Clarifications for Using TCP Encapsulation in IKEv2

draft-smyslov-ipsecme-tcp-guidelines

Valery Smyslov

svan@elvis.ru

IETF 103

TCP Encapsulation in IKEv2

- Defined in RFC 8229
- Modifies IKEv2 behavior in various situations
  - original Initiator is responsible to restore TCP connection if it is broken
  - with MOBIKE in case IP address is changed first try UDP and then switch to TCP
  - NAT keepalives are redundant
  - IKE Fragmentation is redundant
  - etc.
- However, some changes in behavior are missing. Most of them are for optimization, however few are needed for reliability and interoperability

Retransmissions

- RFC 7296 requires exchange initiator to retransmit request periodically until either response is received or the SA is deemed to have failed
- TCP is reliable protocol, there is generally no need to retransmit
  - in congested networks retransmitting requests can increase congestion making things worse
- However, if TCP connection is lost and then restored, then IKE implementation must retransmit all outstanding requests

Using COOKIE and PUZZLE

- Using COOKIE allows responder to make sure the initiator’s IP address is real
- In general COOKIE is less useful with TCP
  - TCP itself verifies that initiator’s IP address is real
  - TCP creates states before packet reaches IKE, that violates stateless nature of COOKIE
- Using PUZZLE still makes sense
- if COOKIE (or PUZZLE) request is sent by responder
  - TCP connection should be immediately closed (to keep responder at least partially stateless)
  - COOKIE calculation must not include initiator’s port number (since it will most probably be different)

Error Handling in IKE_SA_INIT

- RFC 7296 advises initiator not to act immediately if error notification is received in IKE_SA_INIT because it can be forged; instead wait for more responses
- With TCP this makes little sense
  - if this is genuine message from responder, then other responses won't be sent
  - if TCP is hijacked and this is message is forged by attacker, then genuine response won't be received or will be corrupted (because TCP sequence numbers will already be consumed by attacker's message)

Interaction with MOBIKE

- RFC 4555 defines MOBIKE protocol
- RFC 8229 recommends, that if IP is changed, then initiator first try to send UPDATE_IP_ADDRESSES notify using UDP and then switch to TCP if no response is received
- When switching to TCP
  - the content of the NAT_DETECTION_*_IP notifications must be recalculated if source/destination ports differ from UDP's
  - Message ID for TCP-based exchange must remain the same as for (failed) corresponding UDP-based one

Interaction with High Availability Clusters

- RFC 6311 defines IKE Message ID & ESP SN synchronization mechanism between IKE peer and HA cluster
  - when cluster failover takes place the new active node initiates INFORMATIONAL exchange containing new Message IDs & SN gap
- In case of cluster failover the existing TCP connection is broken and the new active node cannot initiate the exchange until the client restores it (by sending fresh IKE or ESP packet)
  - the client is unaware of the fact that the connection is broken, so if it has nothing to send, the connection won’t be restored for a long time, and the cluster would eventually tear down the IKE SA
- Advise clients to send Liveness Check messages periodically if the partner is HA cluster and there is no outgoing ESP traffic?
- Or make clusters wait until TCP connection is restored after failover (probably for a long time)?

TCP Proxies

- TCP encapsulation can be implemented using proxy and unmodified IKE
- However, in this case the behavior would differ, making protocol less reliable
  - If original initiator is unmodified IKE behind TCP proxy
    - TCP connection won’t always be restored in timely fashion if it is broken and initiator has nothing to send
    - in case of MOBIKE initiator won’t first try UDP and then TCP if local IP is changed
  - If original responder is unmodified IKE behind TCP proxy
    - if TCP connection is broken responder will still try to send packets (if there are any) and probably times out before the connection is restored by initiator
  - peers would always think there is a NAT in between, NAT keepalives will be sent

Way Forward

- Comments? Questions?
- More issues with TCP encapsulation?
- Time for RFC8229–bis?

Thank you!