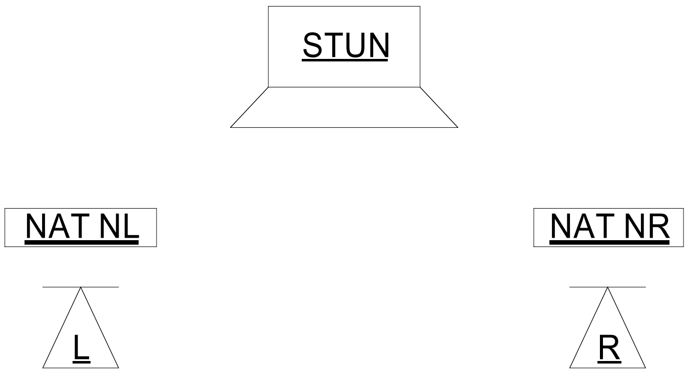
Example: Two Endpoint-Independent Mapping, Address-Dependent Filtering NATs

[See ReadMe document for notation and conventions used]



L and R are behind two different NATs (labeled NL and NR respectively). Each NAT is [BEHAVE-UDP] compliant, but has the address-dependent filtering property. L and R both use a public STUN server, but this server does not support the STUN Relay usage (= no TURN).

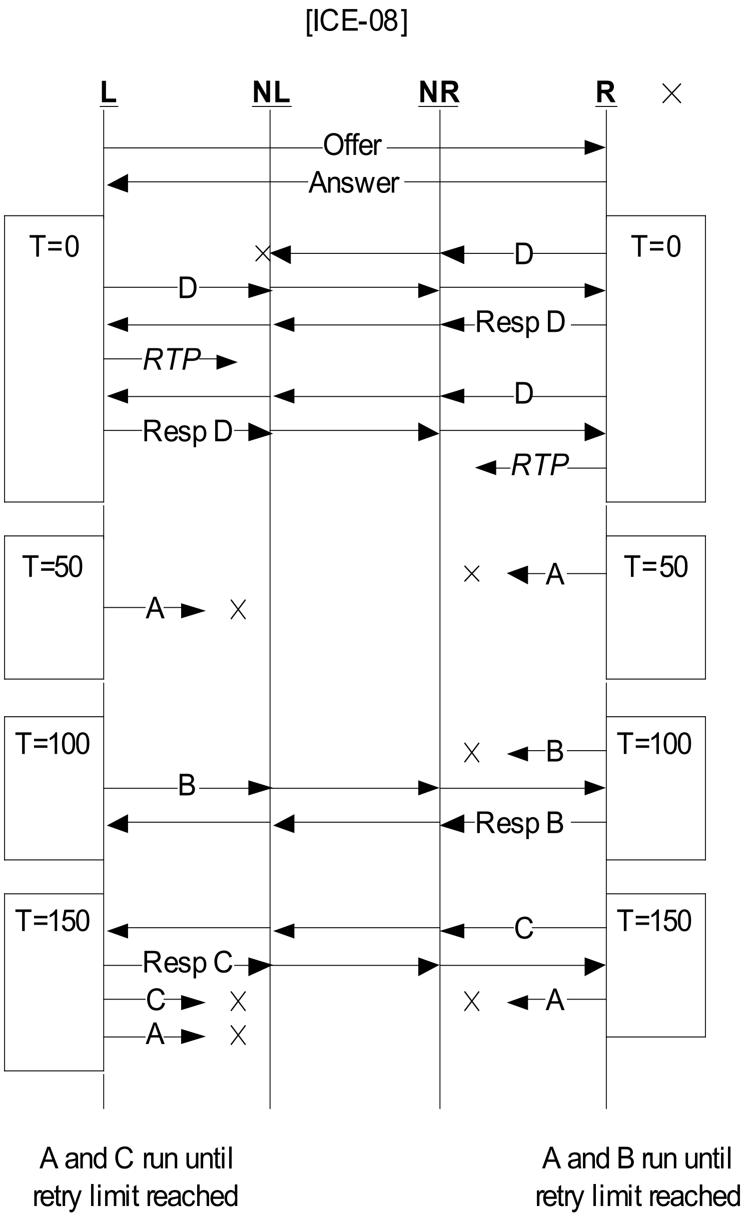
The candidates offered by L and R are:

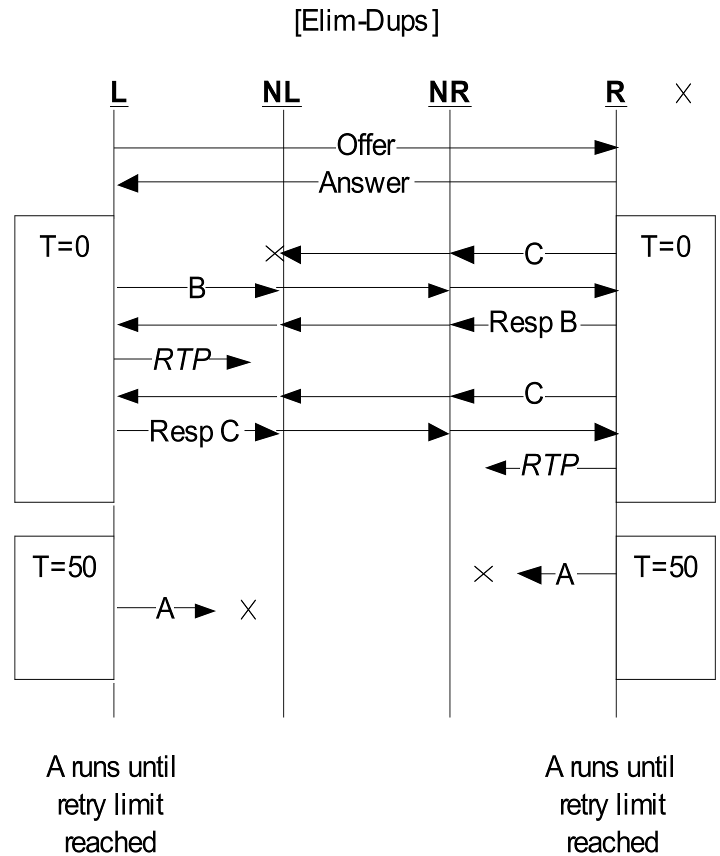
- L_1 – A local candidate; q = 1
- L_2 – A server-reflexive candidate; q = 0.7
- R_1 – A local candidate; q = 1
- R_2 – A server-reflexive candidate; q = 0.7

In this example, L and R choose L_2 and R_2 respectively as the initially active candidates. Thus (L2, R2) is the first pair in the [ICE-08] check ordering. In [Elim-Dups], there are no Tx candidate pairs that directly correspond to (L2, R2), since neither candidate is a base candidate, but this pair is equivalent to the checks ($L_1 \rightarrow R_2$) and ($L_2 \leftarrow R_1$) so these checks are done first in the [Elim-Dups] check ordering.

Note how [ICE-08] needs 8 checks (one in each direction for each of the 4 candidate pairs), while [Elim-Dups] needs only 4 checks (since [Elim-Dups] only does those checks that originate from a base candidate).

Label	ICE-08 candidate pairs and their check ordering	Tx pairs on L and their check ordering	Tx pairs on R and their check ordering
A	(L_1, R_1) 2nd	$L_1 \rightarrow R_1$ 2nd	$L_1 \leftarrow R_1$ 2nd
B	(L_1, R_2) 3rd	$L_1 \rightarrow R_2$ 1st	
C	(L_2, R_1) 4th		$L_2 \leftarrow R_1$ 1st
D	(L_2, R_2) 1st		


[ICE-08]
L
NL
NR
R
Offer
Answer
T=0
D
D
Resp D
RTP
D
Resp D
RTP
T=0
T=50
A
A
T=50
T=100
B
B
Resp B
T=100
T=150
C
Resp C
C
A
A
T=150
A and C run until retry limit reached
A and B run until retry limit reached


[Elim-Dups]
L
NL
NR
R
Offer
Answer
T=0
C
B
Resp B
RTP
C
Resp C
RTP
T=0
T=50
A
A
T=50
A runs until retry limit reached
A runs until retry limit reached

Elapsed time	[ICE-08] Processing	[Elim-Dups] Processing
T = 0	R begins by sending a Binding Request for check D, which installs a filtering rule towards L_2 in R's NAT, but is dropped by L's NAT. Shortly afterwards, L sends a Binding Request for check D, which makes it to R. When the response arrives back at L, L's state machine goes into the Recv-Valid state and can start sending media. The receipt of a Binding Request for check D causes R to resend its own STUN Request for D, which makes it through L's NAT this time. When the response arrives back at R, R can also start sending media.	R begins by sending a Binding Request for check C (which is equivalent to check D from R's perspective). As in [ICE-08], this installs a filtering rule towards L_2 in R's NAT, but is dropped by L's NAT. Shortly afterwards, L sends a Binding Request for check B (which is equivalent to check D from L's perspective). This makes it to R, which replies. When the response arrives back at L, L's Tx state machine goes Valid and thus L can start sending media. The receipt of a Binding Request for check B causes R to resend the Binding Request for check C, since the source and destination transport addresses in the received Binding Request for B (when swapped) match check C. When the response for C arrives back at R, R can also start sending media.
T = 50	R and L both try check A, which fails because the respective destination addresses are private.	R and L both try check A, which fails because the respective destination addresses are private. At this point, all checks have been tried once. Since there is no re-offer, check A will continue to run until it reaches it retry limit.
T = 100	R tries check B, which fails. L then tries check B, which succeeds in the L→R direction.	
T = 150	L and R try check C, which succeeds in the L←R direction, but fails in the L→R direction. Both L and R also retry check A. At this point, all checks have been tried once. Since there is no re-offer, checks A and C will continue to run until they reach their retry limits.	

Using [ICE-08], L sends a total of 22 messages and R sends a total of 23 messages, giving 45 messages in all.
Using [Elim-Dups], L sends a total of 11 messages and R sends a total of 12 messages, giving 23 messages in all.
Thus [Elim-Dups] has only 51% of the messages of [ICE-08] in this example.
Both procedures discover a working path at approximately the same time.