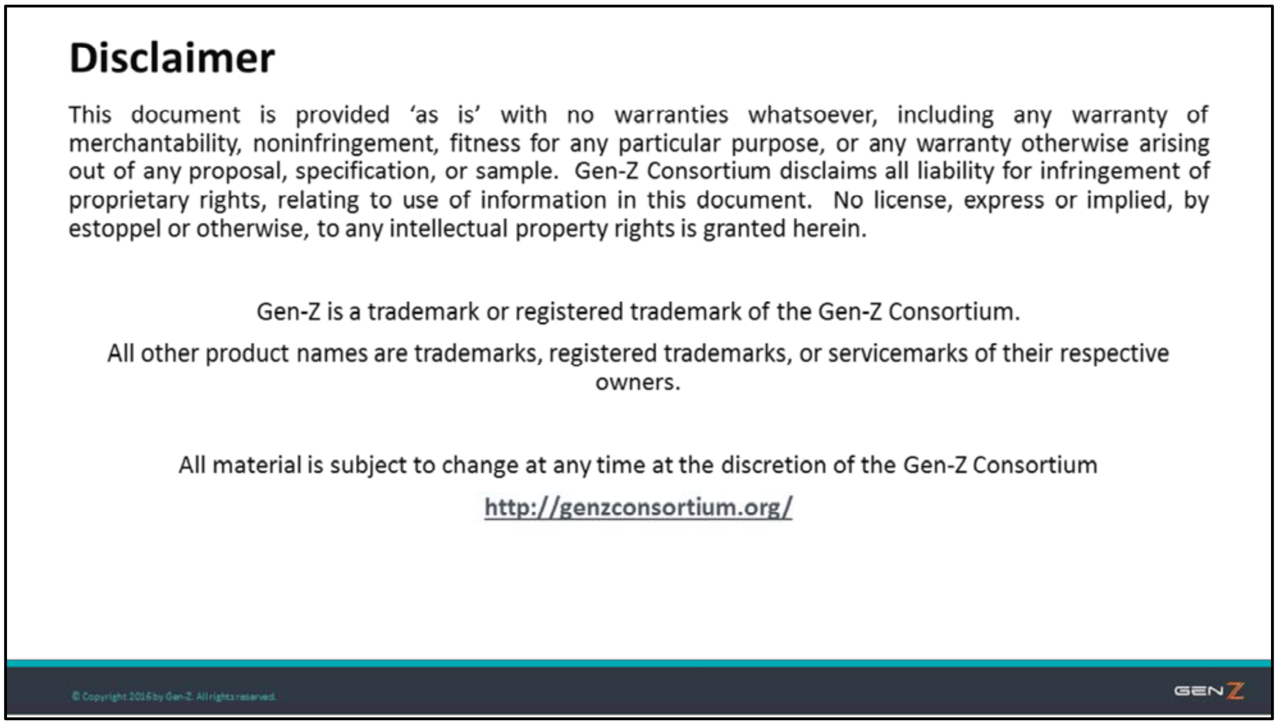
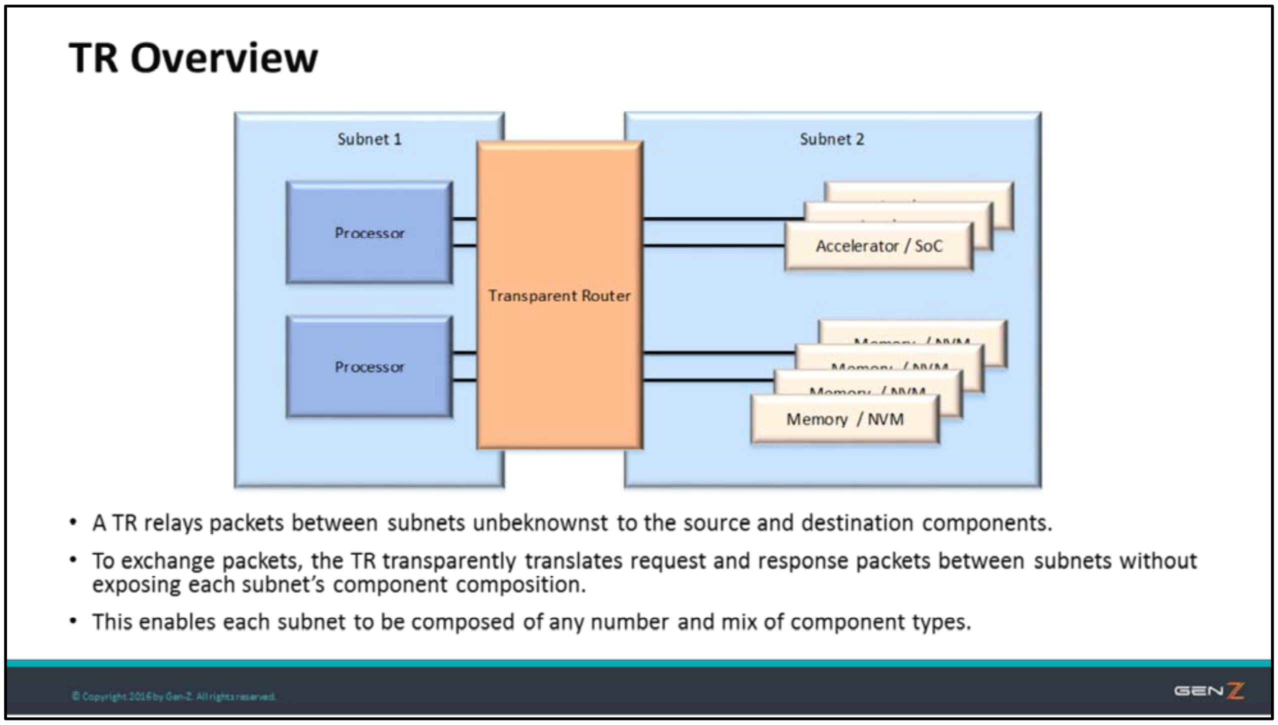
Gen-Z Transparent Routers (TRs)

July 2017

This presentation covers Transparent Routers.   A TR is used to transparently join two or more subnets together without the Requester or Responders being aware that their packets are crossing subnet boundaries.



Disclaimer



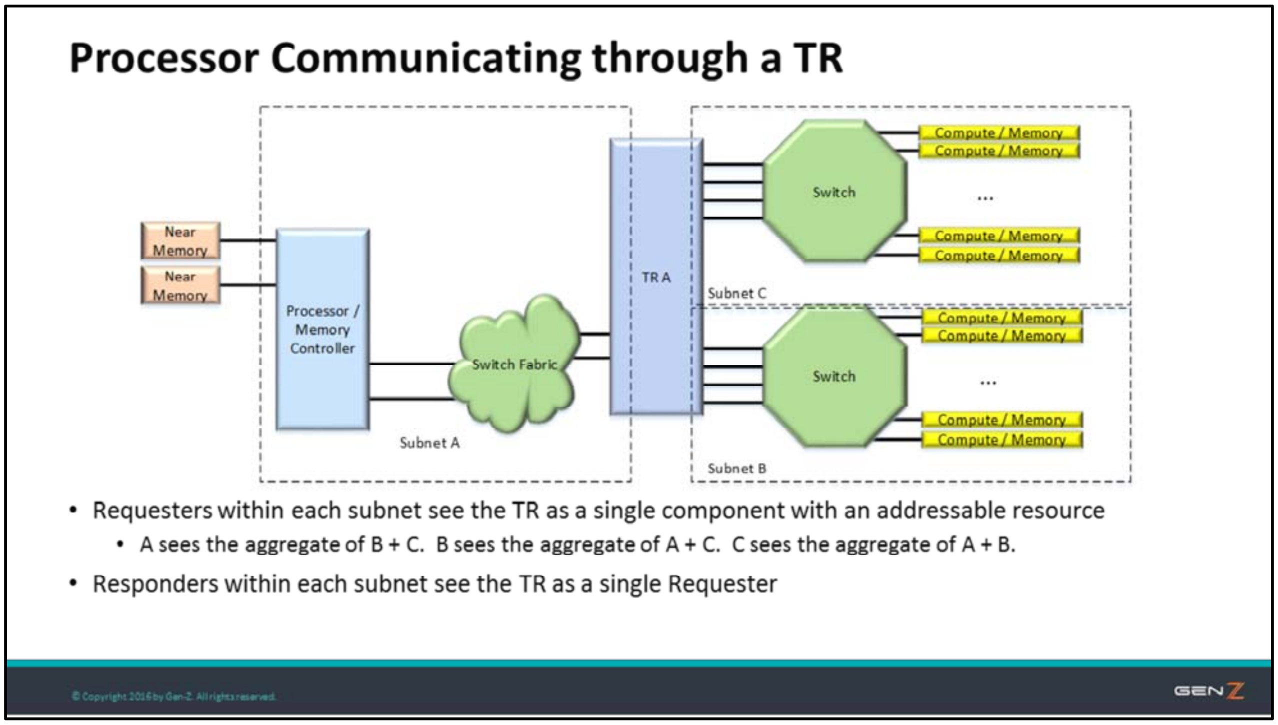
TR Overview

- A TR relays packets between subnets unbeknownst to the source and destination components.
- To exchange packets, the TR transparently translates request and response packets between subnets without exposing each subnet's component composition.
- This enables each subnet to be composed of any number and mix of component types.

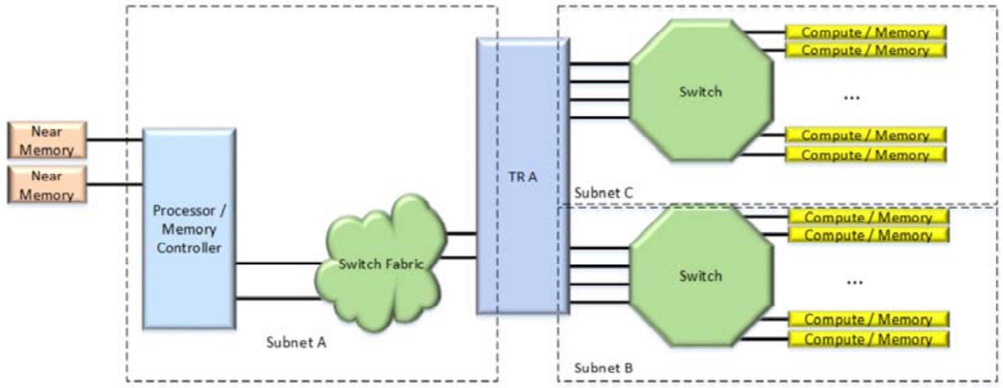
A TR enables:

- Isolation—events and services that occur behind a TR are invisible to external processors, operating systems, application software, management, etc. For example, a subnet may provide memory resiliency or transparently migrate memory between tiers to optimize performance without processor or application coordination.
- Serviceability—components within a subnet can be mechanically serviced conceptually similar to how a storage controller enables underlying storage targets to be transparently serviced / replaced.
- Differentiated services—multiple services can be transparently instantiated or executed or performance accelerated on all or a subset of a multi-component subnet's resources.
- Power optimization—components within a subnet can be transparently power optimized to meet environmental and dynamic workload needs.
- A TR may be integrated into any component type including processors, e.g., to take advantage of a TR-optimized MMU and reduce solution component count.
- A TR proxies request and response packets between subnets without revealing the actual source and destination components within each subnet.

- A TR may be used to transparently translate packets from one OpClass to another.
- A TR may be used to transparently perform memory interleaving on behalf of one or more Requesters.

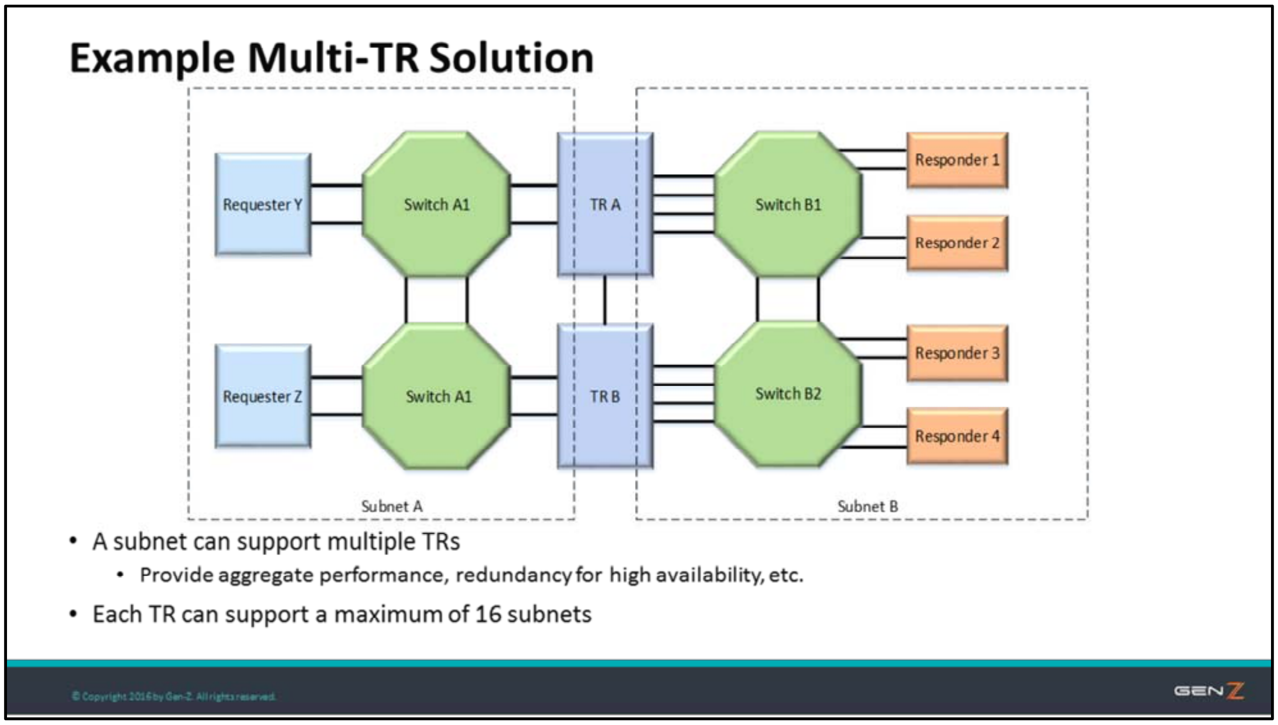


Processor Communicating through a TR

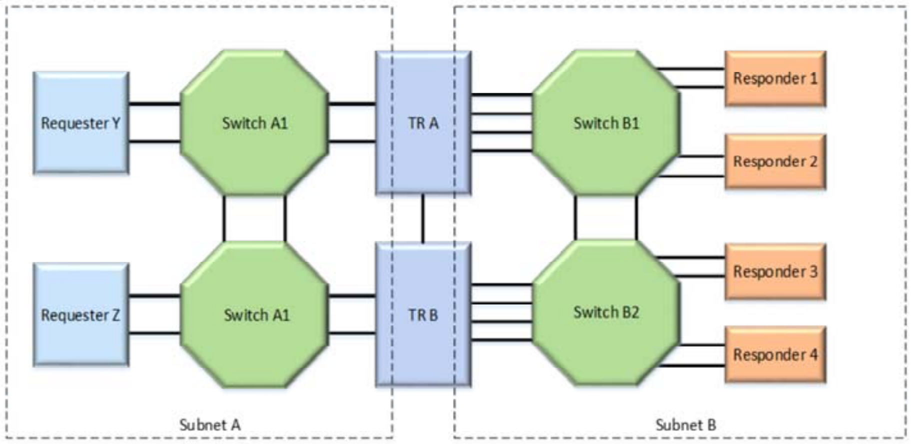


- Requesters within each subnet see the TR as a single component with an addressable resource
  - A sees the aggregate of B + C. B sees the aggregate of A + C. C sees the aggregate of A + B.
- Responders within each subnet see the TR as a single Requester

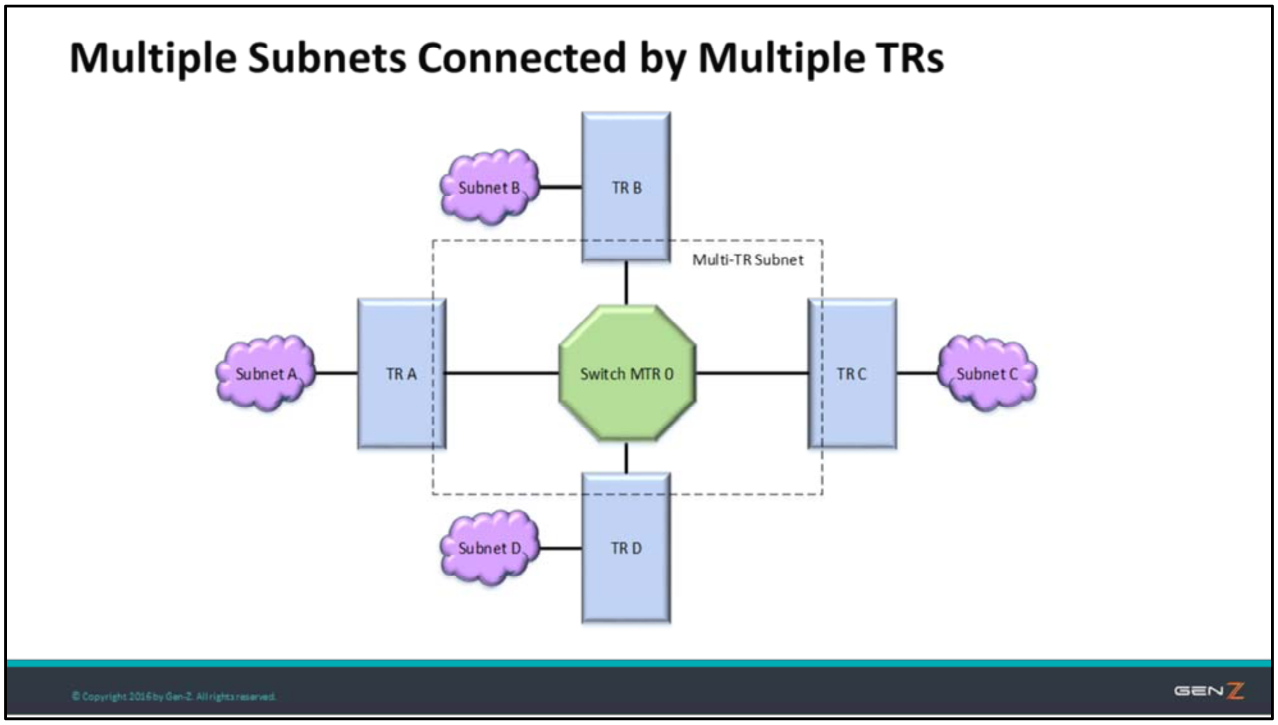
The Data Space presented to each directly-attached subnet may be mapped to multiple subnets. For example, this illustrates three subnets. For each subnet, the TR advertises a unique Data Space, e.g., subnet A sees a single Data Space that enables transparent access to subnets B and C, subnet B sees a single Data Space that enables transparent access to subnets A and C, and subnet C sees a single Data space that enables transparent access to subnets A and B.



Example Multi-TR Solution

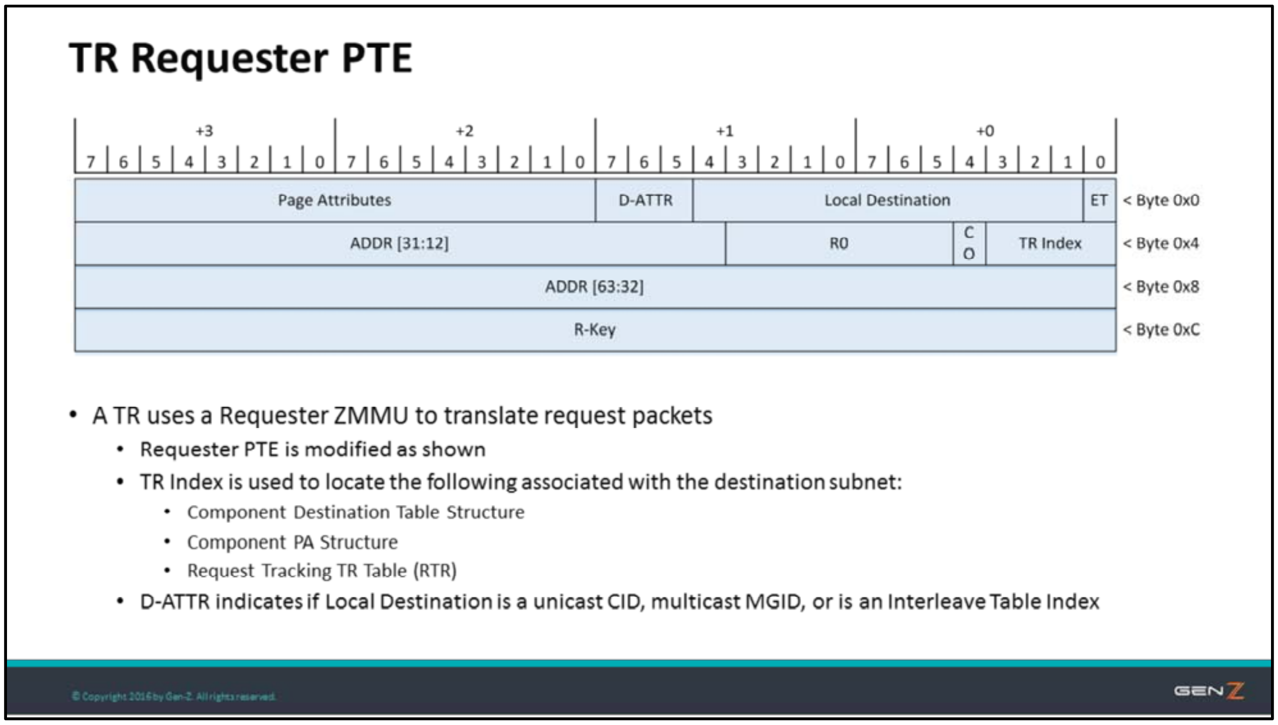


- A subnet can support multiple TRs
  - Provide aggregate performance, redundancy for high availability, etc.
- Each TR can support a maximum of 16 subnets



Multiple Subnets Connected by Multiple TRs

To provide additional scale-out, multiple TRs can be connected within a separate TR subnet.



TR Requester PTE

+3 bits 7–0	+2 bits 7–0	+1 bits 7–0	+0 bits 7–0	
Page Attributes (+3 7–0, +2 7–0)		D-ATTR (+1 7–5); Local Destination (+1 4–0, +0 7–1); ET (+0 0)		< Byte 0x0
ADDR [31:12] (+3 7–0, +2 7–0, +1 7–4)		R0 (+1 3–0, +0 7–5); CO (+0 4)	TR Index (+0 3–0)	< Byte 0x4
ADDR [63:32]				< Byte 0x8
R-Key				< Byte 0xC

- A TR uses a Requester ZMMU to translate request packets
  - Requester PTE is modified as shown
  - TR Index is used to locate the following associated with the destination subnet:
    - Component Destination Table Structure
    - Component PA Structure
    - Request Tracking TR Table (RTR)
  - D-ATTR indicates if Local Destination is a unicast CID, multicast MGID, or is an Interleave Table Index

TR

Example Request-Response Flow Through a TR

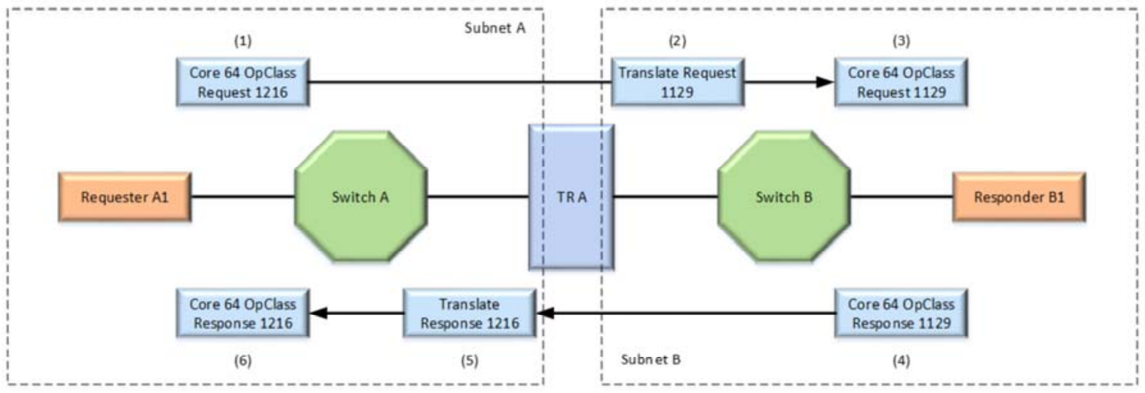


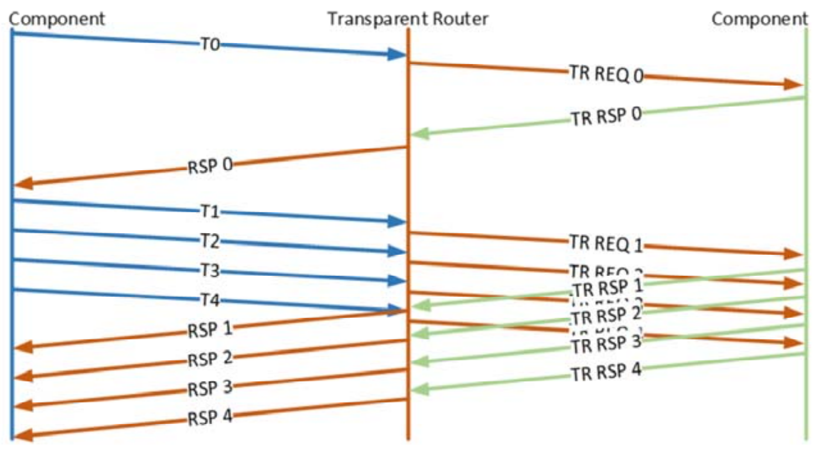
Figure illustrates two subnets—Subnet A and Subnet B connected by a single TR A. The following steps are taken to transmit request and response packets between subnet A and subnet B:

Requester A1 transmits (1) request 1216 to TR A.

1. TR A generates a local RTR and translates (2) the request packet.
    1. The TR uses the Requester PTE TR Index to locate the destination subnet's *Component Destination Table Structure.* The Local Destination field is used to index the Component Destination Table structure to identify the potential egress interfaces and the associated VC.
    2. The TR uses the Requester PTE TR Index to locate the destination subnet's *Component PA Structure.* The Local Destination field is used to index the supported Component PA structure's tables to translate specific request packet protocol fields, e.g., the Access Key, Next Header, etc., and to take applicable security actions.
    3. The TR uses the Requester PTE TR Index to locate the destination subnet's RTR, and to allocate and populate an RTR table entry to reflect the translated request packet.

2. TR A transmits (3) the translated request packet to Responder B1.
3. Responder B1 executes the request, and transmits (4) the response packet to TR A.
4. TR A identifies the RTR, and translates (5) the response packet.
    1. The TR uses the TR Index in ingress interface’s *Interface Structure* to locate the RTR associated with the source subnet.
    2. The TR uses response packet protocol fields, e.g., [SCID, Tag], to locate the RTR table entry and translate the response packet.
5. TR A transmits (6) the response packet to Requester A1.

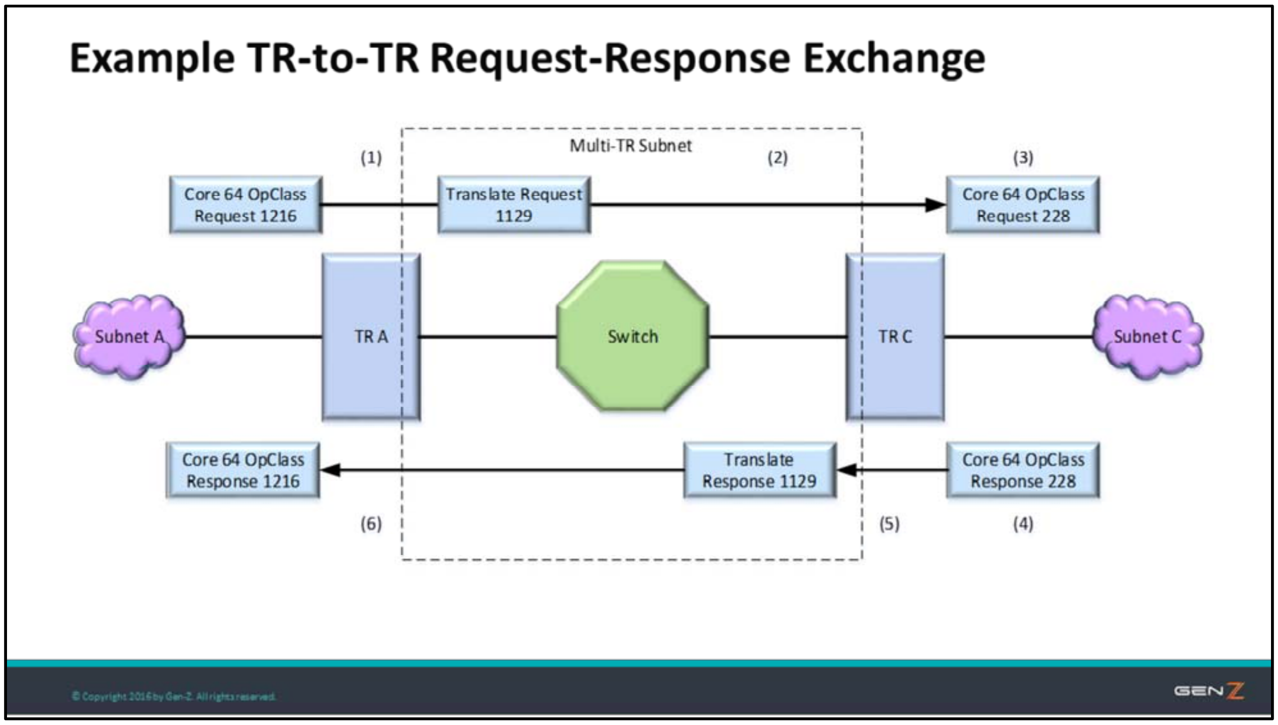
Series of Packets: Requester to TR to Responder



GEN Z

Figure illustrates how a TR translates a series of request and response packets. The term “destination subnet” refers to the subnet of the destination component associated with a given packet.

1. Request T0 is translated into destination subnet TR REQ 0.
2. TR RSP 0 is translated into destination subnet response RSP 0.
3. Request packets T1-T4 are translated into destination subnet request packets TR REQ 1-TR REQ4.
4. TR RSP 1-TR RSP4 are translated into destination subnet response packets RSP 1-RSP 4.



Example TR-to-TR Request-Response Exchange

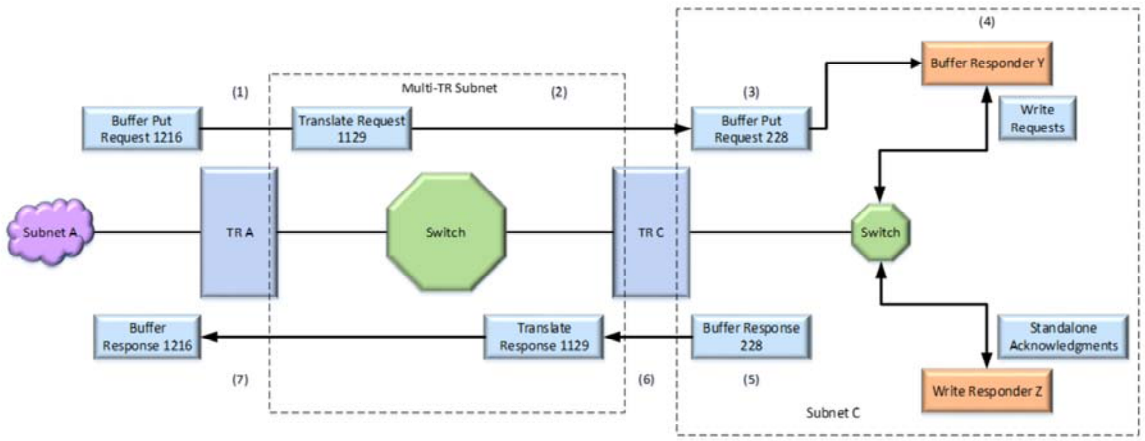
Multi-TR Subnet
(1)
(2)
(3)
Core 64 OpClass Request 1216
Translate Request 1129
Core 64 OpClass Request 228
Subnet A
TR A
Switch
TR C
Subnet C
Core 64 OpClass Response 1216
Translate Response 1129
Core 64 OpClass Response 228
(6)
(5)
(4)

Figure illustrates three subnets—Subnet A, the intervening Multi-TR subnet, and Subnet C. The following steps are taken to transmit request and response packets between subnet A and subnet C:

1. TR A's Requester ZMUU is configured such that TR A is able to determine that the request packet is destined for a Responder in subnet C.
2. TR A receives request 1216 for the Requester. To forward the packet across the multi-TR subnet:
    1. TR A generates a local RTR, translates (1) the request packet, and transmits (2) the packet to TR C.
    2. TR C generates a local RTR, translates (3) the request packet into a subnet C-local request packet, and transmits the packet to the Responder.
    3. The Responder executes the request packet, and transmits a response packet to TR C.
    4. TR C receives the corresponding response from the subnet-C Responder, and translates (4) the packet into the TR A response.
    5. TR C translates the response packet (5), transmits the packet to TR A, and releases the local RTR.

6. TR A receives the response packet, translates (6) the packet, transmits the response to the original Requester, and releases the local RTR.

Example TR-to-TR Buffer Op



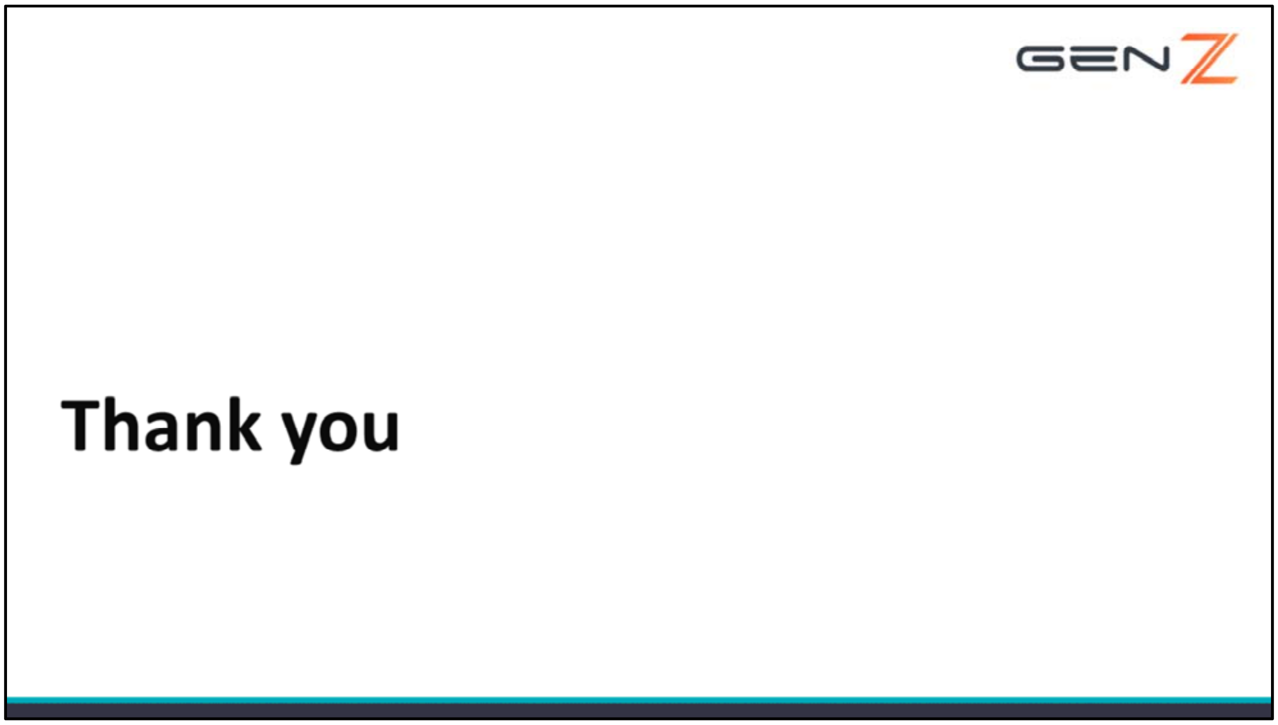
GEN Z

Figure illustrates a remote Buffer Put operation where the Requester believes it is migrating data within a single Responder component.
TR A receives request 1216 for the Requester. To forward the packet across the multi-TR subnet:

1. TR A generates a local RTR. In this particular case, TR A needs to examine the two addresses (A and B) within the Buffer Put request to confirm both map to subnet C.
2. TR A translates (1) the request packet (in general, only the fields required to traverse the multi-TR subnet and the Tag field need to be translated), and transmits (2) the request packet to TR C.
3. TR C generates a local RTR, translates (3) the request packet into a subnet C-local Buffer Put request packet, and transmits the packet to the Buffer Responder Y.
4. The Buffer Responder Y executes the Buffer Put request packet, and generates (4) a series of subnet C-local Write request packets to move buffer A in Responder Y to buffer B in the Responder Z.
5. Once Buffer Responder Y successfully executes the Buffer Put request packet, it generates and transmits (5) a *Standalone Acknowledgment* to

TR C.

6. TR C translates the *Standalone Acknowledgment* (6), transmits the packet to TR A, and releases the local RTR.
7. TR A receives the *Standalone Acknowledgment*, translates (7) the packet, transmits the *Standalone Acknowledgment* to the original Requester, and releases the local RTR.



Thank you

This concludes this presentation. Thank you.