner

> For the retransmitted IKE_SA_INIT message, I confirm that your fix works properly.

OK, the fix will be included in the next release.

> For the SAs, I confirm that they are keep rekeyed despite the fact it should be deleted.

That's strange. Either the queued job should destroy them after a while, or the kernel should expire them. However, the latter might not be the case if you didn't set a time based limit, i.e. only one for packets/bytes, is that the case in your scenario?

#14 - 03.12.2018 14:59 - Alexandre Martins

Tobias Brunner wrote:

> > For the retransmitted IKE_SA_INIT message, I confirm that your fix works properly.
>
> OK, the fix will be included in the next release.
>
> > For the SAs, I confirm that they are keep rekeyed despite the fact it should be deleted.
>
> That's strange. Either the queued job should destroy them after a while, or the kernel should expire them. However, the latter might not be the case if you didn't set a time based limit, i.e. only one for packets/bytes, is that the case in your scenario?

I think that I found the cause of duplicate. Can you check the issue #2847 ?

#15 - 07.12.2018 10:41 - Tobias Brunner

- Subject changed from retransmit task is not stopped when a packet is received to IKE_SA is inadvertently reset when a delayed COOKIE payload is received with the same value already in use

- Status changed from Feedback to Closed

- Assignee set to Tobias Brunner

- Resolution set to Fixed