Topic: DNSSEC Ops
Problem: SEP provisioning

Edward Lewis

RIPE 59

Abstract

- An SEP is a DNSSEC public key that an administrator generates as part of the signing process
- An SEP is a DNSSEC public key that an an administrator receives as input, leading to DS records at a delegation
- There is no standard way to transfer the SEP despite many admin-admin environments

Why do we need a standard

- Today's ad-hoc situation isn't working
- The absence of a standard means the exchanges are informal
  - Informal does not scale
  - New players don't know where to start
  - Disenfranchised demographic stays that way
- Integrate as many players as possible, safely

A dilemma I live with

- A gTLD/ccTLD registry is expecting to rely on a EPP server as its provisioning ingress point
- A DNS managed service, not a registrar, does not operate a EPP client
- How do they talk to each other?
  - Even within the same organization?

Secure Entry Point (SEP)

- A Secure Entry Point is a key (KSK) that is intended to
  - Produce a DS record at the parent
  - Be configured in a Trust Anchor list
  - Be redistributed by a Trust Anchor Repository

Trust Anchor Repository

- TAR is a "security surrogate"
  - To a DNS administrator, it acts like the parent with respect to the SEP submission
  - To a DNS cache operator, it is a registry of security meta data (SEPs) with domain names
- A TAR is yet another form of a registry
  - Focus differs from a Domain Name Registry or RIR

SEP Lifecycle

- If an SEP was permanent we have no problem, but circumstances may require it be changed
- An SEP's "lifecycle" may include these stages
  - generation
  - preview (which might include emergency)
  - active
  - revoked (a la RFC 5011)
  - removed

Swapping an SEP

- One approach
  - Start with existing SEP, signed
  - Add new SEP to set, signed
  - Request a swap of DS records at parent or TAR
  - Confirm change, revoke (RFC 5011) the old
  - Remove the old SEP

Addendum

- There may be more than one SEP for a zone
  - For example, one per crypto-algorithm
  - For any operational reason
- The SEP change process presented here is just one model
  - This isn't an effort to pick one change process
  - The resulting provisioning process should accommodate many different change processes

The problem

- Middle step: *Request a swap of DS records at parent and/or TAR*
  - An external dependency
  - Few have specified how this will be done
    - There is RFC 4310 (EPP for DNSSEC) but that has limited scope
  - Test beds offer web pages; key scrapers pick
  - Building scripts for SEP change is not easy
- Needs to address: security, service level agreement

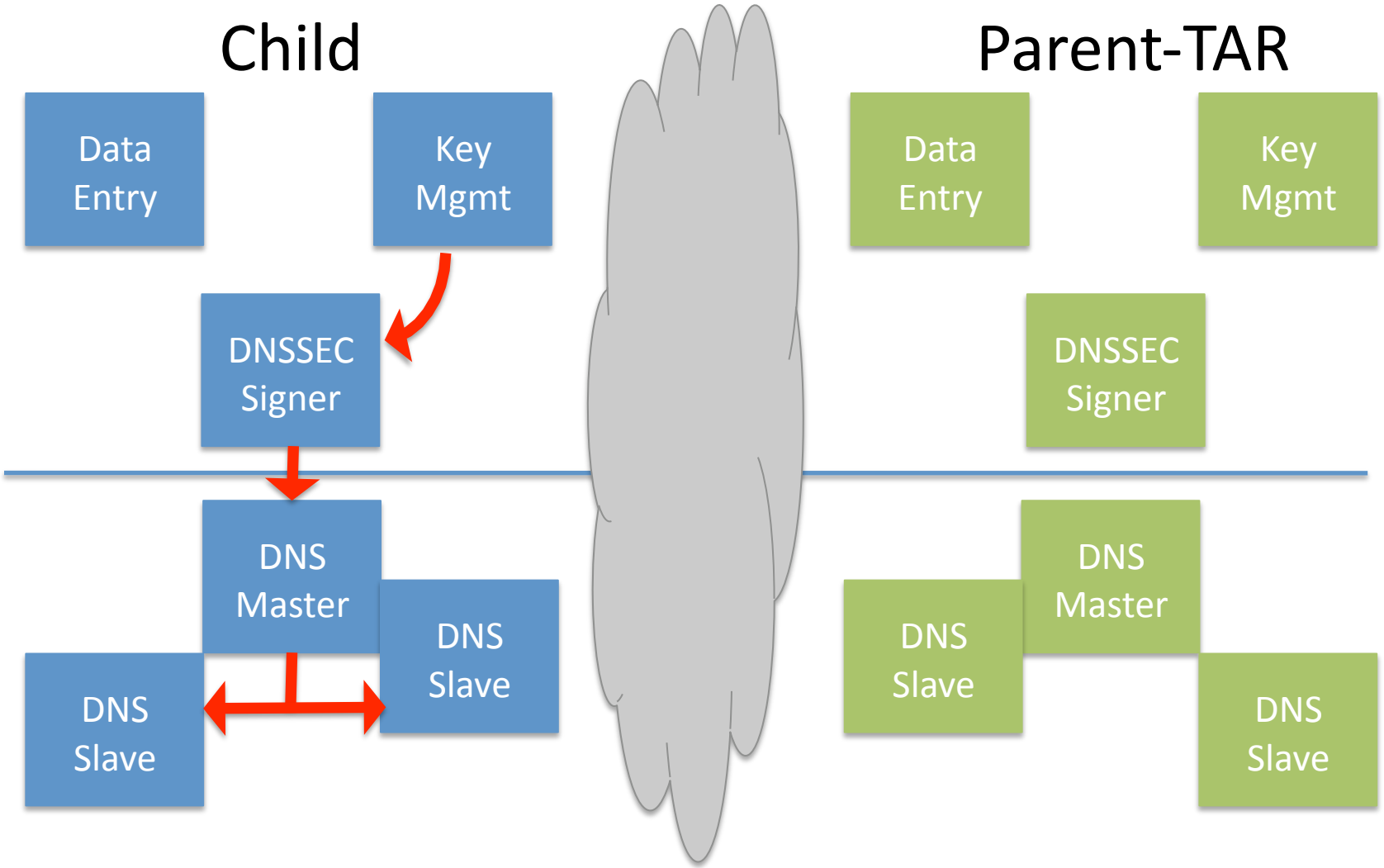
Why didn't RFC 5011 solve this?

- RFC 5011 "Automated Updates of DNSSEC Trust Anchors"
  - No mention of redistribution issues
  - No confirmation step (not needed because this wasn't meant for redistribution to other parties)
- Without confirmation, this doesn't provide the necessary feedback to the provisioning client

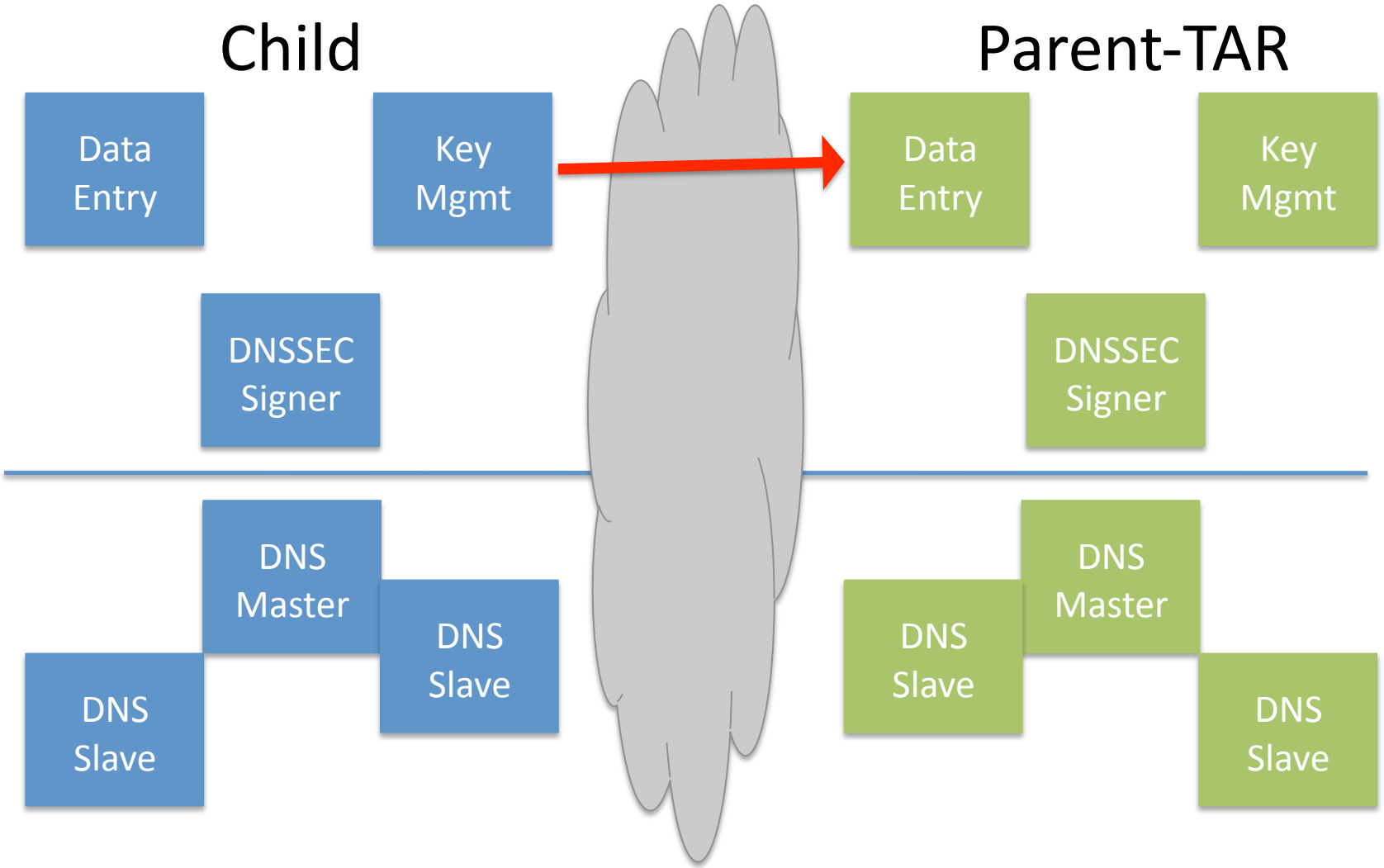
Visualizing the Problem

- The next five slides show these four steps
  - The child publishes a new SEP(-to-be)
  - The DS (new SEP) gets to the parent-TAR
  - Parent-TAR publishes the (Signed) DS
  - The child revokes the old SEP
- Hmm, before I said there were five steps
  - This focuses on step #2, #3, #4, dividing #3 in half

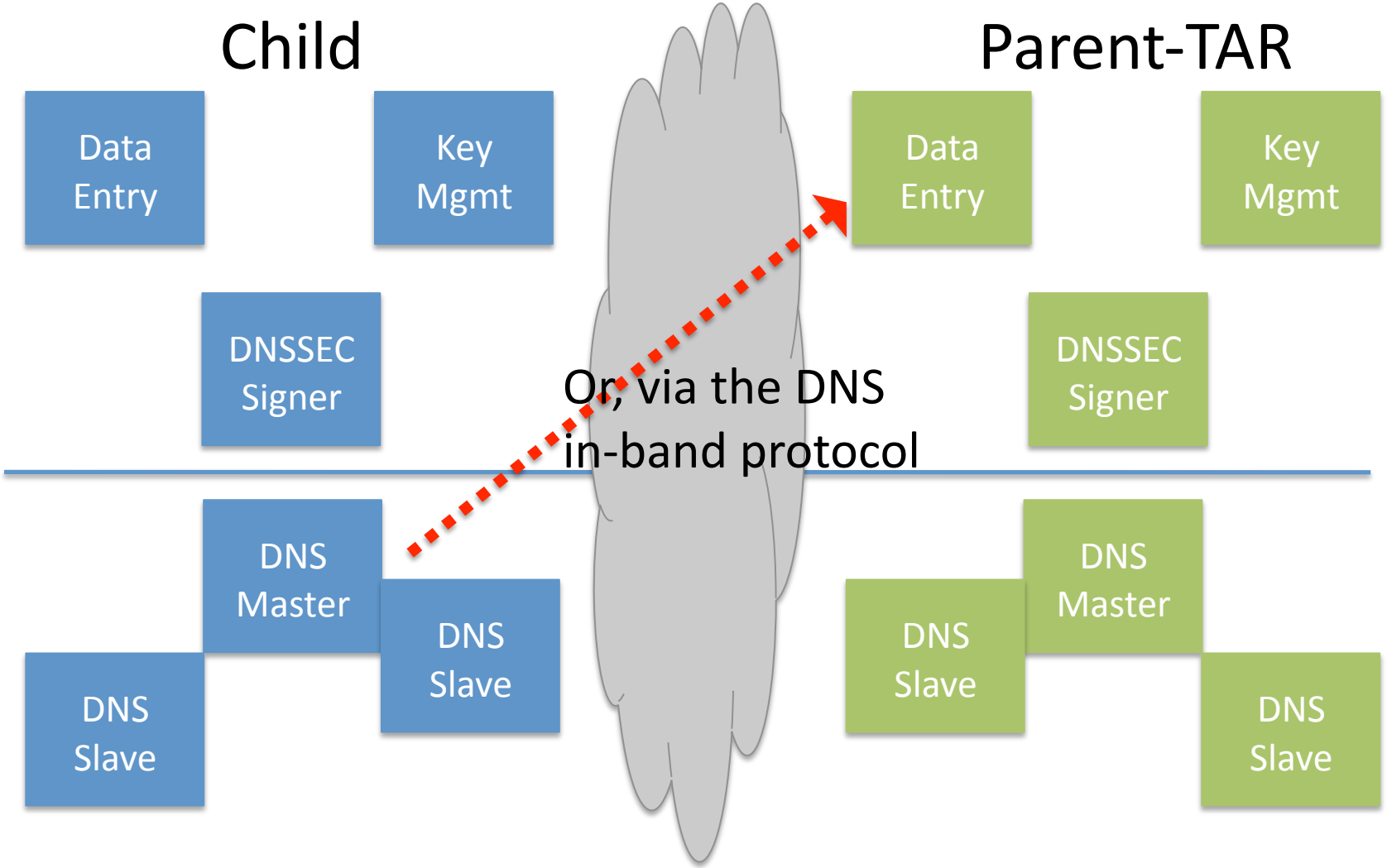
SEP: Pre-publish in DNS



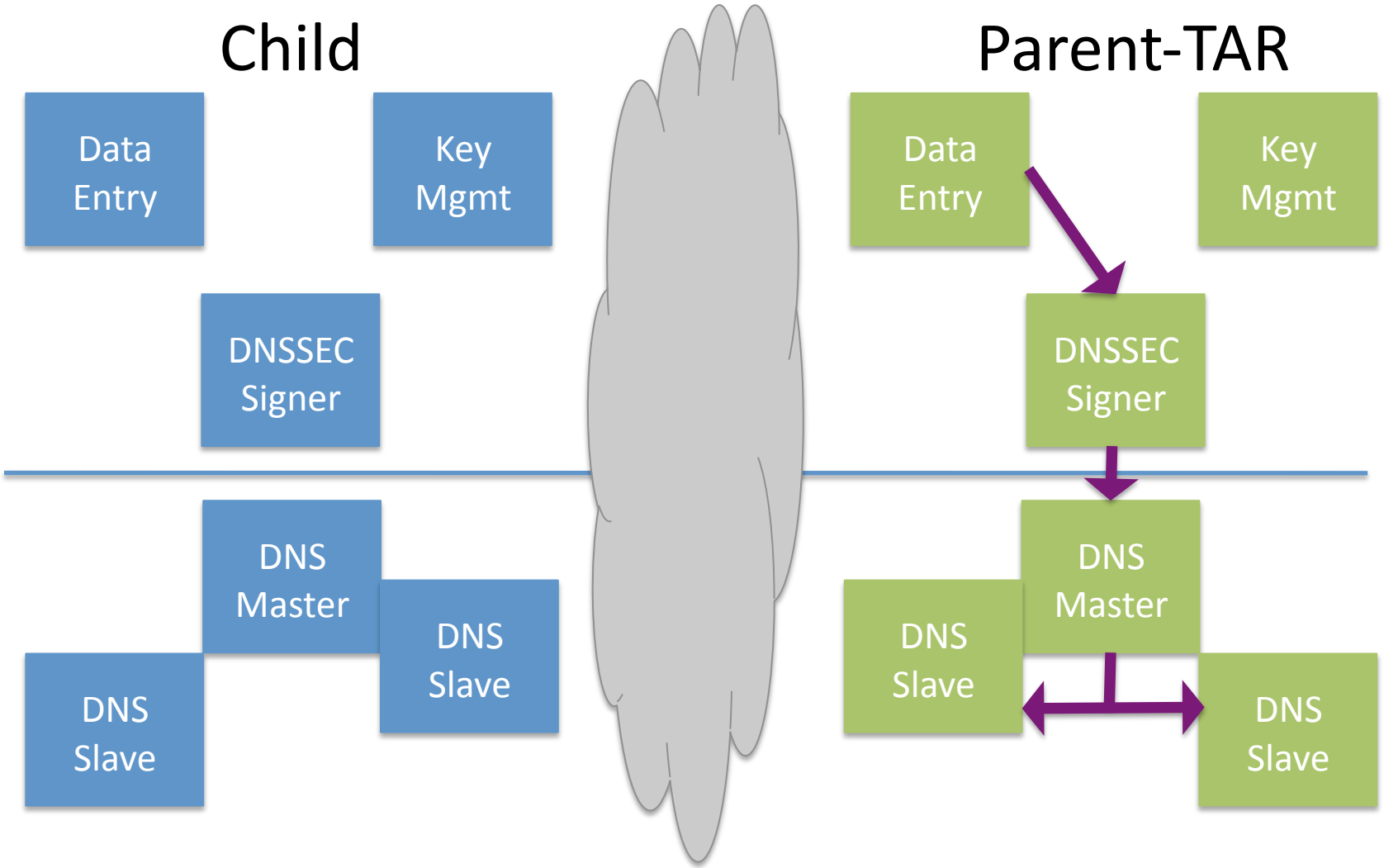
SEP: Request DS swap



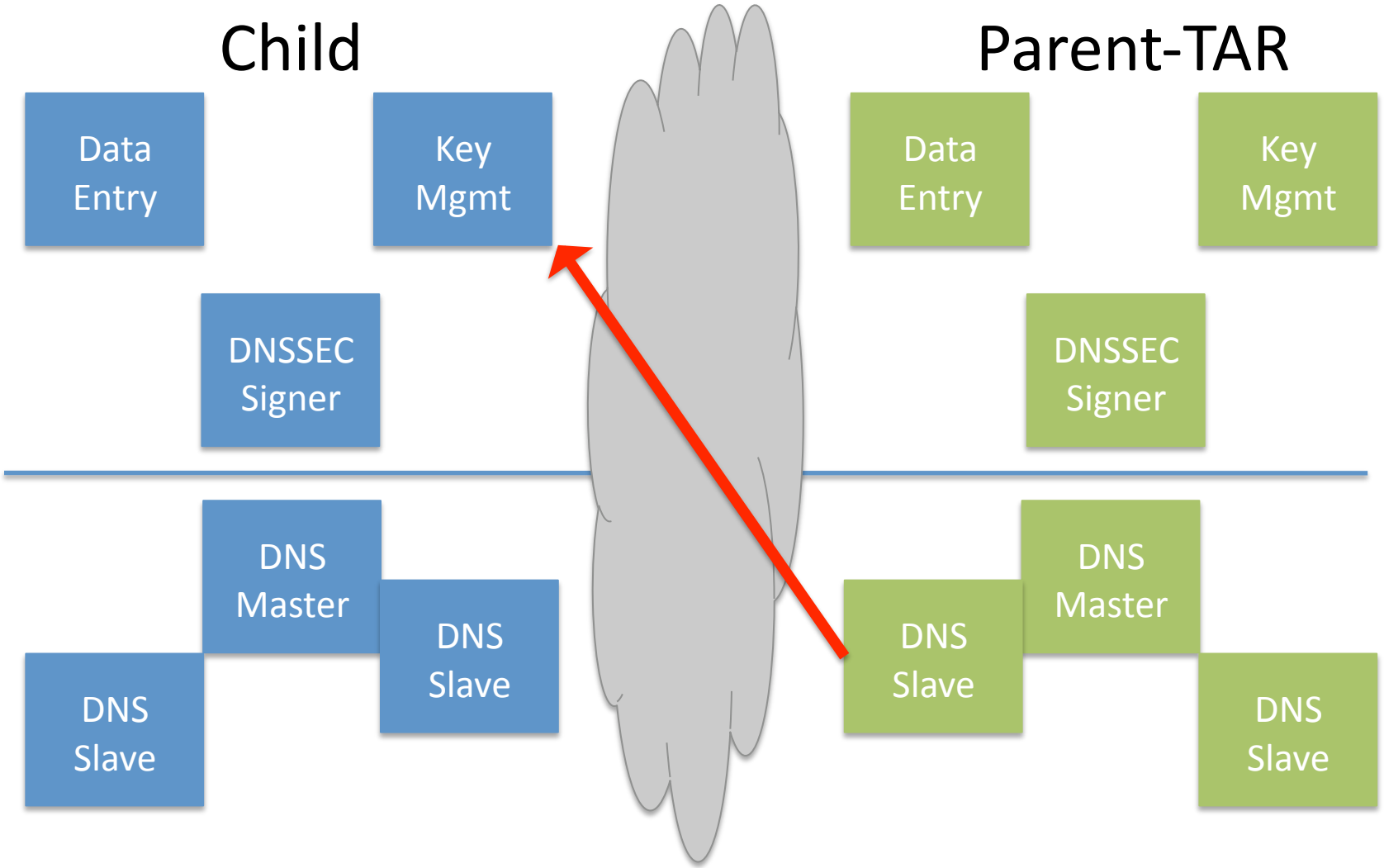
SEP: Request DS appear in parent



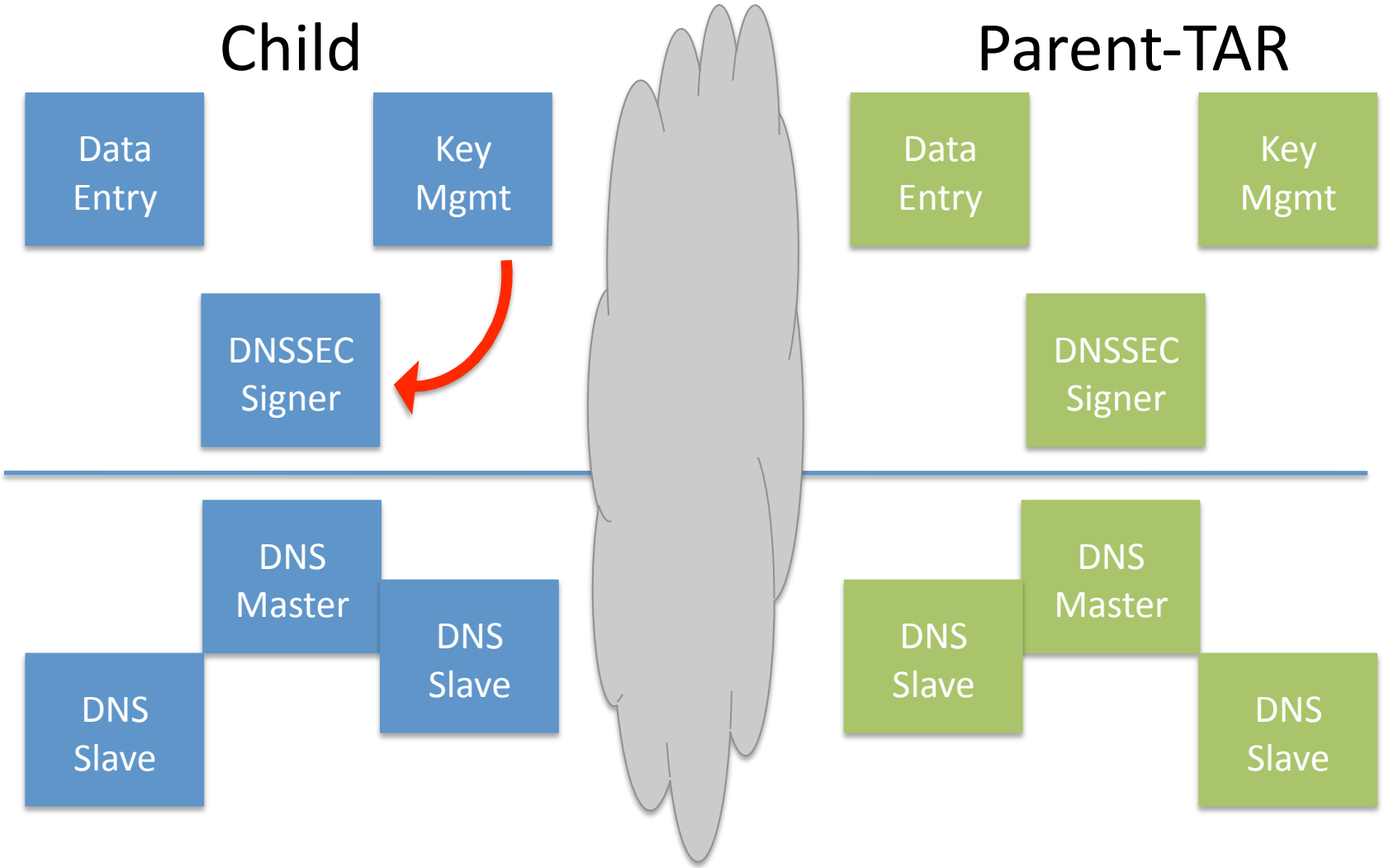
SEP: Parent-TAR signs



SEP: Confirm DS



SEP: Activate - revoke old that is



The basic steps

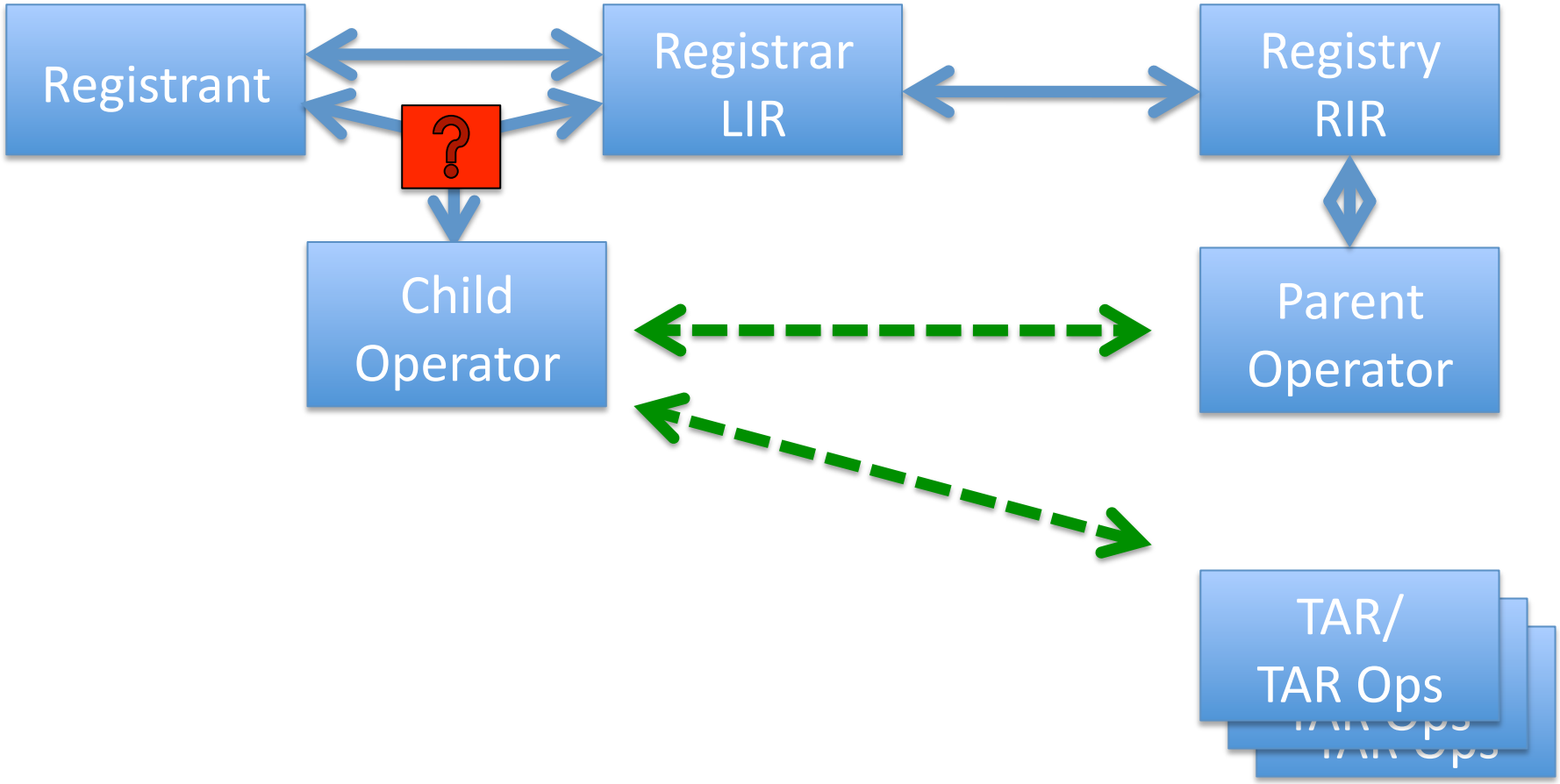
- The child publishes a new SEP
- The DS (new SEP) gets to the parent-TAR
- Parent-TAR publishes the (Signed) DS
- The child revokes the old SEP

- The above list does not addressing timing
- And it doesn't address including all parent & TARs

Shared Registry Model

- ICANN has specified a particular model
- Basic idea - separation between registrant and registry, registrar is middle-man; no consideration was given to DNS operations
  - Good for business
  - Causes a barrier for DNS in-band updates
- But **this is not the only way to do this**, arguably not even the majority of environments

Generalized *Provisioning* Model



Remember, ***Provisioning***

- When looking at this, remember we have to think provisioning (set-up) and not the lookup
  - This means that the parent has to get the data into the registry, not just a dynamic update
  - This does not preclude the use of the DNS protocol to pick up information
- That is why the validating cache using the parent-TAR DS record is not shown

Known requirements

- Function
  - Send new DNSKEY/DS to parent when it should replace existing; parent informs of completion; confirmation
  - More general, we should use the traditional add/modify/delete paradigm to accommodate more situations
- Security - Pair-wise authentication, tamper-proof xfer
- Accountability - Existing ops models need to be maintained
- Performance - SLA for request and response
- Predictable - E.g., Time to completion

Environments

- Registrant to Registry, each as own operator
- DNS outsourced by Registrant
- DNS outsourced by Registry
- Registrar in the middle (or chain of them)
- Registrar as DNS operator
- Registrant has registrar and separate operator
- EPP interface, SOAP/XML-based approaches

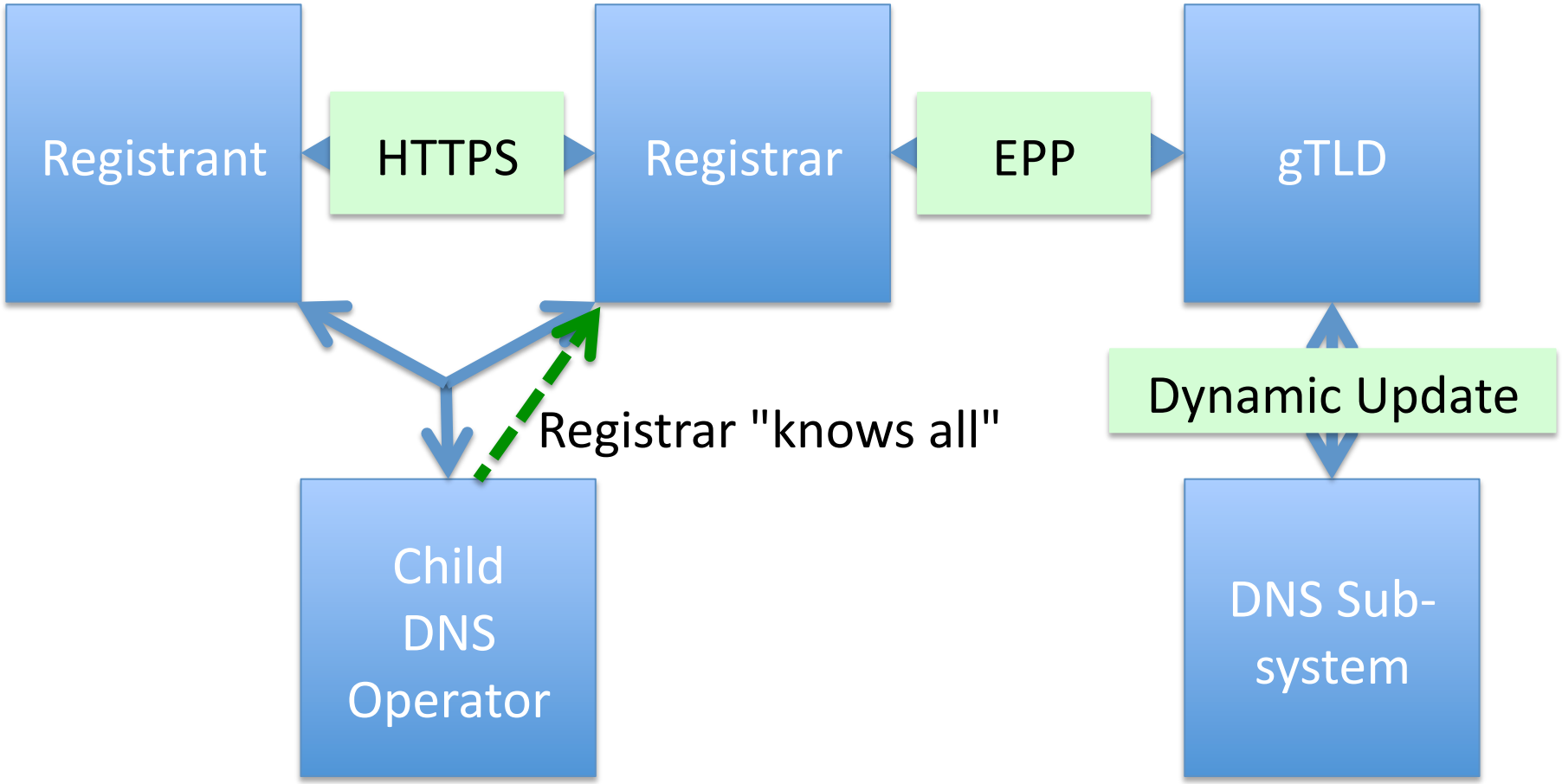
Related Problem

- Some DNS operators are signing all of their customer's zones
- When one of their customers transfers DNS operations (with or without changing "registrar"), the old DS record remains in the registry
- If the customer cannot remove the old DS, the zone will begin to fail DNSSEC validation

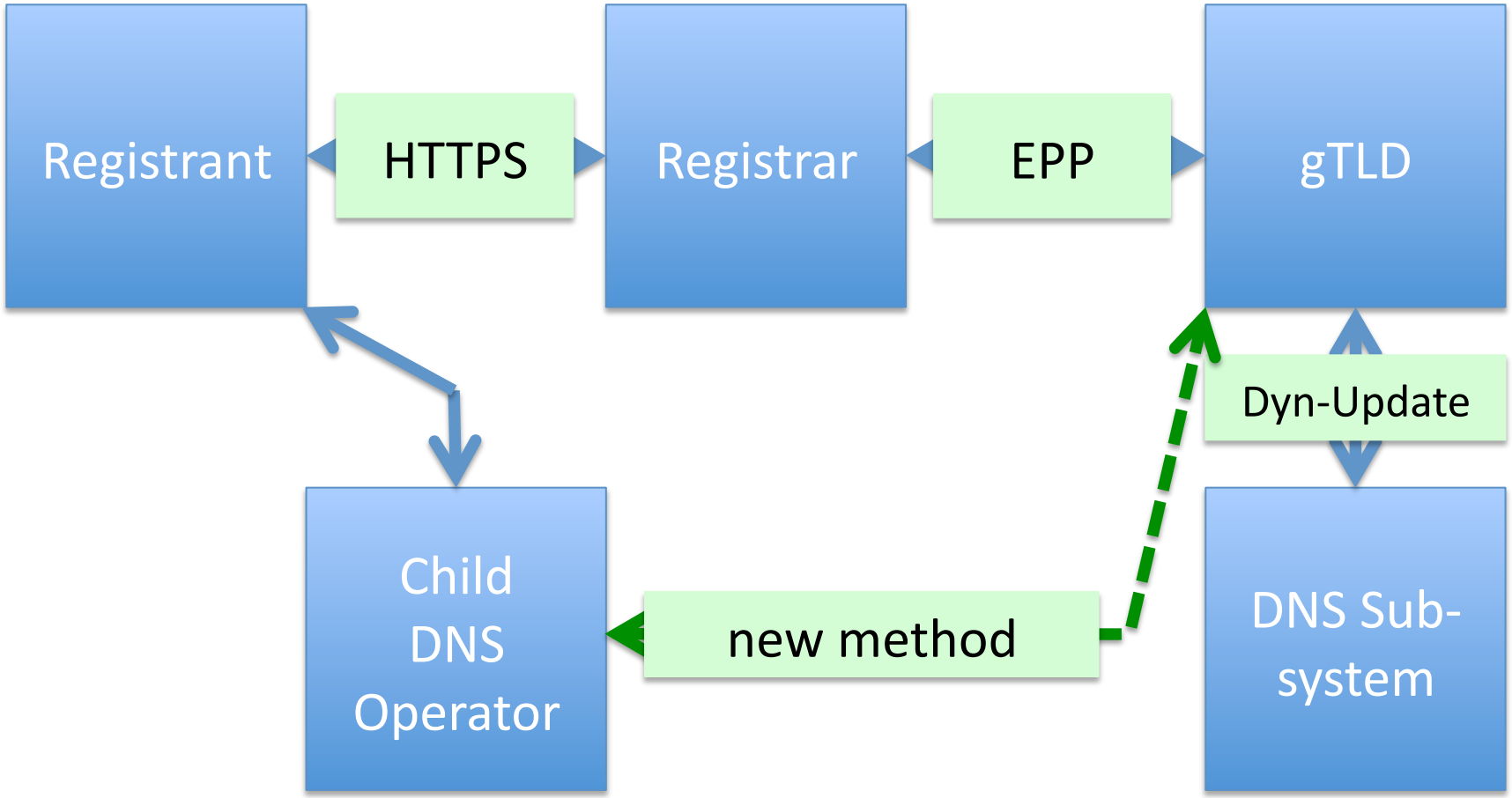
The next few slides are for ideas

- A few environments are sketched out
- Not complete, not particularly important
- But there to capture the wider issues involved

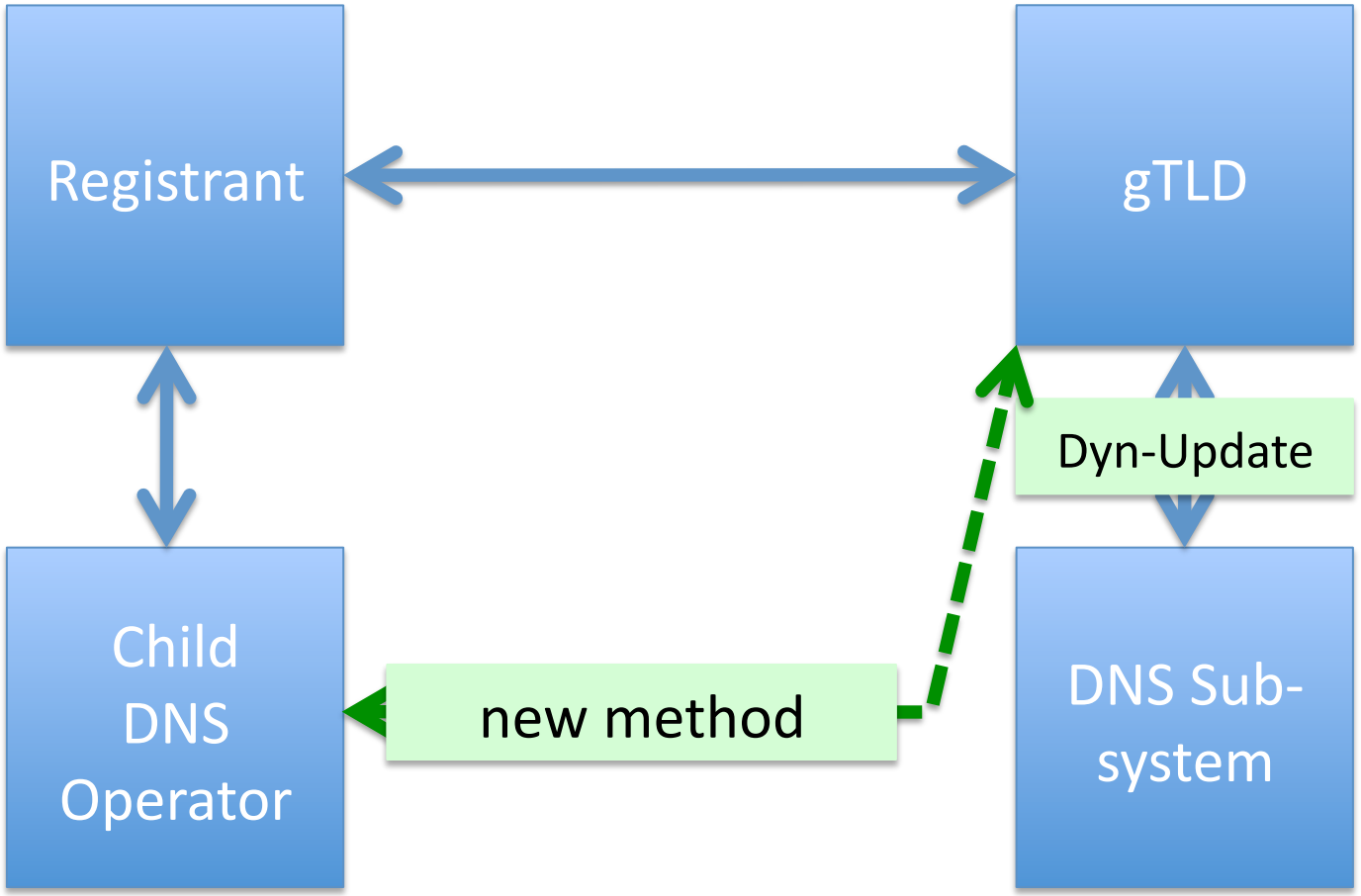
Fudging into an EPP SRM



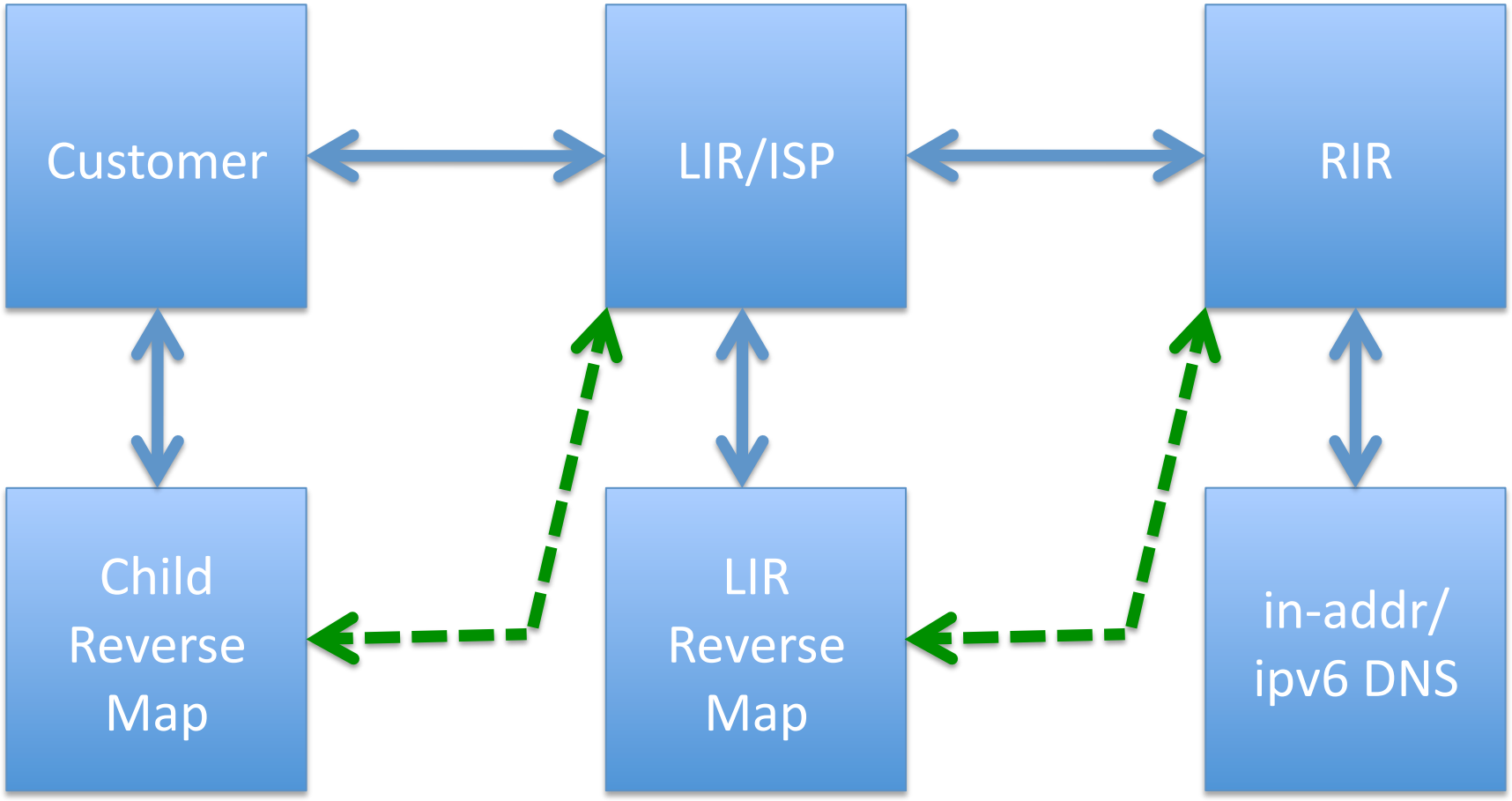
As an addition to EPP SRM



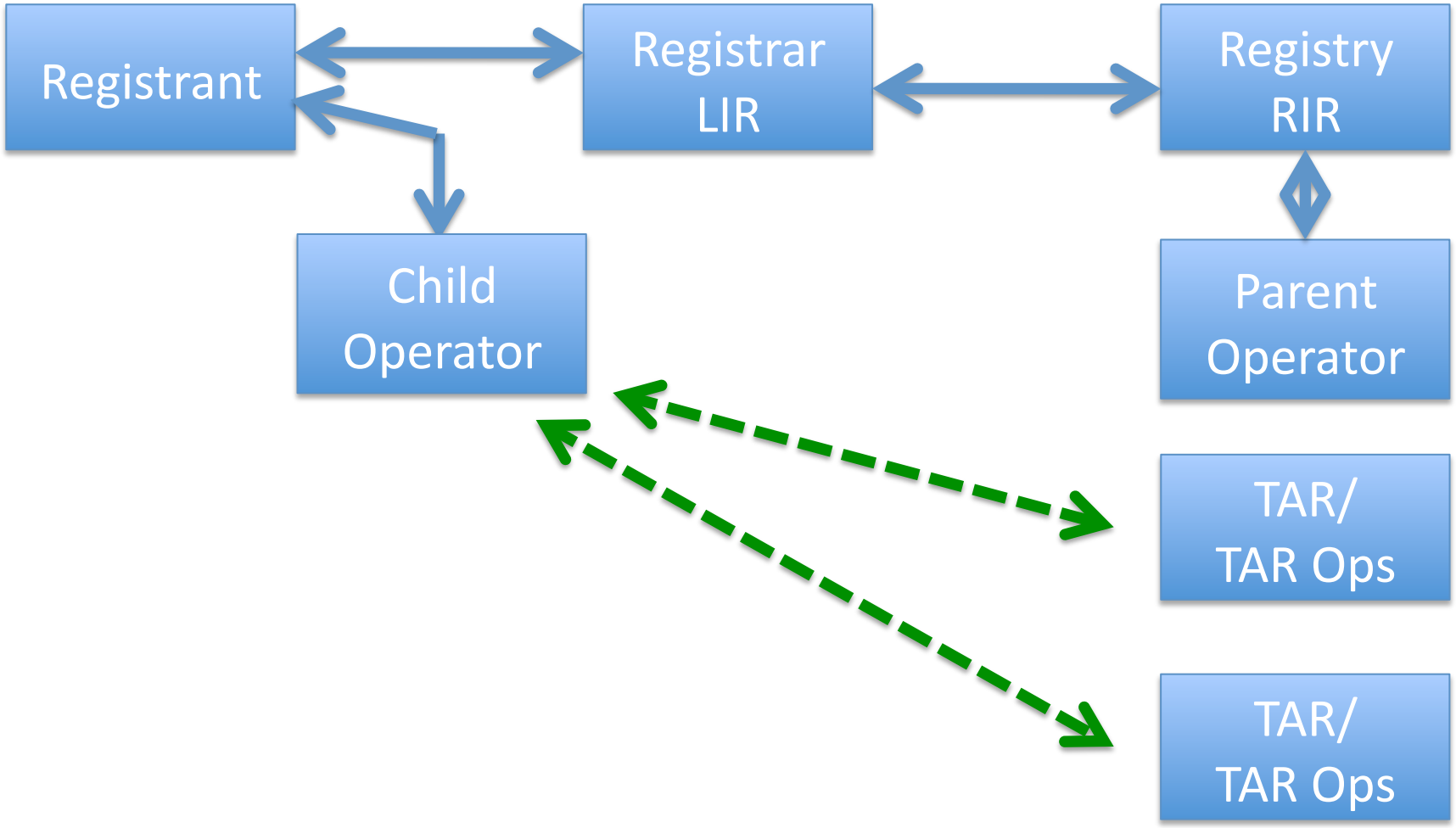
No Registrar, outsourced DNS



Reverse Map


Customer
LIR/ISP
RIR
Child Reverse Map
LIR Reverse Map
in-addr/ ipv6 DNS

Unsigned Registry, multiple TAR(s)



Solutions are Tempting

- A few proposed solutions have been out there
- Some claim out for years
- But there's been no good cut at requirements

- When do we need a solution?
  - Of course now, but, let's solve the right problem

Ultimately

- A standard can't be **mandated** for all environments, but we need to have a general purpose solution
- Or we will continue to have issues
- Only a standard will grow

I'm Done

- This is the last slide
  - I'm not even going to "ask" if there are questions.
  - Discussions are bound to follow...maybe not right now in the meeting, but later