strongSwan - Bug #553

IKEv2 reauthentication fails if virtual IP changes

25.03.2014 11:42 - Anonymous

Status:	Closed	**Start date:**	25.03.2014
Priority:	Normal	**Due date:**	
Assignee:	Tobias Brunner	**Estimated time:**	0.00 hour
Category:	libcharon		
Target version:	5.2.0		
Affected version:	5.1.0	**Resolution:**	Fixed

Description

I have a setup where the responder assigns a virtual IP to the initiator, using

```
leftsourceip=%config
```

on the initiator side. The responder requests the virtual IP from DHCP using the DHCP plugin with default random MAC setting. This works fine on connection start, but strangely breaks IKE reauthentication.

What happens is the following: The IKE_AUTH request for reauthentication includes traffic selectors which use only the current virtual IP on the initiator side. However, the responder initiates a new DHCP request, receives a new IP which usually does not match the previous one, thus is incompatible with the requested TS. As a result, the responder sends an IKE_AUTH response with TS_UNACCEPTABLE and reauthentication fails, leading to the connection breaking. Restarting the connection works, however, because in that case the initiator once again accepts the IP and TS provided by the responder. In my test setup with only one initiator I can work around the issue by using an IP pool, because that way the initiator will always be assigned the first address in the pool, avoiding the conflict. However, the specification I'm implementing here requires that IP addresses be randomly assigned to initiators, so that is not acceptable as a permanent solution (and the bug should be fixed anyway).

RFC 5996, Section 2.8.3 says:

> IKEv2 does not have any special support for reauthentication.
> Reauthentication is done by creating a new IKE SA from scratch (using
> IKE_SA_INIT/IKE_AUTH exchanges, without any REKEY_SA Notify
> payloads), creating new Child SAs within the new IKE SA (without
> REKEY_SA Notify payloads), and finally deleting the old IKE SA (which
> deletes the old Child SAs as well).

My interpretation of "creating a new IKE SA from scratch" is that assigning a new IP is entirely valid, so I think the bug is on the initiator side, and should be fixed by letting the initiator accept changed IP and TS.

I've first noticed the problem in Strongswan 5.1.1 from Debian wheezy-backports, and I could confirm it is still present in current master (commit 2d79f6d81e5e2abf8926b8170f1023b0a88d1d50). I have not tested older versions.

Associated revisions

Revision 4469e3d0 - 15.04.2014 16:19 - Tobias Brunner

ikev2: Fix reauthentication if peer assigns a different virtual IP

Before this change a reqid set on the create_child_t task was used as
indicator of the CHILD_SA being rekeyed. Only if that was not the case
would the local traffic selector be changed to 0.0.0.0/0|::/0 (as we
don't know which virtual IP the gateway will eventually assign).
On the other hand, in case of a rekeying the VIP is expected to remain
the same, so the local TS would simply equal the VIP.

Since c949a4d5016e33c5 reauthenticated CHILD_SAs also have the reqid
set. Which meant that the local TS would contain the previously
assigned VIP, basically rendering the gateway unable to assign a
different VIP to the client as the resulting TS would not match
the client's proposal anymore.

Fixes #553.

History

#1 - 15.04.2014 16:27 - Tobias Brunner

- Category set to charon

- Status changed from New to Closed

- Assignee set to Tobias Brunner

- Target version set to 5.2.0

- Affected version changed from dr|rc|master to 5.1.0

- Resolution set to Fixed

Thanks for the analysis.

One of the reasons strongSwan implements reauthentication by terminating the existing IKE_SA before re-initiating the SA from scratch (and not how it is described in RFC 5996) is the potential problems with virtual IP addresses. But mainly the other way around, it would be quite difficult for a responder to assign the same IP address to a client (which some admins like to do), if that address is already used by another IKE_SA (the new IKE_SA has no affiliation whatsoever with the existing one).

Anyway, this is definitely a regression in releases since 5.1.0, subtly introduced with c949a4d5016e33c5. By reusing the *reqid* we fixed issues with *auto=route* and reauthentication (or *close|dpdaction=restart*). But this led to an unintended behavior change in the *child_create_t* task, which used the reqid as an indicator of a rekeying taking place (in which case the traffic selector is expected to stay the same).

The associated commit fixes the issue by using the correct flag to detect the rekeying.

As a workaround you could set *reauth=no* on both peers to disable reauthentication and use inline rekeying instead.

#2 - 15.04.2014 16:27 - Tobias Brunner

- Category changed from charon to libcharon