MODERN framework

Jon Peterson
MODERN WG
IETF 101 (London)
A Win

• It’s done (hurrah), will work with the RFC-Ed to get this printed

• Just wanted to draw attention to the IESG review
  – We added a “privacy considerations” section
  – Numerous small editorial fixes for which we are grateful
  – Got a sense that this all makes a lot more sense when you assume STIR exists
    • And ACME telephone related work in progress
    • MODERN is kind of a framework for all those efforts
TeRI/DriP

Jon & Chris
MODERN WG
IETF 101 (London)
What are TeRI and DRiP?

• TeRI = Telephone-Related Information
  – Record format and client-server operations for acquiring, managing, and resolving telephone numbers

• DRiP = Distributed Registry Protocol
  – Gossip protocol for sharing information about resources like telephone numbers

• Last time we talked a bit about how they might play together
  – Today we have some use cases
First, a picture
Principles

• Ambition: design a distributed registry as an alternative to the centralized TN databases existing today
  – One of the motivating questions for MODERN
  – But this does NOT assume replacing any particular database or existing deployment, just a set of tools
• DRiP provides a way to share TeRI Records between Services
• For the most interesting use cases, DRiP nodes are themselves Authorities for a numbering space
  – Numbering spaces shared by multiple authorities are a key use case
  – We cast these in terms of CSPs, could have said Registrars
Self-Allocation

• Imagine that a CSP has a credential that allows it to sign for telephone numbers in a given range

  ...but initially no numbers in that range are allocated

  – It is a range shared for potential allocation by multiple CSPs
    • Perhaps easiest to imagine something like a new freephone (8xx) area code
      – Multiple RespOrgs can claim numbers through an administrative process: pretend we are automating that process
    • Policy constraints access to resources
      – Perhaps economic policy, perhaps just hard limits (10/month)

  – Imagine something like an experimental North American area code
Self-Allocation Use Cases

• Basically, a CSP creates a TeRI administrative Record, and then
  – signs it with a STIR credential, and then
  – promulgates it with DRiP to its peer nodes

• Two Cases
  – CSP allocates a block to itself
  – CSP allocates a single number to itself
    • Directly comparable to the 8xx case
  – In both cases, policy governs how the distributed registry authorizes the transaction – maybe a ”policy node” oversees
  – In both cases a peer node can vote “no” if a glare condition has arisen and the number had been allocated elsewhere in the gossip network
    • Or if a peer node does not trust the STIR credential, say
Assignment

• Once a carrier acquires a number through this process, it can assign it to a consumer
  – This requires creating a new TeRI Record signed by the carrier
    • Perhaps using teri-valid, or a successor mechanism
    • This Record adds an Element for assignment
• Maybe some node in the gossip network can track assignments by listening to gossip
  – Say, to measure allocation inventory.
    • Loops back into policy decisions made about allocation
Porting

• Start with an assigned number, now how does it get ported?
• New CSP issues a new Record for a single number
  – New CSP knows there is an existing Record covering that number in the DRiP network
• If old CSP disagrees about porting the number, it votes “no” on the new Record
  – Otherwise, the new Record is cached by the other nodes in the gossip network
• TeRI Record retrieval operations should process this correctly and use the new single-number Record
• (Again, this is just the use case, glossing over fine details here!)
Policy node votes “no” to enforce policies of the federation
Next Steps

• Gets us a bit closer to an architecture
• Need some feedback/interest
• Need a green light on TeRI, DRiP, etc.
TeRI and the MODERN Framework

Jon Peterson
MODERN WG
IETF 101 (London)
• Now a -04
• Made a few (very) small alignment tweaks to the model
  – Mostly to make Records less dependent on Operations
  – The framework here is getting mature
• It remains an abstract model
  – Bindings and encodings are modularized out
  – Elsewhere we mocked up a JSON syntax as an example
    • draft-peterson-modern-teri-json
TeRI Records

• TeRI Records would be available at Services
  – Services could be public, centralized and monolithic
    • Distributed, or private
    • The Operations and Info Model will be the same
  – Each TN might be associated with multiple Records
  – Records are trusted based on the Authority that generated them
    • Usually not based on the Service that shared them

• Entities from the MODERN framework act as
  – Clients
    • Users, CSP, Government Entities
  – Services
    • Registries, Registrars, CSPs
Records: Think SCRUD

• Search, Create, Read, Update, Delete
• Creation begins the lifecycle
  – A Registry always creates the first Record
    • Registrars then acquire Authority from Registries
  – Bootstrap administration record designating the Registry itself
• Should Records be partially updated, or wholly replaced?
  – Currently, only wholly replaced
  – Any Authority can update or delete its own records
    • In hierarchical assignment models, Authorities above the chain can delete the records of their delegates
TeRI Operations

• Acquisition operation
  – How do I request and receive numbers?

• Management operation
  – How do I provision information about a number?

• Retrieval operation
  – How do I get information about a number?

• Core conceit: these protocols access overlapping data
  – If you can provision it, you should be able to query for it

• TeRI provides a common information model
The TeRI Interfaces
Operations and Records

• Each Operation consists of a Request and a Response
  – All operate our core building block: TeRI Records
• Requests will have a Source, Subject, and Attributes
  – Source indicates the originator of the Operation
  – Subject would typically be a TN itself (or a range)
• Responses will have a Response Code
• TeRI Records contain information about TNs
  – Some Records might cover a range of TNs
The Acquisition Operation

• **Query:**
  – Source (Query Source, Query Intermediary)
  – Subject (Telephone Number/Range)
    • Used to have SPID, currently removed per MODERN scope
  – Attributes (constrains query, say, to finding a particular number in a range)

• **Response:**
  – Response Code
  – TeRI Record (newly generated assignment granting authority for this TN/Range)

**Result:** This makes the Client an Authority for that TN/range
The Management Operation

• Query:
  – Source (Query Source, Query Intermediary)
  – Subject (Telephone Number/Range)
    • Used to have SPID, currently removed per MODERN scope
  – TeRI Records (including Record ID)

• Response:
  – Response Code

**Result:** This replaces/deletes a previous TeRI Record, or creates a new one
The Retrieval Operation

• Query:
  – Source (Query Source, Query Intermediary)
  – Subject (Telephone Number/Range)
    • Used to have SPID, currently removed per MODERN scope
  – Attributes (constrains query: e.g., “voip” if only looking for VoIP, or Route Source, or Record ID)

• Response:
  – Response Code
  – TeRI Record

  **Result:** Retrieves Record if successful
TeRI Next Steps

• Energy needed, and discussion
• Need more input on Record elements
  – Varies by the use case
• Aligning with use cases
  – e.g. DRIP, Chris’s Identity registry
  – STIR is another
• Define further profiles and bindings
  – Need to flesh out JSON further, but anything else?
• Interest? Adoption?