strongSwan - Bug #3329

Return lifetimes of the actually selected IKEv1 transform

04.02.2020 03:13 - fbh dev

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Tobias Brunner	**Estimated time:**	0.00 hour
Category:	ikev1		
Target version:	5.8.3		
Affected version:	5.8.2	**Resolution:**	Fixed

Description

We use strongswan as a VPN solution for our customers. Customers using IKEv1 with particular device configurations are unable to bring their tunnels up.

When Cisco sends first exchange, it has the following structure:

Proposal 1

Transform 1

encr alg 1
integ alg 1
dh 1
lifetime 1

Transform 2

encr alg 2
integ alg 2
dh 2
lifetime 2

The issue is that strongswan is accepting the algorithm options for Transform 2, but returning lifetime from Transform 1. In the case where lifetime 1 does not match lifetime 2, the tunnel will not come up.

Cisco logs:

```
*Jan 01 00:00:00.000: ISAKMP-ERROR: (0):Lifetime duration offered does not match policy!
*Jan 01 00:00:00.000: ISAKMP-ERROR: (0):atts are not acceptable. Next payload is 0
*Jan 01 00:00:00.000: ISAKMP-ERROR: (0):no offers accepted!
*Jan 01 00:00:00.000: ISAKMP-ERROR: (0):phase 1 SA policy not acceptable!
```

Per RFC2409, https://tools.ietf.org/html/rfc2409#section-7.1

"The responder replies in kind but selects, and returns, one transform proposal (the ISAKMP SA attributes)."

Is there any chance we can get a patch for this issue? We are unable to force customers to use IKEv2 or updating their configurations such that lifetime 1 and lifetime *n* are always equivalent.

Relevant code:

https://github.com/strongswan/strongswan/blob/f78dfb7e28935a7f07802a70a0e93d3d17264840/src/libcharon/sa/ikev1/tasks/main_mode.c#L398-L409

Associated revisions

Revision 14a77995 - 06.03.2020 10:47 - Tobias Brunner

Merge branch 'ikev1-transform-nr'

With these changes we return the lifetimes of the actually selected
transform back to the client, which is an issue if the peer uses
different lifetimes for different proposals. We now also return the
correct transform and proposal IDs.

Fixes #3329.

History

#1 - 04.02.2020 11:13 - Tobias Brunner

- Category set to ikev1

- Status changed from New to Feedback

- Priority changed from High to Normal

> In the case where lifetime 1 does not match lifetime 2, the tunnel will not come up.

What's the idea here? A stronger DH group doesn't require rekeyings that often?

> We are unable to force customers to use IKEv2 or updating their configurations such that lifetime 1 and lifetime *n* are always equivalent.

You really should make them switch to IKEv2. And why not configure a single lifetime?

#2 - 04.02.2020 17:01 - fbh dev

>> In the case where lifetime 1 does not match lifetime 2, the tunnel will not come up.
>
> What's the idea here? A stronger DH group doesn't require rekeyings that often?

Customers are not inclined to modify their configurations since they are using their devices to interconnect multiple sites.

>> We are unable to force customers to use IKEv2 or updating their configurations such that lifetime 1 and lifetime *n* are always equivalent.
>
> You really should make them switch to IKEv2. And why not configure a single lifetime?

We don't have that flexibility since we are servicing a large number of customers.

Let's use this peer configuration as an example:

```
Protection suite of priority 4
encryption algorithm:    AES - Advanced Encryption Standard (128 bit keys).
hash algorithm:        Secure Hash Standard
authentication method:    Pre-Shared Key
Diffie-Hellman group:    2 (1024 bit)
lifetime:        60000 seconds, no volume limit

Protection suite of priority 200
encryption algorithm:    AES - Advanced Encryption Standard (256 bit keys).
hash algorithm:        Secure Hash Standard
authentication method:    Pre-Shared Key
Diffie-Hellman group:    2 (1024 bit)
lifetime:        28800 seconds, no volume limit
```

If the second transform is matched, the lifetime returned is 60000. That isn't correct. This happens irrespective of the algorithms/DH groups used.

#3 - 04.02.2020 17:26 - Tobias Brunner

>> What's the idea here? A stronger DH group doesn't require rekeyings that often?
>
> Customers are not inclined to modify their configurations since they are using their devices to interconnect multiple sites.

Sorry, I don't get that argument.

> Let's use this peer configuration as an example:
>
> ```
> Protection suite of priority 4
> encryption algorithm:    AES - Advanced Encryption Standard (128 bit keys).
> hash algorithm:        Secure Hash Standard
> authentication method:    Pre-Shared Key
> Diffie-Hellman group:    2 (1024 bit)
> lifetime:        60000 seconds, no volume limit
>
> Protection suite of priority 200
> encryption algorithm:    AES - Advanced Encryption Standard (256 bit keys).
> hash algorithm:        Secure Hash Standard
> authentication method:    Pre-Shared Key
> Diffie-Hellman group:    2 (1024 bit)
> lifetime:        28800 seconds, no volume limit
> ```

That makes absolutely no sense at all. What's the point of proposing a lower lifetime with the second proposal? You could obviously avoid the problem by making sure on your end that the first proposal is selected (if you can live with the 128-bit AES key, since the DH group is weak anyway it doesn't really matter, though).

#4 - 04.02.2020 18:30 - fbh dev

> Sorry, I don't get that argument.

They are using global IKE policies that are being shared for multiple IPSec tunnels. So a change in one lifetime impacts the behavior of all other tunnels.

> That makes absolutely no sense at all. What's the point of proposing a lower lifetime with the second proposal? You could obviously avoid the problem by making sure on your end that the first proposal is selected (if you can live with the 128-bit AES key, since the DH group is weak anyway it doesn't really matter, though).

The algorithms specified are just an example and not necessarily reflective of what the actual values are as that is not pertinent. I don't disagree with your point, though that doesn't really help my case either. The reality is that we are encountering many cases of customers having this configuration, and it is handled appropriately by other software stacks like libreswan/openswan. Going to each customer and telling them to change their configuration is a very grueling process, and not necessarily feasible as the router configuration could be managed by a third party.

This is what the RFC says: *"The responder replies in kind but selects, and returns, one transform proposal (the ISAKMP SA attributes)."* This seems to imply that the attributes of the matched transform are returned, not a mix of multiple transforms.

I understand the work-arounds, but I want to know if there is anything that can be done on the strongswan side to alleviate this pain.

#5 - 05.02.2020 10:41 - Tobias Brunner

> > Sorry, I don't get that argument.
>
> They are using global IKE policies that are being shared for multiple IPSec tunnels. So a change in one lifetime impacts the behavior of all other tunnels.

So Cisco can't do different lifetimes per tunnel but they can do it per proposal? Can they do different proposals per tunnel?

> The algorithms specified are just an example and not necessarily reflective of what the actual values are as that is not pertinent.

You wrote: "Let's use this peer configuration as an example"

> I don't disagree with your point, though that doesn't really help my case either. The reality is that we are encountering many cases of customers having this configuration,

Really? Many? I've never seen this before.

> and it is handled appropriately by other software stacks like libreswan/openswan.

strongSwan is mainly an IKEv2 implementation, IKEv1 is just an add-on for legacy reasons. We don't claim compatibility with purposefully weird IKEv1 configs.

> Going to each customer and telling them to change their configuration is a very grueling process, and not necessarily feasible as the router configuration could be managed by a third party.

So?

> This is what the RFC says: *"The responder replies in kind but selects, and returns, one transform proposal (the ISAKMP SA attributes)."* This seems to imply that the attributes of the matched transform are returned, not a mix of multiple transforms.

Yep, but as I said, our IKEv1 implementation is simplified and squeezed into an IKEv2 framework (where lifetimes are not negotiated). So we just copy the lifetimes of the first proposal and return those because they usually don't differ (as mentioned before, they are otherwise ignored).

> I understand the work-arounds, but I want to know if there is anything that can be done on the strongswan side to alleviate this pain.

Sure, there is always the possibility of a patch. But if the effort of modifying code for a [dead protocol] can be avoided by simply changing some configs, why not do so.

#6 - 05.02.2020 10:56 - fbh dev

> So Cisco can't do different lifetimes per tunnel but they can do it per proposal? Can they do different proposals per tunnel?

No, IKEv1 only supports a single proposal for phase 1. Per RFC:

https://tools.ietf.org/html/rfc2409#section-5

Main Mode, Aggressive Mode, and Quick Mode do security association
negotiation. Security Association offers take the form of Tranform
Payload(s) encapsulated in Proposal Payload(s) encapsulated in
Security Association (SA) payload(s). If multiple offers are being
made for phase 1 exchanges (Main Mode and Aggressive Mode) they MUST
take the form of multiple Transform Payloads for a single Proposal
Payload in a single SA payload. To put it another way, for phase 1
exchanges there MUST NOT be multiple Proposal Payloads for a single
SA payload and there MUST NOT be multiple SA payloads. This document
does not proscribe such behavior on offers in phase 2 exchanges.

#7 - 05.02.2020 11:21 - Tobias Brunner

> > So Cisco can't do different lifetimes per tunnel but they can do it per proposal? Can they do different proposals per tunnel?
>
> No, IKEv1 only supports a single proposal for phase 1.

For IKEv1, what we call proposals (a combination of transforms/algorithms, for IKEv1 only a single one of each type) are mapped to transform substructures, which, for IKEv1, also include attributes to negotiate lifetimes. That is, they don't map to proposal substructures in SA payloads there (strongSwan always sends a single one of those unless IPComp is used). The questions still stand.

#8 - 05.02.2020 13:14 - fbh dev

Replied offline.

#9 - 05.02.2020 23:04 - fbh dev

- File 001_lifetime_proposal_match.patch added

#10 - 05.02.2020 23:25 - fbh dev

- File 001_lifetime_proposal_match.patch added

Would you mind reviewing this patch at a high-level? I intend to test it out tomorrow against a cisco device, haven't attempted building it yet. Perhaps we can also merge it up-stream so that it is fixed for future releases. I wrote the patch against 5.7.1, but it's easily adaptable to the master branch.

#11 - 06.02.2020 16:43 - Tobias Brunner

> Would you mind reviewing this patch at a high-level? I intend to test it out tomorrow against a cisco device, haven't attempted building it yet.

I had a quick look. There are definitely some issues, for instance, the return type of get_lifetime_from_transform doesn't match. However, that method is not necessary anyway, because the existing get_lifetime() and get_lifebytes(), which are only used for IKEv1, could just be modified (e.g. add the transform number as argument), same goes for the modifications to sa_payload_t. And the code in transform_substructure_t is redundant (code like that is already in get_life_duration() in proposal_substructure.c). I don't think a setter for the transform number is necessary on proposal_t, just add a separate constructor that takes both numbers.

Anyway, I pushed some commits to the *3329-ikev1-transform-nr* branch. Let me know if those changes work for you.

#12 - 06.02.2020 19:03 - fbh dev

The code for proposal_substructure_create_from_proposals_v1 appears to be wrong. It should start the sequence number from 1 as that is a new request that is being sent from the strongswan side, so it appears to me that the transform #'s should reset from 1 again, which is how it previously was. At that point, we've already identified the lifetime, so there's no need in using the requestor's transform # for the new SA payload.

We probably need to incorporate the same logic for get_auth_method. Otherwise it's looking good.

#13 - 06.02.2020 19:40 - Tobias Brunner

> It should start the sequence number from 1 as that is a new request that is being sent from the strongswan side

It does, because the transform number is not set/0 by default (note the ?: 1 at the end). Only as responder will it be set when the function is called with only the selected proposal.

> At that point, we've already identified the lifetime, so there's no need in using the requestor's transform # for the new SA payload.

It's not a MUST in RFC 2408 (section 4.2 states: "The responder SHOULD retain the Proposal # field in the Proposal payload and the Transform # field in each Transform payload of the selected Proposal."), but I figured why not encode the numbers of the selected proposal/transform if we now have the number available.

> We probably need to incorporate the same logic for get_auth_method.

Do you really have peers that propose different authentication methods in the same SA payload? The authentication method is currently not part of the proposal selection in strongSwan (which only allows configuring a single authentication method per connection).

#14 - 06.02.2020 20:49 - fbh dev

Tobias Brunner wrote:

> > It should start the sequence number from 1 as that is a new request that is being sent from the strongswan side
>
> It does, because the transform number is not set/0 by default (note the ?: 1 at the end). Only as responder will it be set when the function is called with only the selected proposal.
>
> > At that point, we've already identified the lifetime, so there's no need in using the requestor's transform # for the new SA payload.
>
> It's not a MUST in RFC 2408 (section 4.2 states: "The responder SHOULD retain the Proposal # field in the Proposal payload and the Transform # field in each Transform payload of the selected Proposal."), but I figured why not encode the numbers of the selected proposal/transform if we now have the number available.

Ahh, that's a genius hack. This works because when in resonder mode, we only ever have a single transform in the response, correct?

> > We probably need to incorporate the same logic for get_auth_method.
>
> Do you really have peers that propose different authentication methods in the same SA payload? The authentication method is currently not part of the proposal selection in strongSwan (which only allows configuring a single authentication method per connection).

Nah, but I thought for consistency.

Really appreciate the help.

#15 - 07.02.2020 09:59 - Tobias Brunner

> This works because when in resonder mode, we only ever have a single transform in the response, correct?

Yes, two things are true as responder: We only have a single proposal (so the loop and number variable are not really relevant) and that proposal has a transform number set. At least it should, the RFC doesn't explicitly mandate that the numbering starts with 1 (it only has to be monotonically increasing). Because what I noticed is that strongSwan started numbering IKEv1 proposals with 0 (only one is sent unless IPComp is enabled), which I guess is technically not wrong, but e.g. the example in RFC 2408, section 4.2.1 starts numbering proposals with 1. So I've added another commit that changes that, and I also changed the code of the function above and added a comment to clarify the transform numbering.

#16 - 07.02.2020 22:10 - fbh dev

I will do some manual testing next week.
Last thing, how can I run the strongswan tests to make sure they still pass or have you done that already?

#17 - 10.02.2020 11:12 - Tobias Brunner

> Last thing, how can I run the strongswan tests to make sure they still pass or have you done that already?

I have (but it's unlikely to reveal interop issues as strongSwan itself doesn't care for the lifetimes or the proposal/transform numbers). See here for more information on our testing environment.

#18 - 06.03.2020 10:50 - Tobias Brunner

- Tracker changed from Issue to Bug

- Subject changed from IKEv1 tunnel not coming up to Return lifetimes of the actually selected IKEv1 transform

- Status changed from Feedback to Closed

- Assignee set to Tobias Brunner

- Target version set to 5.8.3

- Resolution set to Fixed

Files

001_lifetime_proposal_match.patch	11.3 KB	05.02.2020	fbh dev
001_lifetime_proposal_match.patch	11.3 KB	05.02.2020	fbh dev