Framework for IPv4/IPv6 Multicast Translation

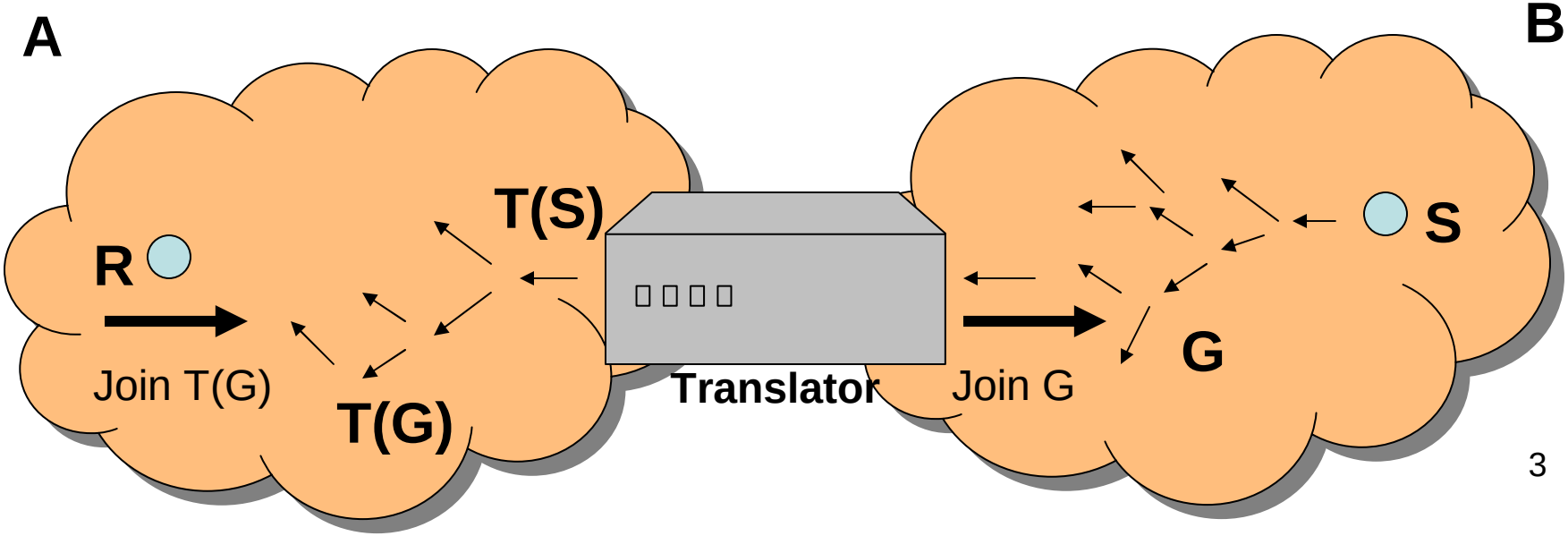
draft-venaas-behave-v4v6mc-framework-00.txt

Overview

- The draft considers the 6 scenarios in the behave wg charter
- It discusses the general problems and some possible ways the might be addressed
- It tries to conclude with some of the necessary components needed
- The draft needs more work to really be a framework

Unicast versus Multicast

- The charter has 6 scenarios talking about “network **A** to network **B”** where a host **R** in **A** initiates a uni/bi-directional flow to a host **S** in **B**
- For multicast the equivalent is “**A** receiving from **B**” where a host **R** in **A** receives a group **G** (in **B**) with a host **S** in **B** as a source
- In both cases, **R** is the initiator and needs to know the translated address(es) **T(S)** and/or **T(G)**



Scenarios

1. **An IPv6 network receiving multicast from IPv4 Internet**
2. IPv6 Internet receiving multicast from an IPv4 network
3. **An IPv4 network receiving multicast from IPv6 Internet**
4. IPv4 Internet receiving multicast from an IPv6 network
5. An IPv6 network receiving multicast from an IPv4 network
6. An IPv4 network receiving multicast from an IPv6 network

Scenario 1 – An IPv6 network receiving multicast from IPv4 Internet

- Not so hard since IPv4 address space can be embedded into IPv6
  - E.g. T(224.1.2.3) = ff1e::ffff:224.1.2.3
  - May need to accommodate for SSM and scopes
  - T(232.1.2.3) = ff3e::ffff:232.1.2.3
  - T(239.1.2.3) = ff35::ffff:239.1.2.3
- R wants to receive G and joins T(G)
- How does R know T(G)?
  - E.g. SDP data may be translated by an ALG
  - Or application or stack on R is translation aware and knows/learns T()
  - Standardised T() (well-known multicast prefixes) or configuration?

Well-known multicast prefix(es) for 4->6?

- Well-known multicast prefixes could be hardcoded in apps/stacks so that they know T() and join translated groups when needed
- Well-known may be useful for e.g. IPv6 Internet receiving from IPv4
  - All receivers joining the same tree
- If not well-known, there can be different prefixes for different translators
  - May choose which translator is used
- For trees to pass through the translator, it may need to be an IPv6 Rendezvous Point. In that case embedded-RP might be useful
  - Embedded-RP encodes the unicast address of the RP in the group address. Hence well-known multicast prefix is hard, unless also well-known unicast address (anycast)

Scenario 3 – An IPv4 network receiving multicast from IPv6 Internet

- We cannot use a simple embedding and stateless translation
- How does translator get a mapping so it can translate an IPv4-join into IPv6?
  - And translate data from IPv6 to IPv4.
- Might use some ALG to translate e.g. SDP as it passes through the translator
  - This is hard
- Or new signaling mechanisms between receiver and translator
- An administrator might add static mappings and inform users, or create new sdp, with groups to use

New signaling mechanisms

- We may need new signaling mechanisms for an IPv4 host or network to be able to receive arbitrary IPv6 groups
- A translation aware application or stack could send a query to the translator saying:
  - I want to receive G, which T(G) should I join?
  - The translator can have a pool of IPv4 addresses and allocate them as needed