strongSwan - Bug #2374

Charon/vici deadlock when auto=start and acting as responder at the same time

03.07.2017 10:31 - Eyal Birger

Status:	Closed	**Start date:**	03.07.2017
Priority:	Normal	**Due date:**	
Assignee:	Tobias Brunner	**Estimated time:**	0.00 hour
Category:	libcharon		
Target version:	5.6.0		
Affected version:	5.5.1	**Resolution:**	Fixed

Description

Similar in nature to issue #1185 (https://wiki.strongswan.org/issues/1185)

Strongswan instances on machines A, B.
Connections are loaded using VICI, pointing towards each other (A->B, B->A). auto=start.

Seems that if an IKE negotiation of the same conn is in progress during the load-conn operation,
a deadlock situation occurs.

Stack trace on hanging machine:

VICI thread:

```
#0 pthread_cond_wait@@GLIBC_2.3.2 () at ../sysdeps/unix/sysv/linux/x86_64/pthread_cond_wait.S:185
#1 0x00007ff2ea2cd993 in wait_ (this=0x8832c0, mutex=0x86cdb0) at threading/mutex.c:231
#2 0x00007ff2ea0318c4 in wait_for_entry (entry=0x884040, segment=<optimized out>, this=0x86cc60) a
t sa/ike_sa_manager.c:737
#3 0x00007ff2ea031d8e in checkout_by_config (this=0x86cc60, peer_cfg=0x7ff2ac008280) at sa/ike_sa_
manager.c:1414
#4 0x00007ff2ea012033 in initiate_execute (job=job@entry=0x7ff2ac0087f0) at control/controller.c:4
09
#5 0x00007ff2ea012b2d in initiate (this=<optimized out>, peer_cfg=0x7ff2ac008280, child_cfg=child_
cfg@entry=0x7ff2ac001840, callback=callback@entry=0x0, param=param@entry=0x0, timeout=timeout@entr
y=0, limits=false)
at control/controller.c:528
#6 0x00007ff2e4bbc367 in run_start_action (peer_cfg=peer_cfg@entry=0x7ff2ac008280, child_cfg=<opti
mized out>, this=0x875c90) at vici_config.c:1696
#7 0x00007ff2e4bbcdd4 in handle_start_actions (this=this@entry=0x875c90, peer_cfg=peer_cfg@entry=0
x7ff2ac008280, undo=undo@entry=false) at vici_config.c:1883
#8 0x00007ff2e4bbdcd5 in merge_config (peer_cfg=0x7ff2ac008280, this=0x875c90) at vici_config.c:19
60
#9 _cb_config_sn (request=request@entry=0x7ff2dd174bd0, message=message@entry=0x7ff2ac005c90, ctx=
ctx@entry=0x7ff2dd174b60, name=<optimized out>) at vici_config.c:2146
#10 0x00007ff2e4bb1dab in parse (this=0x7ff2ac005c90, ctx=0x7ff2dd174b60, section=0x7ff2e4bbcf50 <
_cb_config_sn>, kv=0x0, li=0x0, user=0x7ff2dd174bd0) at vici_message.c:532
#11 0x00007ff2e4bba90b in _cb_load_conn (this=<optimized out>, name=<optimized out>, id=<optimized
 out>, message=<optimized out>) at vici_config.c:2158
#12 0x00007ff2e4bb3b96 in process_request (this=this@entry=0x87f020, name=name@entry=0x7ff2dd174c8
0 "load-conn", id=id@entry=11, data=...) at vici_dispatcher.c:289
#13 0x00007ff2e4bb3f31 in _cb_inbound (this=0x87f020, id=11, data=...) at vici_dispatcher.c:346
#14 0x00007ff2e4bb0aee in _cb_process_queue (sel=0x7ff2ac006090) at vici_socket.c:508
#15 0x00007ff2ea2bb0c2 in execute (this=<optimized out>) at processing/jobs/callback_job.c:77
#16 0x00007ff2ea2bb9fb in process_job (worker=0x87ab60, this=0x83ced0) at processing/processor.c:2
35
#17 process_jobs (worker=0x87ab60) at processing/processor.c:321
#18 0x00007ff2ea2ccdbb in thread_main (this=0x87ab90) at threading/thread.c:331
#19 0x00007ff2e9de66ba in start_thread (arg=0x7ff2dd175700) at pthread_create.c:333
#20 0x00007ff2e9b1c3dd in clone () at ../sysdeps/unix/sysv/linux/x86_64/clone.S:109
```

IKE negotiation thread:

```
#0 __lll_lock_wait () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:135
#1 0x00007ff2e9de8dbd in _GI__pthread_mutex_lock (mutex=0x7ff2ac007f18) at ../nptl/pthread_mutex_lock.c:80
#2 0x00007ff2ea2cd5ca in lock (this=<optimized out>) at threading/mutex.c:106
#3 0x00007ff2ea00f4cc in create_child_cfg_enumerator (this=this@entry=0x7ff2ac008280) at config/peer_cfg.c:357
#4 0x00007ff2ea00f549 in select_child_cfg (this=0x7ff2ac008280, my_ts=0x7ff2a0007750, other_ts=0x7ff2a0004ed0, my_hosts=0x7ff2a0008d80, other_hosts=0x7ff2a0008e50) at config/peer_cfg.c:417
#5 0x00007ff2ea042415 in select_child_cfg (this=0x8869f0) at sa/ikev2/tasks/child_create.c:1204
#6 build_r (this=0x8869f0, message=0x7ff2a0009130) at sa/ikev2/tasks/child_create.c:1285
#7 0x00007ff2ea03c627 in build_response (request=0x7ff2a00020b0, this=0x883f00) at sa/ikev2/task_manager_v2.c:839
#8 process_request (message=<optimized out>, this=0x883f00) at sa/ikev2/task_manager_v2.c:1187
#9 process_message (this=0x883f00, msg=<optimized out>) at sa/ikev2/task_manager_v2.c:1452
#10 0x00007ff2ea02f257 in process_message (this=0x87c100, message=0x7ff2a00020b0) at sa/ike_sa.c:1559
#11 0x00007ff2ea03b63b in handle_fragment (defrag=defrag@entry=0x883fd0, msg=msg@entry=0x7ff2d0001120, this=0x883f00) at sa/ikev2/task_manager_v2.c:1234
#12 0x00007ff2ea03bed3 in process_message (this=0x883f00, msg=0x7ff2d0001120) at sa/ikev2/task_manager_v2.c:1442
#13 0x00007ff2ea02f257 in process_message (this=0x87c100, message=0x7ff2d0001120) at sa/ike_sa.c:1559
#14 0x00007ff2ea027471 in execute (this=0x7ff2d0000ac0) at processing/jobs/process_message_job.c:74
#15 0x00007ff2ea2bb9fb in process_job (worker=0x87b140, this=0x83ced0) at processing/processor.c:235
#16 process_jobs (worker=0x87b140) at processing/processor.c:321
#17 0x00007ff2ea2ccdbb in thread_main (this=0x87b170) at threading/thread.c:331
#18 0x00007ff2e9de66ba in start_thread (arg=0x7ff2dc173700) at pthread_create.c:333
#19 0x00007ff2e9b1c3dd in clone () at ../sysdeps/unix/sysv/linux/x86_64/clone.S:109
```

Associated revisions

Revision 8d4ebb3a - 27.07.2017 13:34 - Tobias Brunner

peer-cfg: Use an rwlock instead of a mutex to safely access child-cfgs

If multiple threads want to enumerate child-cfgs and potentially lock
other locks (e.g. check out IKE_SAs) while doing so a deadlock could
be caused (as was the case with VICI configs with start_action=start).
It should also improve performance for roadwarrior connections and lots
of clients connecting concurrently.

Fixes #2374.

History

#1 - 03.07.2017 10:32 - Eyal Birger

Looks like the IKE negotiation thread is waiting for the child cfg, which is held by the VICI thread,
and the VICI thread is waiting for the IKE SA, held by the IKE negotiation thread.

#2 - 03.07.2017 10:58 - Eyal Birger

This deadlock occurred on version 5.5.1. Relevant code parts do not appear to have been changed in recent versions.

#3 - 03.07.2017 16:10 - Tobias Brunner

- Description updated

- Status changed from New to Feedback

- Target version set to 5.6.0

- Affected version changed from 5.5.3 to 5.5.1

I see. The problem is that the peer_cfg's internal mutex is locked while enumerating the child_cfgs to perform the start action for each CHILD_SA (requiring checking IKE_SAs if *charon.reuse_ikesa* is enabled), while the other thread, holding an IKE_SA locked, wants to acquire the same mutex to select the child_cfg's.

For a possible solution we could probably use an rwlock instead of a mutex so both threads could enumerate the child_cfgs concurrently (might not be a bad idea anyway in case of roadwarrior connections with lots of clients). I did that in the *2374-peer-cfg-rwlock* branch.

A workaround for this particular situation is to disable *charon.reuse_ikesa*, then checkout_by_config won't enumerate existing IKE_SAs.

#4 - 27.07.2017 13:35 - Tobias Brunner

- Status changed from Feedback to Closed

- Assignee set to Tobias Brunner

- Resolution set to Fixed

#5 - 28.07.2017 03:18 - Eyal Birger

Thanks for addressing this (sorry for taking long to ack).