strongSwan - Issue #1334

Version 5.3.5, duplicated tunnel aftr IKE_SA rekeyed

03.03.2016 07:19 - Daniel Chan

Status:	Feedback	
Priority:	Normal	
Assignee:		
Category:	charon	
Affected version:	5.3.5	**Resolution:**

Description

Hi,

I used two strongswan5.3.5 as initiator and responder.
pc1----DUT1-----------------DUT2----pc2
(to reproduce quickly ,we set the ikelifetime and lifetime as short as possible)

DUT1:initiator
remote peer：172.32.1.20
ikelifetime=300s
lifetime=150s

DUT2:responder
remote peer：0.0.0.0
ikelifetime=300s
lifetime=150s

At the start, the IKE_SA and the CHILD_SA can established normally like this:

```
Connections:
    test_ini:  172.32.1.100...172.32.1.20  IKEv1, dpddelay=10s
    test_ini:   local:  [172.32.1.100] uses pre-shared key authentication
    test_ini:   remote: [172.32.1.20] uses pre-shared key authentication
    test_ini:   child:  192.168.1.0/24 === 192.168.0.0/24 TUNNEL, dpdaction=restart
Security Associations (1 up, 0 connecting):
    test_ini[1]: ESTABLISHED 3 seconds ago, 172.32.1.100[172.32.1.100]...172.32.1.20[172.32.1.20]
    test_ini[1]: IKEv1 SPIs: 2af0f37f10a91f53_i* adba57e03b4bfe84_r, pre-shared key reauthenticati
on in 4 minutes
    test_ini[1]: IKE proposal: 3DES_CBC/HMAC_MD5_96/PRF_HMAC_MD5/MODP_1024
    test_ini{1}:  INSTALLED, TUNNEL, reqid 1, ESP SPIs: cb7971b3_i cc475099_o
    test_ini{1}:  3DES_CBC/HMAC_MD5_96, 0 bytes_i, 0 bytes_o, rekeying in 117 seconds
    test_ini{1}:   192.168.1.0/24 === 192.168.0.0/24
```

after several minutes, there is another duplicated tunnel.Sometimes, the duplicated tunnel will disappear but sometimes it won't.

```
Connections:
    test_ini:  172.32.1.100...172.32.1.20  IKEv1, dpddelay=10s
    test_ini:   local:  [172.32.1.100] uses pre-shared key authentication
    test_ini:   remote: [172.32.1.20] uses pre-shared key authentication
    test_ini:   child:  192.168.1.0/24 === 192.168.0.0/24 TUNNEL, dpdaction=restart
Security Associations (2 up, 0 connecting):
    test_ini[5]: ESTABLISHED 3 minutes ago, 172.32.1.100[172.32.1.100]...172.32.1.20[172.32.1.20]
    test_ini[5]: IKEv1 SPIs: 5db344aa30722c0d_i 36fb0f4dbfe100ef_r*, pre-shared key reauthenticati
on in 39 seconds
    test_ini[5]: IKE proposal: 3DES_CBC/HMAC_MD5_96/PRF_HMAC_MD5/MODP_1024
    test_ini{9}:  INSTALLED, TUNNEL, reqid 1, ESP SPIs: cd9f7cd4_i c3d7699a_o
    test_ini{9}:  3DES_CBC/HMAC_MD5_96, 0 bytes_i, 0 bytes_o, rekeying in 68 seconds
    test_ini{9}:   192.168.1.0/24 === 192.168.0.0/24
    test_ini[4]: ESTABLISHED 3 minutes ago, 172.32.1.100[172.32.1.100]...172.32.1.20[172.32.1.20]
    test_ini[4]: IKEv1 SPIs: 4d13d9b46230805a_i* 5dfe42fb1840dd97_r, pre-shared key reauthenticati
on in 39 seconds
```

```
    test_ini[4]: IKE proposal: 3DES_CBC/HMAC_MD5_96/PRF_HMAC_MD5/MODP_1024
    test_ini{8}:  INSTALLED, TUNNEL, reqid 1, ESP SPIs: c2eb6d69_i c103ec96_o
    test_ini{8}:  3DES_CBC/HMAC_MD5_96, 0 bytes_i, 0 bytes_o, rekeying in 68 seconds
    test_ini{8}:   192.168.1.0/24 === 192.168.0.0/24
```

The responder's log:

```
Jan  1 00:27:56 46[IKE] <4> received XAuth vendor ID

Jan  1 00:27:56 46[IKE] <4> received DPD vendor ID

Jan  1 00:27:56 46[IKE] <4> received NAT-T (RFC 3947) vendor ID

Jan  1 00:27:56 46[IKE] <4> received draft-ietf-ipsec-nat-t-ike-02\n vendor ID

Jan  1 00:27:56 46[IKE] <4> 172.32.1.100 is initiating a Main Mode IKE_SA

Jan  1 00:27:56 38[IKE] <test_res|3> initiator did not reauthenticate as requested

Jan  1 00:27:56 38[IKE] <test_res|3> reauthenticating IKE_SA test_res[3] actively

Jan  1 00:27:56 38[IKE] <test_res|3> initiating Main Mode IKE_SA test_res[5] to 172.32.1.100

Jan  1 00:27:56 47[IKE] <test_res|5> received XAuth vendor ID
Jan  1 00:27:56 47[IKE] <test_res|5> received DPD vendor ID

Jan  1 00:27:56 47[IKE] <test_res|5> received NAT-T (RFC 3947) vendor ID
Jan  1 00:27:56 03[CFG] <4> looking for pre-shared key peer configs matching 172.32.1.20...172.32.
1.100[172.32.1.100]
Jan  1 00:27:56 03[CFG] <4> selected peer config "test_res"
Jan  1 00:27:56 03[IKE] <test_res|4> IKE_SA test_res[4] established between 172.32.1.20[172.32.1.2
0]...172.32.1.100[172.32.1.100]
Jan  1 00:27:56 03[IKE] <test_res|4> scheduling reauthentication in 270s
Jan  1 00:27:56 03[IKE] <test_res|4> maximum IKE_SA lifetime 300s

Jan  1 00:27:56 02[IKE] <test_res|5> IKE_SA test_res[5] established between 172.32.1.20[172.32.1.2
0]...172.32.1.100[172.32.1.100]
Jan  1 00:27:56 02[IKE] <test_res|5> scheduling reauthentication in 270s

Jan  1 00:27:56 02[IKE] <test_res|5> maximum IKE_SA lifetime 300s
```

The initiator's log:

```
Jan  1 00:39:04 46[IKE] <test_ini|1> sending DPD request
Jan  1 00:39:14 54[IKE] <test_ini|1> sending DPD request
Jan  1 00:39:14 19[IKE] <test_ini|1> reauthenticating IKE_SA test_ini[1]
Jan  1 00:39:14 19[IKE] <test_ini|1> initiating Main Mode IKE_SA test_ini[2] to 172.32.1.20
Jan  1 00:39:14 09[IKE] <3> received XAuth vendor ID
Jan  1 00:39:14 09[IKE] <3> received DPD vendor ID
Jan  1 00:39:14 09[IKE] <3> received NAT-T (RFC 3947) vendor ID
Jan  1 00:39:14 09[IKE] <3> received draft-ietf-ipsec-nat-t-ike-02\n vendor ID
Jan  1 00:39:14 09[IKE] <3> 172.32.1.20 is initiating a Main Mode IKE_SA
Jan  1 00:39:14 36[IKE] <test_ini|2> received XAuth vendor ID
Jan  1 00:39:14 36[IKE] <test_ini|2> received DPD vendor ID
Jan  1 00:39:14 36[IKE] <test_ini|2> received NAT-T (RFC 3947) vendor ID
Jan  1 00:39:14 11[CFG] <3> looking for pre-shared key peer configs matching 172.32.1.100...172.32
.1.20[172.32.1.20]
Jan  1 00:39:14 11[CFG] <3> selected peer config "test_ini"
Jan  1 00:39:14 11[IKE] <test_ini|3> IKE_SA test_ini[3] established between 172.32.1.100[172.32.1.
100]...172.32.1.20[172.32.1.20]
Jan  1 00:39:14 11[IKE] <test_ini|3> scheduling reauthentication in 270s
Jan  1 00:39:14 11[IKE] <test_ini|3> maximum IKE_SA lifetime 300s
Jan  1 00:39:14 15[IKE] <test_ini|2> IKE_SA test_ini[2] established between 172.32.1.100[172.32.1.
100]...172.32.1.20[172.32.1.20]
```

```
Jan  1 00:39:14 15[IKE] <test_ini|2> scheduling reauthentication in 270s
Jan  1 00:39:14 15[IKE] <test_ini|2> maximum IKE_SA lifetime 300s
```

From the log,it seems that the initiator and the responder reauthenticating the IKE_SA at the same time, so there is another duplicated tunnel,
Normally the initiator can detect the duplicated tunnel and delete it. **But sometimes the initiator can't detect the duplicated tunnel** ,so there are two same tunnels.
By the way, the communication is still OK.

Is this problem a known issue? Or anything wrong with my ipsec.conf in the attachment

History

#1 - 03.03.2016 07:48 - Daniel Chan

- File ipsec.conf of ini.txt added

- File ipsec.conf of res.txt added

the whole file of ipsec.conf

#2 - 03.03.2016 10:12 - Tobias Brunner

- Description updated

- Status changed from New to Feedback

If you configure the lifetimes the same on both hosts and the margin is quite low chances are high that both ends rekey (or reauthenticate) at about the same time. And like initiating SAs at the same time this could lead to duplicates (the duplicate detection can currently not catch all of these depending on the timing). If the lifetimes are higher (and thus the margin is too) the randomization (via *rekeyfuzz*, which you have disabled in your config) should reduce the chances that both peers reauthenticate concurrently. Also refer to ExpiryRekey for details on the rekey time calculation.

#3 - 08.03.2016 02:59 - Daniel Chan

Tobias Brunner wrote:

> If you configure the lifetimes the same on both hosts and the margin is quite low chances are high that both ends rekey (or reauthenticate) at about the same time. And like initiating SAs at the same time this could lead to duplicates (the duplicate detection can currently not catch all of these depending on the timing). If the lifetimes are higher (and thus the margin is too) the randomization (via *rekeyfuzz*, which you have disabled in your config) should reduce the chances that both peers reauthenticate concurrently. Also refer to ExpiryRekey for details on the rekey time calculation.

Hi Tobias Brunner,

Thanks for your answer, so you mean, we'd better make the two peers not to reauthenticate concurrently via modifing "rekeyfuzz"? And what't the meaning of "the duplicate detection can currently not catch all of these **depending on the timing**", do you have any idea to make the duplicate detection more reliable?

#4 - 08.03.2016 09:17 - Tobias Brunner

> so you mean, we'd better make the two peers not to reauthenticate concurrently via modifing "rekeyfuzz"?

Yes, absolutely. That's what this setting is for.

> And what't the meaning of "the duplicate detection can currently not catch all of these **depending on the timing**", do you have any idea to make the duplicate detection more reliable?

Not at the moment.

#5 - 09.03.2016 09:58 - Daniel Chan

Tobias Brunner wrote:

> > so you mean, we'd better make the two peers not to reauthenticate concurrently via modifing "rekeyfuzz"?
>
> Yes, absolutely. That's what this setting is for.
>
> > And what't the meaning of "the duplicate detection can currently not catch all of these **depending on the timing**", do you have any idea to

> > make the duplicate detection more reliable?
>
> Not at the moment.

Hi Tobias Brunner,

And I found another thing, when the charon received main mode and detected reauth of existing IKE_SA, it would adopting the CHILD_SA of the old IKE_SA and delete the old IKE_SA after 10s. But before the old IKE_SA was deleted, it was also able to reauthenticate IKE_SA again, which would cause the new IKE_SA and its CHILD_SA been deleted.(we can see from the log)

```
Jan  1 01:42:29 24[CFG] <50> looking for pre-shared key peer configs matching 172.32.1.200...172.32.1.20[172.3
2.1.20]
Jan  1 01:42:29 24[CFG] <50> selected peer config "Y_L2tp_client_ini" 
Jan  1 01:42:29 24[IKE] <Y_L2tp_client_ini|50> IKE_SA Y_L2tp_client_ini[50] established between 172.32.1.200[1
72.32.1.200]...172.32.1.20[172.32.1.20]
Jan  1 01:42:29 24[IKE] <Y_L2tp_client_ini|50> scheduling reauthentication in 258s
Jan  1 01:42:29 24[IKE] <Y_L2tp_client_ini|50> maximum IKE_SA lifetime 288s
Jan  1 01:42:29 24[IKE] <Y_L2tp_client_ini|49> detected reauth of existing IKE_SA, adopting 3 children and 0 v
irtual IPs
Jan  1 01:42:34 22[IKE] <Y_L2tp_client_ini|49> reauthenticating IKE_SA Y_L2tp_client_ini[49]
Jan  1 01:42:34 22[IKE] <Y_L2tp_client_ini|49> initiating Main Mode IKE_SA Y_L2tp_client_ini[51] to 172.32.1.2
0
Jan  1 01:42:34 40[IKE] We have all 3 child_sas now.
Jan  1 01:42:34 40[IKE] <Y_L2tp_client_ini|51> received XAuth vendor ID
Jan  1 01:42:34 40[IKE] <Y_L2tp_client_ini|51> received DPD vendor ID
Jan  1 01:42:34 40[IKE] <Y_L2tp_client_ini|51> received NAT-T (RFC 3947) vendor ID
Jan  1 01:42:35 51[IKE] We have all 3 child_sas now.
Jan  1 01:42:35 05[IKE] We have all 3 child_sas now.
Jan  1 01:42:35 05[IKE] <Y_L2tp_client_ini|51> IKE_SA Y_L2tp_client_ini[51] established between 172.32.1.200[1
72.32.1.200]...172.32.1.20[172.32.1.20]
Jan  1 01:42:35 05[IKE] <Y_L2tp_client_ini|51> scheduling reauthentication in 247s
Jan  1 01:42:35 05[IKE] <Y_L2tp_client_ini|51> maximum IKE_SA lifetime 277s
Jan  1 01:42:41 09[IKE] We have all 3 child_sas now.
Jan  1 01:42:41 09[IKE] <Y_L2tp_client_ini|50> CHILD_SA test8_ini{101} established with SPIs c280c433_i cdf08d
02_o and TS 192.168.1.0/24 === 192.168.0.0/24
Jan  1 01:42:42 07[IKE] We have all 4 child_sas now.
Jan  1 01:42:42 07[IKE] <Y_L2tp_client_ini|51> In process_r rekey=0 false=0 
Jan  1 01:42:42 46[IKE] We have all 4 child_sas now.
Jan  1 01:42:42 46[IKE] <Y_L2tp_client_ini|51> CHILD_SA test8_ini{102} established with SPIs ca15a161_i c7eed7
f9_o and TS 192.168.1.0/24 === 192.168.0.0/24
Jan  1 01:42:45 01[IKE] We have all 4 child_sas now.
Jan  1 01:42:45 01[IKE] <Y_L2tp_client_ini|50> received DELETE for IKE_SA Y_L2tp_client_ini[50]
Jan  1 01:42:45 01[IKE] <Y_L2tp_client_ini|50> deleting IKE_SA Y_L2tp_client_ini[50] between 172.32.1.200[172.
32.1.200]...172.32.1.20[172.32.1.20]
```

So, what can we do to avoid this condition? Do you think set the margintime bigger and try to make the gap of two peer's rekey time bigger than 10s is reasonable way? Or any better idea? Thanks

#6 - 09.03.2016 10:31 - Tobias Brunner

> And I found another thing, when the charon received main mode and detected reauth of existing IKE_SA, it would adopting the CHILD_SA of the old IKE_SA and delete the old IKE_SA after 10s. But before the old IKE_SA was deleted, it was also able to reauthenticate IKE_SA again, which would cause the new IKE_SA and its CHILD_SA been deleted.(we can see from the log)

I suppose, after adopting the CHILD_SAs, we could set the "old" IKE_SA in a state (e.g. IKE_PASSIVE or a new state) that prevents it from getting reauthenticated actively.

> So, do you agree that the gap of two peers's rekey time is better to bigger than 10s?

That would make sense anyway. A higher *margintime* will increase the chances that the difference is more than 10s (again see ExpiryRekey for the formula). And depending on your use case you might not need to reauthenticate very often.

#7 - 10.03.2016 03:47 - Daniel Chan

> after adopting the CHILD_SAs, we could set the "old" IKE_SA in a state (e.g. IKE_PASSIVE or a new state) that prevents it from getting reauthenticated actively.

Hi Tobias Brunner,

Thanks for your suggestion, is there any existing patch can do this? I think this modification is quite requisite.

#8 - 10.03.2016 11:06 - Tobias Brunner

> > after adopting the CHILD_SAs, we could set the "old" IKE_SA in a state (e.g. IKE_PASSIVE or a new state) that prevents it from getting reauthenticated actively.
>
> Thanks for your suggestion, is there any existing patch can do this?

No, currently not.

Files

ipsec.conf of ini and res.txt	1.06 KB	03.03.2016	Daniel Chan
ipsec.conf of ini.txt	1.36 KB	03.03.2016	Daniel Chan
ipsec.conf of res.txt	1.41 KB	03.03.2016	Daniel Chan