LISP Predictive-RLOCs
Mobility with Near-Zero Packet Loss

draft-farinacci-lisp-predictive-rlocs-00

LISP Working Group - Berlin IETF
July 2016

Dino Farinacci and Padma Pillay-Esnault

Problem Statement

- The mobility problem is simple ;-)
  - When an EID moves, you send packets to the new location
- **NOT** ;-)
  - Packets already in the network are going to the old location (where the EID is no longer)
  - EID has arrived at the new location but it is not receiving packets (sender doesn't know about the move yet)
- This is not "make-before-break"

Struggling Solutions

- Mobile-IP
  - You can't send to home agent because it doesn't know where the new location is
- Host Routes
  - They point to the old and the new location at the same time in different parts of the network
  - Handoffs are slow because the EID host route has to go everywhere
- Locator/ID Separation
  - A good solution if signaling is fast - sender gets new location quickly

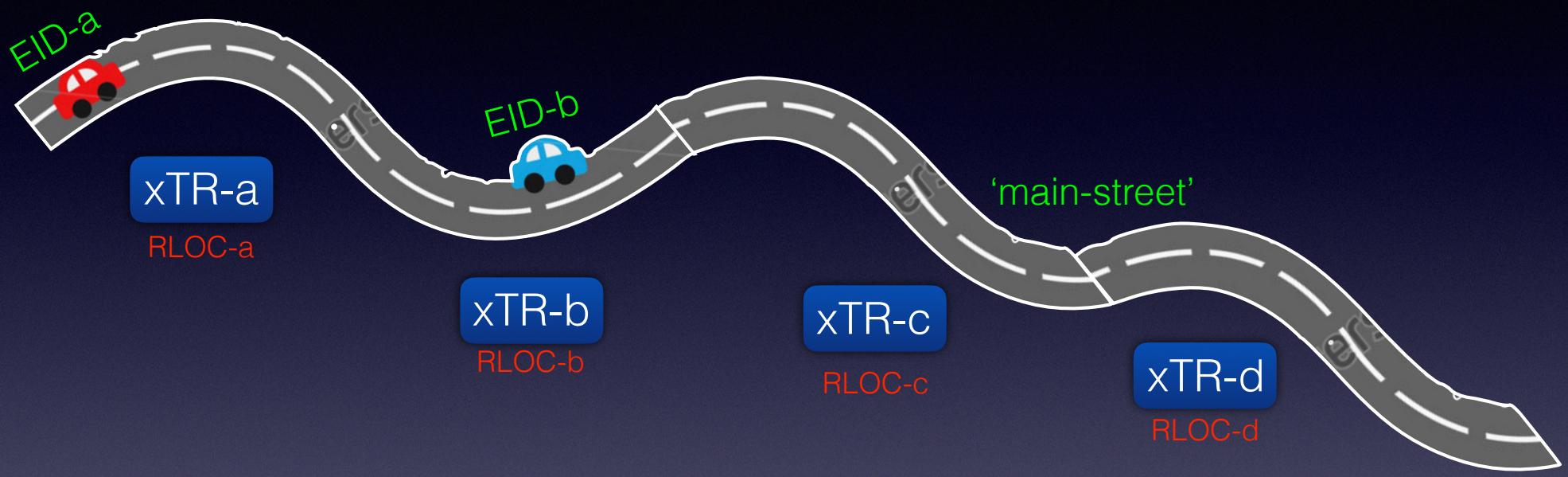
Near-Zero Packet Loss

- We really don’t want to drop **any** packets
- We want handoffs to be instantaneous (atomic)

The Future is Clear

- What if we know all new locations?
- Have source send to all new locations
- We'll search (and find) where the EID has roamed to
- Exercising a bandwidth/signaling tradeoff

Predictive RLOCs



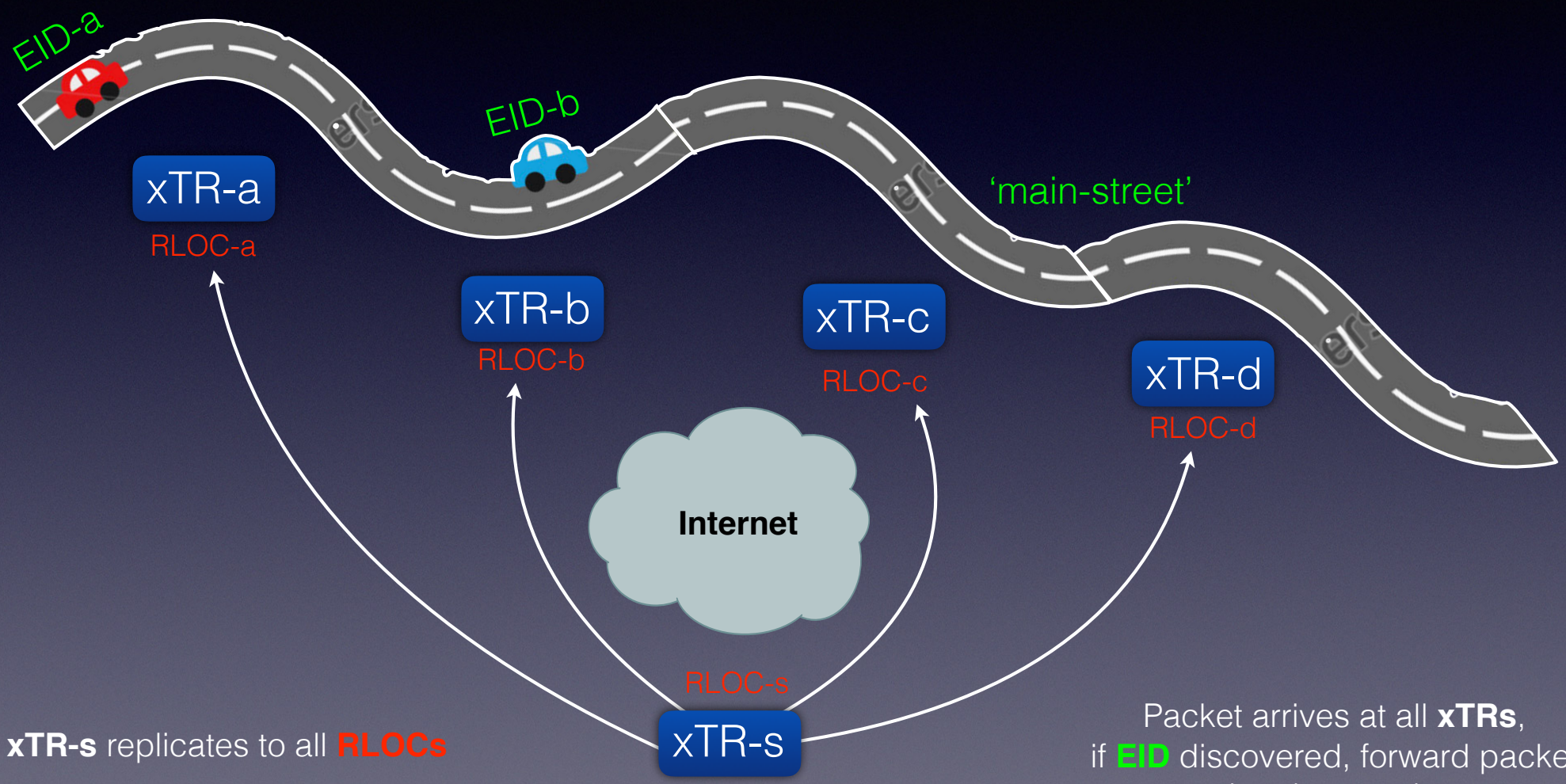
LISP Mapping Database

'main-street' ->
RLE: RLOC-a, RLOC-b, RLOC-c, RLOC-d

EID-a ->
RLOC: 'main-street'

EID-b ->
RLOC: 'main-street'

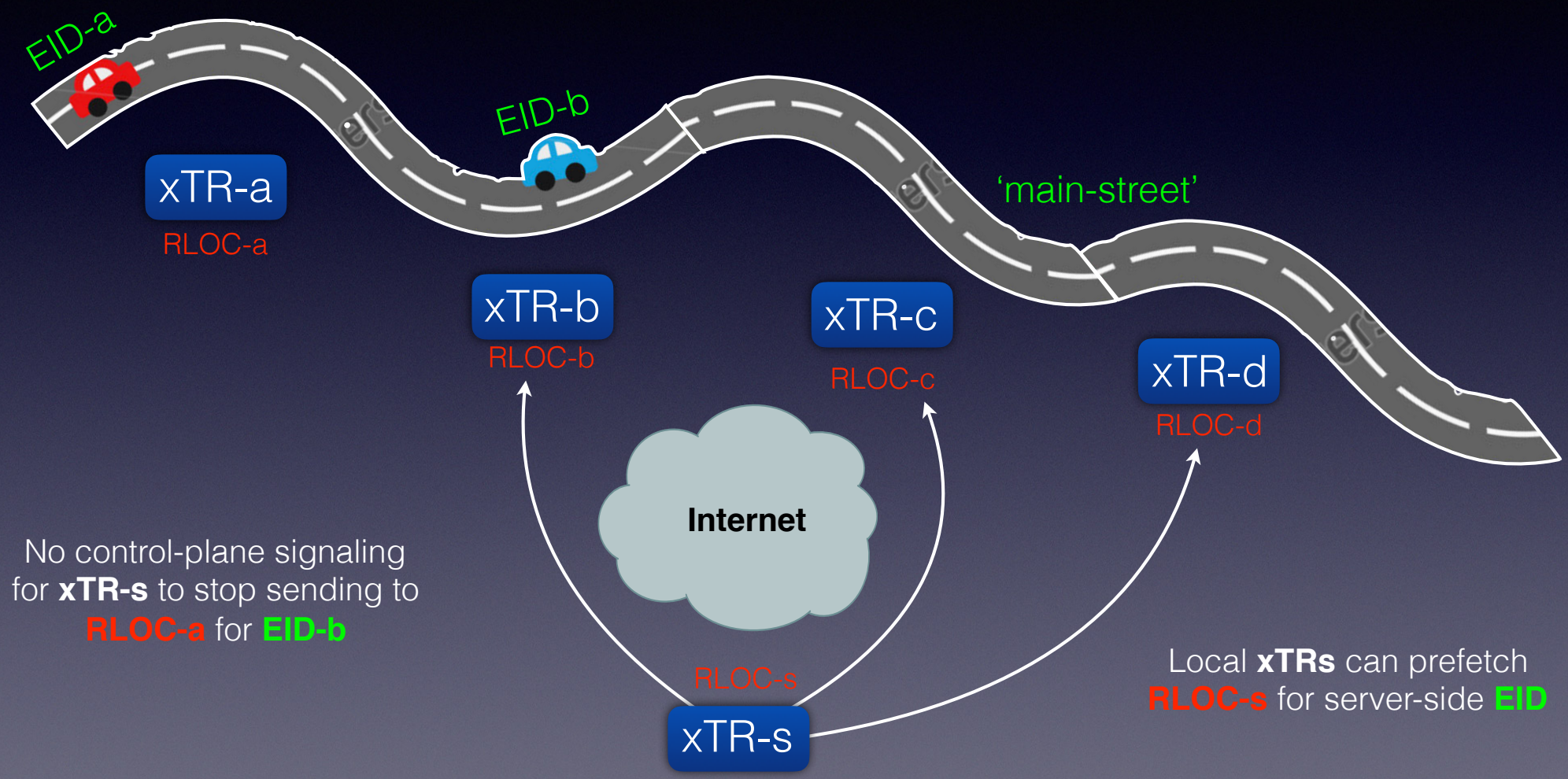
Predictive RLOCs



xTR-s replicates to all **RLOCs**

Packet arrives at all **xTRs**,
if **EID** discovered, forward packet
else drop packet

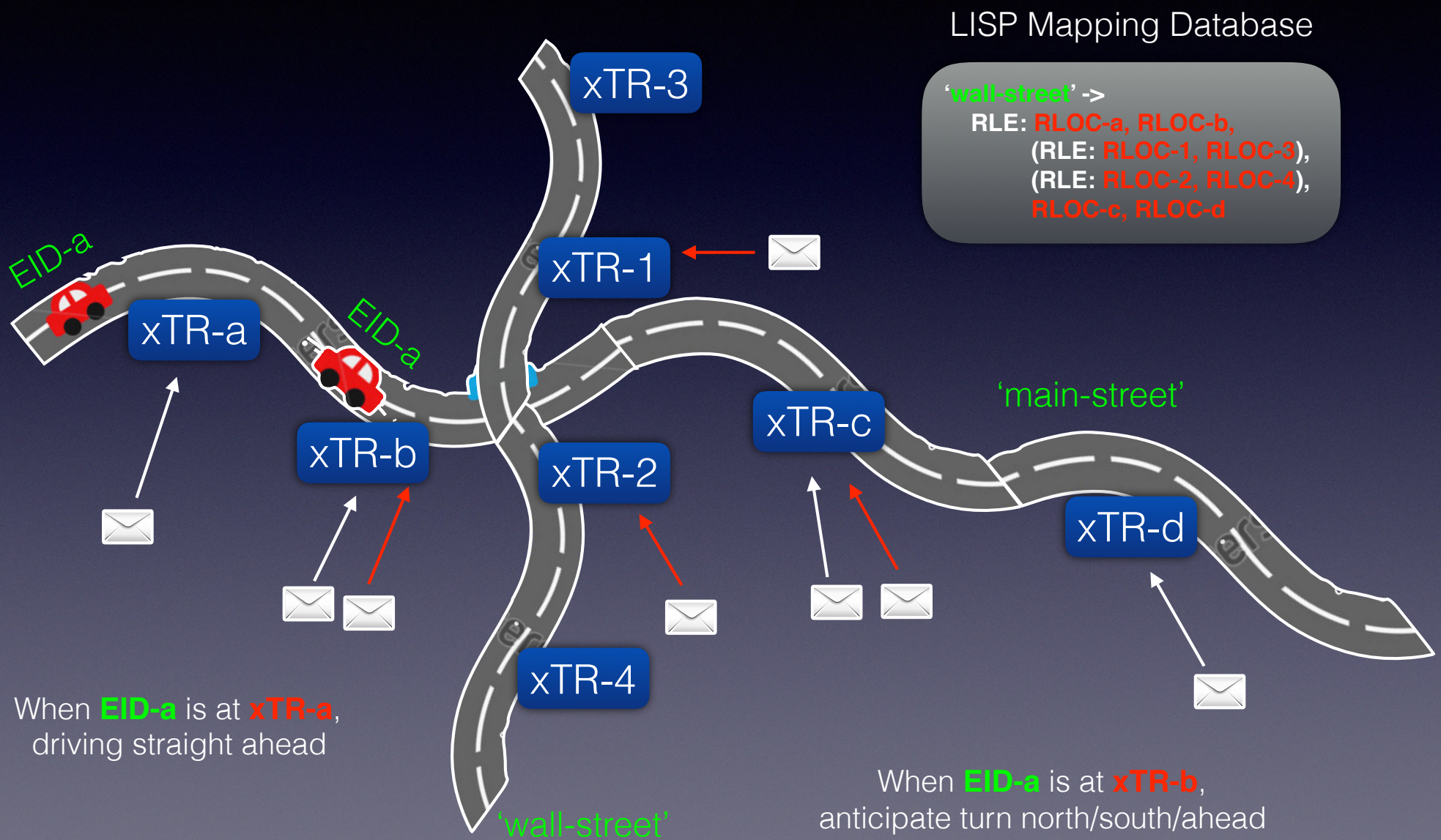
Predictive RLOCs



No control-plane signaling for **xTR-s** to stop sending to **RLOC-a** for **EID-b**

Local **xTRs** can prefetch **RLOC-s** for server-side **EID**

Intersections

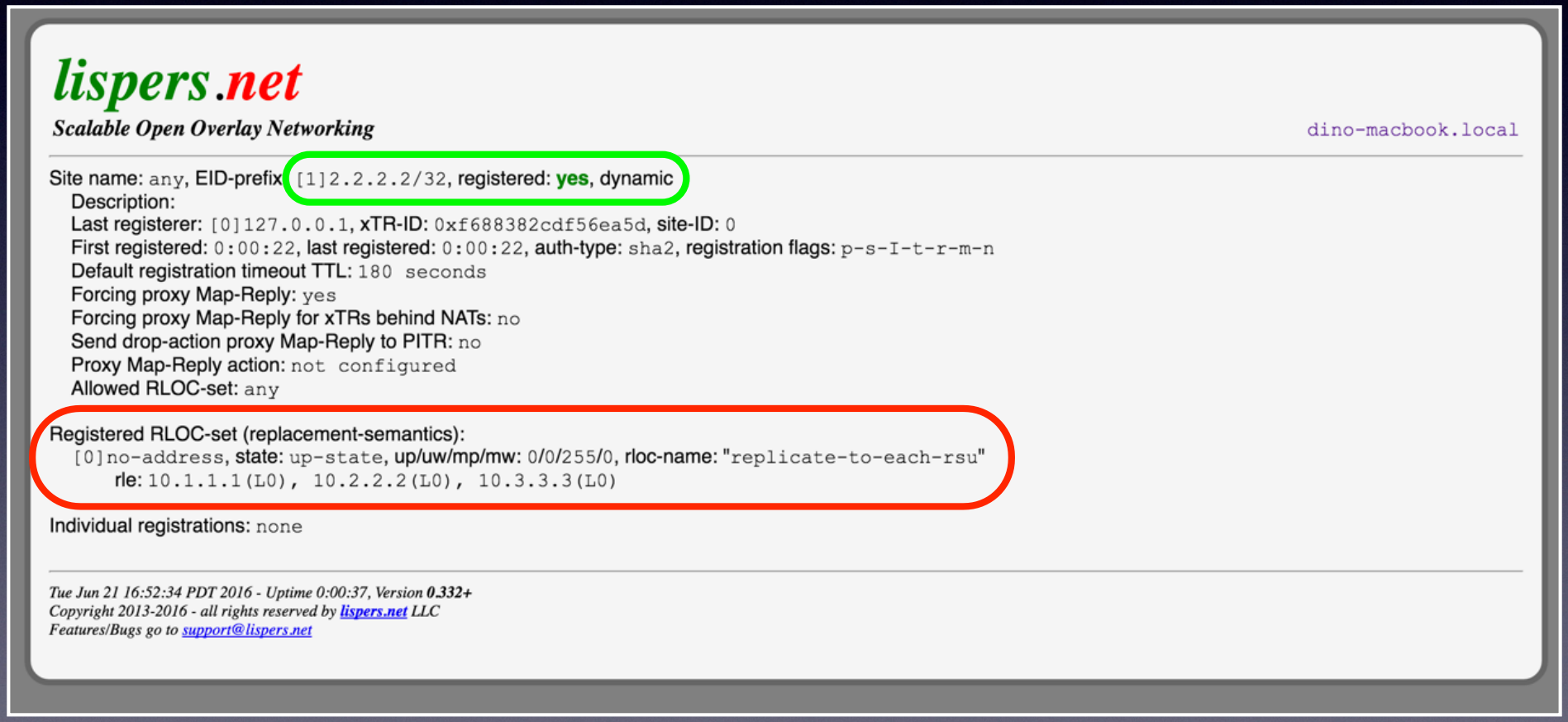


LISP Protocol Changes

- None
- Use RLE LCAFs for unicast map-cache entries
- By the way, multicast just works
  - When roaming EID is a receiver
  - When roaming EID is a source, (S-EID, G) cannot be pre-fetched

Quick Demo

Any road-side-unit xTR discovers a roaming EID **[1]2.2.2.2** . . .



lispers.net

Scalable Open Overlay Networking

dino-macbook.local

Site name: any, EID-prefix [1]2.2.2.2/32, registered: **yes**, dynamic
Description:
Last registerer: [0]127.0.0.1, xTR-ID: 0xf688382cdf56ea5d, site-ID: 0
First registered: 0:00:22, last registered: 0:00:22, auth-type: sha2, registration flags: p-s-I-t-r-m-n
Default registration timeout TTL: 180 seconds
Forcing proxy Map-Reply: yes
Forcing proxy Map-Reply for xTRs behind NATs: no
Send drop-action proxy Map-Reply to PITR: no
Proxy Map-Reply action: not configured
Allowed RLOC-set: any

Registered RLOC-set (replacement-semantics):
[0]no-address, state: up-state, up/uw/mp/mw: 0/0/255/0, rloc-name: "replicate-to-each-rsu"
rle: 10.1.1.1(L0), 10.2.2.2(L0), 10.3.3.3(L0)

Individual registrations: none

*Tue Jun 21 16:52:34 PDT 2016 - Uptime 0:00:37, Version **0.332+***
Copyright 2013-2016 - all rights reserved by lispers.net LLC
Features/Bugs go to support@lispers.net

. . . the RSU or a controller could register the predictive-RLOC mapping

Quick Demo

ITR has EID **[1]2.2.2.2** in its map-cache . . .

```
06/21/16 16:08:50.742: itr: Receive en0, MACs: a45e-60ed-e363 -> 0023-688d-8500, [1]1.1.1.1 -> [1]2.2.2.2, tos/ttl: 0/64, length: 84, packet: 45000054 541f0000 40012085 01010101 0202020
2 080022a1 022f008a 5769c902 0009c72d 08090a0b 0c0d0e0f 10111213 14151617 18191a1b 1c1d1e1f       ...
06/21/16 16:08:50.742: itr: Lookup for EID [1]2.2.2.2 found map-cache entry [1]2.2.2.2/32
06/21/16 16:08:50.742: itr: Packet hash is 0, best-rloc-list:
06/21/16 16:08:50.742: itr: Replicate-to-L0 LISP packet, outer RLOCs: [1]192.168.3.201 -> [0]10.1.1.1, outer tos/ttl: 0/63, outer UDP: 61440 -> 4341, inner EIDs: [1]1.1.1.1 -> [1]2.2.2.
2, inner tos/ttl: 0/63, length: 120, Replicate-to-L0 LISP-header -> flags: NlevIpk, nonce: f8e4e7, iid/lsb: 100, packet: 45000078 dfdf4000 3f118c22 c0a803c9 0a010101 f00010f5 00640000 8
4f8e4e7 00000100 45000054 541f0000 3f012185 01010101 02020202       ...
06/21/16 16:08:50.743: itr: Replicate-to-L0 LISP packet, outer RLOCs: [1]192.168.3.201 -> [0]10.2.2.2, outer tos/ttl: 0/63, outer UDP: 61440 -> 4341, inner EIDs: [1]1.1.1.1 -> [1]2.2.2.
2, inner tos/ttl: 0/63, length: 120, Replicate-to-L0 LISP-header -> flags: NlevIpk, nonce: feedff, iid/lsb: 100, packet: 45000078 dfdf4000 3f118b20 c0a803c9 0a020202 f00010f5 00640000 8
4feedff 00000100 45000054 541f0000 3f012185 01010101 02020202       ...
06/21/16 16:08:50.743: itr: Replicate-to-L0 LISP packet, outer RLOCs: [1]192.168.3.201 -> [0]10.3.3.3, outer tos/ttl: 0/63, outer UDP: 61440 -> 4341, inner EIDs: [1]1.1.1.1 -> [1]2.2.2.
2, inner tos/ttl: 0/63, length: 120, Replicate-to-L0 LISP-header -> flags: NlevIpk, nonce: feedff, iid/lsb: 100, packet: 45000078 dfdf4000 3f118a1e c0a803c9 0a030303 f00010f5 00640000 8
4feedff 00000100 45000054 541f0000 3f012185 01010101 02020202       ...
```

. . . replicates to predictive-RLOCs **10.1.1.1**, **10.2.2.2**, and **10.3.3.3**

Work in Progress

- Use geo-prefixes to reduce replication scope for future RLOCs
- Use overlapping RLEs to reduce replication scope
- Use multiple RLOC-records with shorter RLEs to reduce replication scope
- Use RTRs close to ETR so replication is O(1) over RANs
- Use a level of indirection with distinguished-names for grouping roaming-EIDs to reduce predictive-RLOC duplication in different mappings
- LISP-crypto operation
  - Encrypt for each predictive-RLOC replication (like **draft-ietf-lisp-signal-free-multicast**)
  - Or encrypt once and replicate many (would have to share keys)

Questions/Comments/Tomatoes?