Actually, this didn't make it into the IETF system in February, but it should in March.

Be sure to read http://www.firstpr.com.au/ip/ivip/Ivip-summary.pdf and my message on Mobility, scaling and update rates: http://psg.com/lists/rrg/2008/msg00535.html

Even with 10 billion micronets, most of them mobile, my guess is that the system only needs to handle a few hundred updates a second on average.

Network Working Group
Internet-Draft
Intended status: Experimental
Expires: August 21, 2008

R. Whittle
First Principles

February 18, 2008

Ivip Mapping Database Fast Push
draft-whittle-ivip-db-fast-push-00.txt

Status of this Memo

Copyright Notice

Abstract

Ivip (Internet Vastly Improved Plumbing) is a proposed map-encap system which is intended to provide a solution for the routing scaling problem - supporting growing numbers of end-user networks with multihoming, traffic engineering and portability, without further growth in the global BGP routing table. Ivip is also intended to provide other benefits, including a new form of IPv4 and IPv6 mobility and better utilization of IPv4 address space. To achieve these benefits, Ivip relies on a "fast mapping database push" system, which is required to securely and reliably deliver updates to the mapping database to hundreds of thousands - or potentially millions - of ITRs (Ingress Tunnel Routers) and Query Servers (QSes) all over the Net, ideally within a few seconds. This ID describes the requirements of such a system and how it could be implemented so as to cope with very large numbers of updates and ITR/QS sites.

Table of Contents

1. Introduction

The aim of this ID is to establish that the fast push approach to map-encap schemes is practical and desirable for very large numbers of micronets (EIDs in LISP terminology) and rates of change of the mapping database.

It is too early to quantify scaling limits and costs - and likewise there are no concrete design goals for the future. This ID is the first detailed step to developing at least one kind of global, fast push, mapping distribution system. Others may well be developed. Each such proposal provides a challenge to those who advocate a "full pull" global query server system (such as LISP-ALT), since the arguments for "full pull", with its inherent delay of some or many initial packets, rely largely on how impractical or undesirable it would be to use a full push or hybrid push-pull system instead.

This ID describes in some detail the most novel and perhaps difficult part of the Ivip system. The rest of Ivip's functionality will be comparatively easy to implement compared to the equivalents in other systems. For instance, the fast push system means that ITRs do not need complex mapping information, do not need to probe ETRs for reachability and do not need to make decisions about which ETR to tunnel packets to.

While contemplating this ambitious proposal, the reader is requested to remember that a successful implementation of something like Ivip would add immense value to the Internet - and not just by saving money due to solving the routing scalability problem. Immense value would be added by better utilisation of IPv4 address space and by the system's ability to provide a new form of mobility, for both IPv4 and IPv6, with generally optimal path lengths, few changes to the mobile host and no changes required for the correspondent host.

The benefits of a scheme such as this should motivate considerable effort to develop and deploy some kind of fast push map-encap scheme. These benefits are not just for the long-term good of the Net or Humanity, but include direct benefits to those who provide the new form or address space, and to those end-users who adopt it.

Ivip's overall architecture is described in [I-D.whittle-ivip-arch]. This ID is the first in a series to describe particular aspects of the proposed system, starting with the most ambitious feature of the design: a system to push large numbers of small items of "mapping data" to potentially millions of sites all over the Net, securely and reliably - and ideally within a few seconds. Please see the Ivip homepage http://www.firstpr.co.au/ip/ivip/ for further material and latest updates, including the text of IDs which are delayed by the

IETF submission cut-off dates.

Ivip is one of several "map-encap" schemes currently being considered by the IETF Routing Research Group. Others include LISP (Locator/ID Separation Protocol) [I-D.farinacci-lisp], APT (A Practical Transit Mapping Service) [I-D.jen-apt] and TRRP (Tunneling Route Reduction Protocol) [TRRP].

The most unusual and demanding part of Ivip's fast-push system is the network of "Replicator" servers which fan the mapping updates out to the full database ITRs (ITRDs) and full database Query Servers (QSDs) at recipient sites. Before describing this, several subjects are discussed in some detail:

1. The benefits which the fast push system brings to Ivip, compared to other map-encap schemes.

2. The goals, non-goals and challenges of this fast push system.

3. How multiple RUAS (Root Update Authorisation System) systems combine their mapping changes into a form which can be fanned out to the "Replicators".

1.1. Outline of the RUAS, Launch and Replicator systems

In this ID, the largest part of the fast push system is comprised of thousands (perhaps several hundred thousand in the long term future) of essentially identical "Replicator" servers. There may be other, better, approaches, but this serves as a starting point.

There is a single stream of packets which carry the combined mapping updates for the whole Ivip mapped address space. A finite number (ten to a few dozen at most) of RUASes work together with a shared "Launch system" of distributed servers, which generates multiple identical streams of update packets over secure links to the first level of Replicators.

At the first level, each Replicator receives two identical streams, over separate authenticated and encrypted links, from two different Launch servers in different geographical locations, and over different physical long distance links. The Launch system and perhaps the first level (1) of Replicators will probably be implemented with private network links, rather than relying on open Internet addresses which are subject to flooding attacks.

If a packet goes missing from one stream, it will probably be present in the second. As the packets arrive, the Replicator takes the first one from either stream and sends its contents out simultaneously on a

larger number of similar links to the next level of Replicators. Consequently, the delay time for update information passing through a Replicator is measured in milliseconds, and is comparable to the delays experienced in routers.

In this way, each Replicator consumes two identical streams from geographically and topologically different sources, and fans the content of the streams out to some larger number of Replicators, ITRDs or QSDs at the next level. This number of output streams per Replicator may be in the tens to one hundred range, depending on the volume of updates. Initially, it would be quite high, when update rates are low - meaning that the initial global Replicator network could serve the growing number of ITRDs a QSDs with few levels of Replicators, and with each one fanning out updates to a large number of Replicators at the next level. (It is possible to imagine multiple parallel Replicator networks to share the load, but this is not contemplated further in this ID.)

After some number of levels of replication, determined by local conditions, the streams deliver the update information at an ITRD or QSD. Ideally, each such end-point receives two streams from two geographically dispersed Replicators. These need not be at the same level, so the system is relatively flexible, and each Replicator will generally be sending a complete streams of packets.

The Launch system generates the stream as a variable number of packets on a regular schedule, such as every second. Data within each packet enables ITRDs and QSDs to authenticate the mapping information, and to request from remote servers any packets which did not arrive.

1.2. Background assumptions

For the purposes of this discussion, it is assumed there will be a single global Ivip system, with multiple organisations being responsible for the management of the various blocks of address space which are managed with Ivip. It would be technically possible to run multiple Ivip systems, or Ivip-like systems, in parallel, with separate networks of ITRs, or with separate database fast push systems and some separate ITRs with some ITRs handling traffic for multiple such systems.

It would also be possible for an organisation to establish an Ivip-like system, without reference to any IETF RFCs, and to conduct a business renting out address space in small, flexible, chunks, with portability and multihoming via any ISP who provides the requisite, relatively simple, ETRs. Likewise for the mobility potential of Ivip.

However, for simplicity, this ID assumes that Ivip development will be coordinated into a single global system, as DNS is, following appropriate IETF engineering work and administrative decisions in RIRs and other relevant organisations. A development timeframe of 2009 to 2011 is assumed, with widespread deployment being achieved in the 2013 to 2015 timeframe.

Except where noted, it is also assumed that all full database ITRs and Query Servers receive a single global body of mapping data. An alternative to be considered in the future is more complex, and has various problems, but may be of value: that each site may choose to receive a full push feed of mapping information for only some parts of the global database, and rely on access to query servers in another network when packets must be handled which are addressed to micronets not included in the pushed subset. This approach is contemplated in LISP-NERD [I-D.lear-lisp-nerd].

In addition to the global fast push database update distribution system discussed in this ID, Ivip also involves Query Servers sending "notifications" to ITRs which recently requested mapping for a micronet whose mapping has just changed. This is a second form of push - on a local scale - and will be discussed in a future ID concerning ITRs and Query Servers. (It is also discussed in the ivip-arch-00 ID.)

The fast push system is complemented by a second system (discussed later in this ID) by which ITRs or Query Servers initiate downloads of snapshots of sections of the database - for initial boot up - and by which they can request specific update packets which did not arrive via the fast push system.

This ID concentrates on IPv4, since the future map-encap scheme is urgently needed for IPv4, but will not be so urgent for IPv6 for at least several more years. In principle, the same arrangements will apply for IPv6, with a different and more verbose data format than the 12 or so bytes required for each IPv4 mapping update. It may make sense to defer finalisation of any future IPv6 map-encap scheme until substantial operational experience was gained with the IPv4 scheme.

A contrary perspective is that IPv6 will never be widely adopted until end-users have multihomed (and portable) address space. Since SHIM6 cannot provide the required network-centric approach to multihoming (though SixOne [I-D.vogt-rrg-six-one] may achieve this), the only way of providing multihoming to large numbers of IPv6 end-user networks without the unwanted bloat in the DFZ routing table is to deploy a good map-encap scheme ASAP.

1.3. It may not be so daunting...

Ivip documentation is written with a preference for detailed discussion over terseness. So Ivip IDs may appear rather daunting at first. Hopefully these IDs will be clearly understandable, and the reader will recognise that the future map-encap scheme is a momentous development, requiring detailed consideration.

This ID focuses on handling billions of micronets and potentially thousands or tens of thousands of updates a second. Ideally, with good design, some more elegant approaches can be found than those presented below.

Also, during initial deployment, the demands on the fast push system will be far lighter than those anticipated below, so the system might initially be somewhat simpler. In the initial stages of introduction, there may be little need to deploy dedicated servers for the "Replicator" functions, since the volume of updates may be so light as to make it practical to run this software on existing servers, such as nameservers.

Furthermore, in the early years of introduction, when there are hundreds of thousands or a few million micronets, the low level of update packets (compared to the highest imaginable levels contemplated below) should enable each Replicator to fan out to many more next-level Replicators than would be possible when hundreds of millions or billions of micronets are handled by the system. This would mean fewer levels of Replicators, fewer Replicators and generally faster delivery of the mapping information than would be possible with current technology if the system was handling billions of micronets.

2. Objections to push and hybrid push-pull schemes

Objections to a full push or hybrid push-pull map-encap schemes constitute arguments for full pull schemes such as LISP-ALT, including:

1. The size of the database (primarily the number of micronets multiplied by the average size of the mapping data) will grow to be so large that it will be impractical or undesirable either to concentrate the data in any one place, or to make copies of it to multiple locations. This ID is intended to show that a push scheme, in this case fast push, can scale to very large update volumes and numbers of micronets.

2. The rate of change to the database will grow to the extent that it will be impractical or undesirable to send all those changes to all ITRs (full push) or to some ITRs and Query Servers (hybrid push-pull). Ivip's flexibility addresses this second question to a significant degree by enabling the one consistent architecture to be deployed with local decisions about how far to push the mapping data, and therefore how much remaining distance from the location of most ITRs to be sending map requests and getting responses. As long as the optimal number of full database query servers in the world is a few hundred or more, then Ivip's hybrid push-pull approach is clearly superior to a global query server system, because the paths of queries and responses will be much shorter and therefore more reliable and cheaper.

3. That any degree of push typically involves sending mapping data to sites which will not use it. This is a valid concern and Ivip is not intended to provide mapping changes for end-users necessarily free-of-charge, just as the TCP/IP protocols are not intended to be used in a way in which any one party persistently sends unwanted packets to the service of any other party. Administrative and business arrangements for this, to deter frequent changes and/or to ensure end-users' mapping changes involve a contribution to the cost of the fast push system, will be discussed in the planned ivip-deployment ID.

3. Ivip compared with other map-encap schemes

LISP-NERD [I-D.lear-lisp-nerd] is a "full push" map-encap system, in which the full mapping database and updates are "pushed" to every ITR. Updates are sent from servers in response to periodic requests from ITRs. Ivip's fast push involves a dedicated network of "Replicator" servers, which push a continual stream of updates to all full database ITRs (ITRDs) and full database Query Servers (QSDs). These devices passively receive the updates, which arrive ideally within a few seconds of the end-user changing their mapping.

Because Ivip involves caching ITRs (ITRCs), there is no need to push the full set of database updates to every ITR, thus overcoming the primary inefficiency and scaling objections to a "full push" scheme.

LISP-ALT [I-D.fuller-lisp-alt] is a "full pull" system, with a global ALT network by which ITRs send mapping queries to the authoritative query servers, which are typically ETRs. (ALT also involves sending initial traffic packets by this global network, where they also constitute a request for mapping information.) The primary benefit of a "full pull" system is that the mapping database is fully distributed, and no traffic or hardware is involved in pushing the mapping data anywhere. This means the end users can have as much mapping information as they like, and change it as frequently as they desire, without requiring that these changes be sent to ITRs and QSes all over the world. The primary objection to such a scheme is that the necessarily global nature of the query server network will often delay the delivery of initial packets by times which are likely to cause significant slowdowns in session establishment, causing potential difficulties for higher level protocols and dissatisfaction for users. Other objections include difficulty trading off caching time for faster responses to mapping changes, and bottlenecks in the ALT network and in the few authoritative Query Servers (ETRs).

TRRP [TRRP] too involves a global query server system, based on separate DNS-like network, so the same difficulties arise with initial packets in a new communication session potentially being delayed for large fractions of a second, or longer.

In Ivip, all mapping queries are handled by local query servers, which are likely to be faster, more reliable and involve less overall query-response traffic than any global system such as LISP-ALT or TRRP.

APT [I-D.jen-apt] is the only proposal other than Ivip which involves "hybrid push-pull" - pushing the full database to a subset of the ITRs and to full database Query Servers (APT's Default Mappers integrate both functions), with the remainder of the ITRs sending

their mapping queries to a local Default Mapper.

APT involves a new instance of BGP operating on existing routers, to flood the mapping changes to all participating ISPs. This is a much slower form of push than is intended with Ivip's new protocols and specialised "Replicator" servers.

4. Benefits of Fast-Push

Many of the benefits of Ivip are entirely dependent upon the ability to convey to every full database ITRD and QSD in the world an end-user's command to change the mapping of one of their micronets (one or more contiguous IPv4 addresses or /64 prefixes for IPv6). Before describing the goals and potential implementation of the fast-push system, the benefits will be discussed in some detail.

The future map-encap architecture should be as powerful and flexible as possible - to solve the immediate routing scalability problem (which is closely bound to the IPv4 address depletion problem) and to provide as many other benefits as possible. For instance Ivip is intended to provide a new form of efficient mobility. A widely deployed map-encap scheme is a powerful piece of infrastructure which may in the future play a role in migrating from IPv4 to IPv6 or some other future Internet addressing architecture. The high speed with which information can be transmitted to the sites containing ITRDs and QSDs is likely to make such a system more suitable for architecturally important tasks in the future which cannot be foreseen today.

Since the future map-encap architecture is a major addition to the Internet, with its new kind of address space ideally being adopted ubiquitously be end-users large and small, it makes sense to implement the architecture with specifically designed protocols and servers which enhance the new architecture's modularity, power, speed and scope for future enhancements. This general principle and the specific reasons listed below are strong arguments for developing an ambitious and novel proposal such as Ivip.

However, the new protocols and software which will be needed for this fast push system are not necessarily highly demanding. All elements of the proposed fast push system can be implemented as software on conventional servers. The overall fast push system will ideally be a much more secure, reliable, predictable and easy to manage system than any global query server system such as LISP-ALT.

4.1. Modular separation of the multihoming restoration functions

Map-encap schemes other than Ivip (LISP, APT and TRRP) are based on the assumption that due to the vast size of the mapping database and/or its rapid rate of change, that it is or will in the future be impossible for the end-user's wishes to be conveyed to all the world's ITRs within a few seconds.

This assumption is untested. Perhaps this ID will convince some people that the assumption is wrong. Perhaps it will fail to do so,

and hopefully a better proposal for a fast push mapping distribution system will be developed. It would be a terrible lost opportunity if the new architecture was built on the assumption that it must be based on pure pull, or slow push, when in fact it is possible and clearly more desirable to use fast push.

With the assumption that fast push is impossible, or for some reason undesirable, each ITR must make its own decisions about multihoming service restoration.

For instance, the ITR must be given two or more ETR addresses and some criteria for choosing which one to tunnel traffic packets to. The decision could involve Traffic Engineering (TE) functions such as load balancing, but the most important decision is which ETR to send traffic to when one or more of the ETRs is unreachable. This means that each individual ITR needs to determine reachability to each ETR listed in the mapping information and to make decisions based on this reachability and the criteria contained in the mapping information.

Consequently, these proposals would result in the following tasks being built into the map-encap scheme itself:

1. The exact methods by which each ETR's reachability could be determined, presumably by each ITR operating in isolation.

2. Similarly, any other reachability functions, such as determining whether and ETR is capable of delivering packets to the destination network.

3. The logic of all decisions regarding ensuring continued connectivity for multihomed networks, and likewise for TE. These need to be codified as part of the map-encap protocol, because they need to be part of the functional specification for all ITRs.

4. Similarly, the logic of these decisions needs to be fixed as part of the map-encap system in order that a format for mapping information can be defined.

5. Since these functions involve ITRs probing ETRs, it is also necessary for the map-encap scheme to standardise the ways ETRs respond to such probes. This may involve ETRs making decisions based upon their own reachability and the reachability of other ETRs (however determined).

Consequently, a great deal of complex functionality needs to be defined in RFCs and implemented in every ITR and ETR. This becomes frozen into the map-encap scheme, making it difficult to implement

even minor variations on these functions once the system is widely deployed.

The inability of these other schemes to give the end-user direct real-time control of how ITRs handle packets whose destination address falls within one of their micronets means that the map-encap scheme is a monolithic system. In addition to tunnelling packets from ITRs to ETRs, these systems force all end-users to rely on each system's inbuilt functions for detecting reachability, making decisions about where to send packets etc.

Unless the map-encap scheme is made exceedingly complex (with consequent development delays, costs and security problems with ITRs and ETRs) it is likely that some or many end-users will be dissatisfied with the limited functionality the system provides. Similarly, the system cannot be used for any other purpose without a complete upgrade to all ITRs, and possibly ETRs.

Ivip provides a map-encap scheme whose sole function is to collect traffic packets into ITRs and to tunnel them to the ETR the end-user specifies for whichever micronet the packet is addressed to. Although ITRs and ETRs do need to work together to solve some Path MTU Discovery and Fragmentation problems, the ITRs are not involved at all in determining reachability or making any decisions.

The rapid (ideally, a few seconds or less) response of all Ivip ITRs to the end-user's mapping commands means that end-users can (and must) supply their own multihoming monitoring system and make their own decisions about how to control the behaviour of ITRs, for multihoming, TE, portability or whatever other purposes the end-user requires.

There may well be a role for IETF work regarding detecting reachability of multihomed networks via various ETRs, but this is not part of the current Ivip proposal.

End users can supply their own systems, make manual decisions, or hire the manual or automated services of other organisations to control the mapping of their micronets. This is a completely modular approach to multihoming etc. - in contrast to the other proposals which monolithically build those functions into their proposed global networks and protocols.

4.2. Reduction in the size of the mapping information

With real-time end-user control of ITR behavior, it is not necessary to provide multiple ETR addresses, together with priority information regarding multihoming etc. Consequently, the quantity of mapping

information in each update can be greatly reduced.

In Ivip, for each micronet, only three items of information are specified:

1. Start address of the micronet: 4 bytes for IPv4, 8 for IPv6 (assuming /64 granularity).

2. Length of the micronet, as an integer number of IPv4 addressees or IPv6 /64s: in principle 4 bytes for IPv4 and 8 for IPv6, but in practical terms, half these figures are probably adequate.

3. Address of the sole ETR to which packets addressed to this micronet should be addressed: 4 bytes for IPv4 and 16 bytes for IPv6.

   Note: Ivip is less functional than the other schemes in one important respect. The other schemes provide TE in the form of load spreading over multiple ETRs for each given micronet (EID prefix, in LISP terminology). Ivip has no such capability. TE for a single Ivip micronet consists solely of steering the traffic for this micronet to one ETR or another. Load sharing for a single IPv4 address or IPv6 /64 is not possible with Ivip. However if the traffic can be split over multiple such IPv4 addresses or /64s, then each can be made into a separate micronet so that load sharing can be achieved by mapping each micronet to a different ETR. Despite this limitation, Ivip may prove to be better for many TE applications due to end-users being able to fine-tune the mapping in real-time.

All other schemes involve the specification of (typically) two or more ETR addresses, plus other information regarding priorities and service restoration decisions. Ivip's more compact mapping information makes the task of distributing updates easier than for a monolithic scheme in which ITRs make multihoming restoration decisions. Ivip may involve a greater number of updates, so this advantage may be reduced or reversed. However Ivip's functionality is different from that of competing schemes, so direct comparisons of the compactness of mapping updates are not particularly illuminating.

Since mapping information must be stored in every full database ITR or Query Server, Ivip's more compact mapping is an advantage in terms of storage space compared to that required for LISP-NERD or APT. Another consideration is that the other schemes use full prefixes for their micronet/EID lengths, which is more compact, but less flexible, than Ivip's integer number of IPv4 addresses or IPv6 /64s.

4.3. Reduced ITR and ETR functionality

As noted above, the fast push system enables real-time end-user control of the world's ITRs, removing the need for decision making and reachability probing from ITRs and ETRs. This contributes to Ivip being simpler to design, deploy and manage.

4.4. Greater security through simplification and modularization

Similarly, the many security problems, including Denial of Service (DoS) problems, which arise in other schemes when ITRs receive mapping information from distant, unknown, ETRs are avoided when the ITR no longer needs to make decisions about reachability and multihoming service restoration.

Instead, security of the mapping information needs to be assured as part of the design of the fast push system. Since this consists of a limited number of streams of data, from well-established sources, this should be easier in general than relying on ITRs and ETRs to communicate across the Net, without prior arrangements, and without prior knowledge of each other's existence.

4.5. IPv4 and IPv6 mobility with generally optimal path lengths

Ivip enables end-users to exercise fast, essentially real-time, control of which ETR packets addressed to their micronet(s) are tunnelled to by the global system of ITRs. This enables a new form of mobility with some unique and favourable characteristics compared to traditional approaches to mobile IP. This is discussed further in [I-D.whittle-ivip-arch] and in a forthcoming ID devoted to Mobility.

Briefly, the idea is that the mobile host (or whatever device is the recipient of traffic for a micronet of addresses) retains its IP address wherever it is located, and establishes one or more care-of addresses in various networks.

For instance, a laptop or cellphone may have a WiFi connection to ISP A and so a temporary care-of address (perhaps or probably behind NAT) in that network. It then establishes a link via 3G to ISP B, with another care-of address there. The mobile device needs to establish tunnels from each care of address to one or more ETR-like devices, which are optimised for mobility. These Translating Tunnel Routers (TTRs) combine ITR and ETR functions with the ability to authorise and service a two-way encrypted tunnel established from the mobile device. An external, distributed system of servers enables the mobile host's software to choose TTRs which are either within, or close to, the network it is currently connected to. The TTRs and the TTR location systems would typically be operated by companies who

charge end-users.

The mobile device sends outgoing packets to the TTRs, which are able to forward them to the rest of the Internet, perhaps performing ITR encapsulation at that point. The mobile device and/or some external system controls the mapping of the micronet for this device's address space, causing all the world's ITRs to tunnel traffic packets to one or the other of the two TTRs which the device has connections to.

Assuming the TTRs are relatively close to each point of connection to the separate networks, then total path lengths from corresponding hosts will generally be optimal or close to optimal. There is no "home agent" or "triangle routing". The system should work fine with both IPv4 and IPv6, with no changes required for corresponding hosts, and only some additional software, rather than actual host stack changes, for the mobile host.

Ivip's fast push system is instrumental in enabling this new form of mobility. Mobility such as this cannot be achieved with a slow push system, or with a pure pull system such as LISP-ALT - unless perhaps such a system had a fast, global-scale notify (cache invalidation and mapping data update) system, which would probably be more complex and less secure than Ivip's fast push system.

Even when not used for multihoming or mobility, the real time control of mapping enables the micronet address space of end-users to be completely portable between any ISPs with suitable ETRs. Portability and multihoming are the most important goals being considered by the RRG [I-D.irtf-rrg-design-goals] (though "portability" is generally described in other terms). These are marketable attributes of the new address space. The real time mobility which Ivip can provide is still more marketable, and a further reason to expect that the new architecture will be adopted willingly and profitably by ISPs and end-users alike, rather than due to them having to be cajoled into using it, for instance on the basis that it is the responsible way to obtain address space compared to gaining conventional BGP-managed PI space.

This form of mobility is not available via other map-encap schemes. It does not seem to be widely known, or considered to be a possibility by most mobile IP developers - probably because they either haven't heard of the concept of a global ITR-ETR network, they don't think any such thing will be built, or they haven't contemplated that such a network could be driven by a fast push mapping distribution system.

4.6. Better suited to future enhancements

A well designed centralised database update distribution system may be more suitable than a global query system such as LISP-ALT for enhancement in the future, in which the ITR-ETR system is required to perform new and unanticipated functions.

For instance, perhaps the ITR-ETR system could be used in some creative way, with special addressing arrangements, to provide automatic communication between IPv4 hosts and IPv6 hosts, via gateways which the ITRs would tunnel packets to. Perhaps this could be done with no IPv4 host changes and some minimal IPv6 host changes. This is a highly speculative suggestion, but is an example of how the ITR-ETR network could be used to create, or support, an important new architectural development. Some ITRs and Query Servers could be upgraded to the new functionality and the information to control these new functions would be sent as part of the main stream of updates, in a distinct format which would be ignored by standard ITRs and Query Servers.

5. Goals, Non-Goals and Challenges

5.1. Goals

The overall goal of the fast push system is to enable end-users, who manage the mapping of their one or more micronets of address space, to securely, reliably and easily communicate their mapping change command to some organisation with which they have a business relationship, so that that change will be propagated to every full database ITR and Query Server as soon as possible.

"As soon as possible" means typical delay times of a few seconds, ideally zero seconds, but in practice probably four to five seconds. (Most of this delay is in the RUAS and Launch systems, which could be optimised in the future to process the updates much faster than this, without affecting the much larger Replicator system.

"Reliably" means that in the great majority of cases, the ITRs and Query Servers receive every mapping change as expected, but that in the relatively rare event of this being impossible due to packet loss, that the device can recover from this situation within one or at the most two seconds by requesting a copy of the packet from a remote server. Reliability also involves robustness against DoS attacks. This can never be completely protected against for any device on the open Internet, since its link(s) can easily be flooded by packets sent from botnets etc.

"Securely" means that each full database ITR and Query Server which receives the updates will be able to instantly verify that the updates are genuine, rather than the result of an attacker who might, for instance, send forged packets to that device or to some other part of the fast push system.

The mapping change command, as received by the ITR or Query Server, consists, as noted above, of a starting address and length specification of the micronet, followed by the address of the ETR. A zero for the ETR address indicates the ITR should drop the packets. Multiple mapping updates would be embodied in a datastream providing suitable context for a stream of such updates for IPv4, with a separate set of packets probably handling another, similar, type of mapping information for IPv6. The data format needs to provide for open-ended extensions in the future and to support authentication at the time of reception.

The mapping change command, as sent by the end-user, or by some other organisation or device which has the end-user's credentials, would involve the length of the micronet being checked to ensure it is the same as the currently configured length of the micronet which starts

at that location. The end-user's command might be part of an encrypted exchange involving a challenge-response protocol and the end-user's private key. Alternatively, an encrypted link could be used, such as via HTTPS, and a conventional username and password given as part of the command.

The end-user would previously have communicated directly or indirectly with their RUAS to configure their total assigned address space into one or more micronets. This ID concentrates on the changes to existing micronets. The ITR and Query Servers should reject change commands for micronets which overlap previously defined micronets which had a non-zero ETR value. So to the ITR or Query Server, a micronet mapped to zero can be remapped in whole or in part to any address, including zero, or can become part of another encompassing micronet mapped to any address. Micronets which are currently mapped to a non-zero address can only have their mapping changed for the entire micronet.

From this it can be seen that the ITRs and Query Servers perform minimal sanity checking on the mapping changes they receive, once they have been authenticated. A considerable level of sanity checking is therefore to be performed in each RUAS - for instance to ensure that micronets are never mapped to an address which is part of any micronet. (In LISP terminology: "the ETR address must be an RLOC".) There may also be additional lists of addresses which all RUASes are prohibited from using as ETR addresses.

RUASes and the multiple servers of the Launch system are few in number and will be administered carefully, so this ID does not consider automated aids to their management and debugging. However, the Replicators will be numerous and operated by a wide range of organisations. It is a goal of this proposal to maximise the degree to which this network can be robustly and easily managed, rather than requiring a great deal of manual configuration etc. This goal is discussed addressed in the current ID, but is for future work.

In order to debug the way the Ivip system is used, such as transient erroneous or malicious mapping updates which cause packets to be tunnelled to addresses where they are not welcome, there will need to be a system which monitors all mapping changes and keeps a lasting record of them. Then, aggrieved parties can search such a system for the address on which the received the unwanted packets, and so determine the micronet involved. This enables the aggrieved party to complain to the RUAS which is responsible for that micronet. This "mapping history" function could be performed by one or multiple separate systems, each simply taking a feed from the Replicator system. Something like this needs to exist for all map-encap schemes. This is not pursued in greater detail in the current Ivip

IDs.

5.2. Non-goals

Apart from checking the ETR address against any specific exclusion lists (such as specific prefixes, private and multicast space) and to ensure it is not part of a Mapped Address Block (MAB - a BGP advertised prefix containing micronets), the entire Ivip system takes no interest in whether there is a device at that address, whether the address is advertised in BGP, whether there is or was an ETR at that address, whether the ETR is reachable or whether the ETR can deliver packets to the micronet's destination device.

These are all matters which fall under the responsibility of the micronet's end user.

It is not a goal of the system to keep mapping changes secret from any party. This would be impossible. Therefore, it cannot be a goal of this or probably any map-encap scheme that in a mobile setting, the movement of an individual's device from one network to another could not be inferred by anyone who monitors the mapping updates. Consequently, there are fundamental privacy and security limitations to the use of this new form of address space. End users who want or need to keep their physical location secret will need to make other arrangements than direct reliance on Ivip.

Query Servers will issue map replies with a caching time of their own choosing. It is not a goal of the fast push system to allow end-users to affect that caching time. This reduces the amount of data in each update, and enables operators of Query Servers to use their own rules or algorithms to optimise the various costs and benefits of longer or shorter caching times in their own network. The longer the caching time the less often the Query Server will be queried about a particular micronet, but the longer it must send notifications for to any ITR which made such a query. Long caching times may burden the memory of ITRs which handle many micronets, and the proliferation of P2P traffic means that ITRs will often be handling packets addressed to a broadly scattered set of micronets.

As part of handling PMTUD and Fragmentation, ITRs may discover that an ETR to which they are attempting to tunnel packets is unreachable. There is no provision in the current Ivip proposal for this to be communicated back to other ITRs or to the RUASes. There could be some benefits to this if it could be done securely and so as not to allow DoS attacks, but in the current proposal, it is the sole responsibility of the end-user to determine that the ETR selected is reachable. This could be achieved quite well by hiring the services of a widely distributed monitoring service, with servers at many

physical and topological locations in the Net. These servers tunnel packets to the ETR, just as an ITR would, so they are sent to the destination network, where some process reports their arrival to the monitoring system. This could be a good area for IETF engineering work, but is not part of the current proposal.

Replicators perform a best-effort copying of mapping update packets. They do not store these packets for any appreciable time or attempt to request a packet in the sequence which is missing from their two or more input streams.

5.3. Challenges

There are obvious challenges building a global network which is distributed, to avoid any single point of failure whilst also being highly reliable, coordinated and secure. For this network to propagate information from one of many input points to a very large number (potentially millions) of endpoints, with very low levels of loss, is a further challenge on the open Internet.

The Replicator system needs to operate on the open Internet, as do the end-users' methods of interaction with the RUASes, directly or indirectly. However the RUASes, the Launch servers and the level 1 Replicators are probably best connected using private network links.

The closest existing technology to what is required may be Reliable Multicast, but this is optimised for long block lengths. This technology should be considered in greater depth as an alternative to what is proposed here, but the rest of this ID is based on the assumption that novel techniques are required.

Building a new, moment-to-moment crucial, architectural structure into the Internet is a serious undertaking, and conservative approaches using established techniques have obvious advantages because the component protocols are already implemented and well known. Assuming no such techniques can do the job, it is a challenge to devise some new techniques which RRG members will confidently assess as being capable of robust implementation, without significant risk of the design later being found to have fundamental flaws.

Every map-encap scheme faces challenges in convincing first the RRG, then the IESG, that the proposed architecture is necessary, desirable and better than all alternatives. Assuming the proposal is developed to the point of becoming Standards Track RFCs, the proposal needs to be enthusiastically adopted by ISPs and end-users of all sizes. A proposal which relies for its adoption on notions of impending doom if not adopted, or on coercion, cajoling or appeals to benevolence is not going to be widely adopted. The future map-encap scheme needs to

be very widely adopted in order to solve the immediate problem of routing scaling, and to make a serious contribution towards better utilization of IPv4 address space.

Ivip's difficulties in this respect will hopefully be fewer than those of competing schemes, because money can probably be made from the outset not just by renting out micronet space for multihomed end-users of all sizes, but from using the same techniques, plus a global network of TTRs, for the new approach to mobility.

Internet history is littered with ambitious protocols and business ventures which never delivered. Ivip, or any other map-encap scheme, will need broad support from ISPs, end-users and RIRs before it can be widely adopted. Hopefully, fast push will be widely regarded as both practical and desirable.

6. Definition of Terms

6.1. RLOC address space

Borrowing LISP's Routing Locator term, RLOC describes any address or range of addresses in which packets are delivered to the destination via conventional BGP routing mechanisms. All BGP advertised address space today is RLOC space.

6.2. Mapped address space

Once Ivip is operational, a growing subset of the total space used will be handled by ITRs tunnelling the packets to an ETR, which delivers the packets to the destination. As such, this address space is "mapped" by the Ivip map-encap scheme. Therefore, it can be divided into smaller sections than is possible with BGP (256 granularity for IPv4, due to restrictions on lengths of advertised routes) and each such section can be used via any ETR in the world.

6.3. MAB - Mapped Address Block

A MAB is a BGP advertised prefix which is Mapped address space rather than RLOC space. ITRs all over the Net advertise this prefix, tunnelling the packets to ETRs according to the current mapping for the destination address of each packet.

A MAB could, in principle, be as large as a /8. Larger MABs are preferred in general, because each one burdens the BGP system with only a single advertisement, but includes the Mapped address space of many end-users. However, for reasons discussed below - including load sharing between ITRs and ease of initially loading snapshots of the mapping database - it may be best if MABs are more typically in the /12 to /17 range.

6.4. UAB - User Address Block

Each MAB typically contains address space which has been assigned by some means to many (perhaps tens of thousands) separate end-users. A UAB is a contiguous range of addresses within a MAB which is assigned to one end-user.

A MAB could be assigned entirely to one end-user - as might be the case if the end-user converted a prefix of theirs which was previously conventional RLOC space to be managed by the Ivip system. Generally speaking, MABs are ideally large (short prefixes) and each contains space for multiple end-users. An end-user might have multiple UABs in a MAB, but for simplicity is assumed each has a single UAB. UABs are specified by starting address and length - they

need not be on power of two boundaries.

UABs are important constructs for the entities which control the mapping information, but are not seen or used by ITRs or the fast push mapping distribution system.

6.5. Micronet

Following Bill Herrin's suggestion, the term "micronet" refers to a range of Mapped address space for which all addresses have the same mapping. In LISP and APT, these are known as EID prefixes. In Ivip, a micronet need not be on binary boundaries - it is specified by a starting address and a length, in units of single IPv4 addresses or IPv6 /64 prefixes.

An end-user could use their entire UAB as a single micronet, or they could split it into as many micronets as they wish, and change these divisions dynamically.

Any micronet which is mapped to address zero will cause ITRs to drop packets addressed to this micronet. A micronet can be defined within the whole or part of a contiguous range of address space which is currently mapped to zero, by the fast push mapping distribution system carrying an update message specifying the new micronet's starting address, its length, and a non-zero address for its mapping.

6.6. RUAS - Root Update Authorisation System

Multiple RUASes collectively generate the total stream of mapping update messages. Each RUAS is responsible for one or more MABs. There may be a dozen to perhaps a hundred RUASes. End-users with Mapped address space have an arrangement either directly with the RUAS which handles the MAB their space is located within, or indirectly through an organisation such as a UAS.

6.7. UAS - Update Authorisation System

A UAS is the system of an organisation which accepts mapping change commands from end-users, and conveys them directly - or perhaps indirectly via another UAS - to the RUAS which handles the relevant MAB. An RUAS which accepts mapping update commands from end-users does so via its own UAS system.

A UAS accepts upstream input from end-users and/or other UASes. It generates output to downstream RUASes and/or other UASes. One UAS may have relationships with multiple RUASes. A MAB may be assigned to an RUAS and control of parts of this may be delegated to multiple UASes. A single UAS may work only with a single RUAS, or with

multiple and perhaps all RUASes.

Whether the MAB itself is administratively assigned (by an RIR, or some national Internet Registry) to the UAS or to the RUAS is not important in a technical sense. End-users will choose address space according to the RUAS (and any UASes) it depends upon with care, because the reliability of this MAB's address space will forever be dependent on these organisations.

The number of RUASes will be limited to enable them to efficiently and reliably work together to create a single stream of updates for the entire Ivip system. The ability of UASes to act as agents for RUASes and/or to have their own MABs which they contract a RUAS to handle the mapping for, enables a large number of organisations to compete in the sale/rent of Mapped address space.

6.8. UMUC - User Mapping Update Command

A UMUC is whatever action the end-user performs on one or more different user-interfaces of whatever UAS they use to change the mapping of their one or more micronets. The system would also be able to tell the user the current mapping and also confirm that a requested change to the mapping was acceptable address.

For instance, the system would generate an error if the mapping was to a disallowed address - multicast, Mapped address space, private address space or to some other prefixes which the Ivip system does not support the tunnelling of packets. Similarly, and error would be generated if the end-user attempted to change the mapping for some address space outside their UAB, or if they defined a new micronet within that space with non-zero mapping, which overlapped some addresses for which the mapping was currently non-zero.

For the sake of discussion, it will be assumed that all UMACs have passed these basic sanity tests at the UAS and are for valid mapping addresses - so a UMAC is a successfully accepted update command from the end-user, or some person or system or with the end-user's credentials.

There could be many methods by which this command is communicated, including HTTPS web forms with username and password authentication. Challenge response SSL sessions might be more suitable for automated mapping change systems, such as a multihoming monitoring system which the end-user authorises to control the mapping of some or all of their UAB.

In addition to authentication, the command takes the form of the starting address of the micronet, the length of the micronet, and a

single IP address to which this micronet will have its mapping changed to.

6.9. SUMUC - Signed User Mapping Update Command

This is the information contained in a UMUC, signed by the UAS which accepted it from the user (or by some other UAS), being handed down the tree to another UAS or to the RUAS of the tree, so that the recipient UAS/RUAS can verify the signature and regard the UMUC as authoritative.

6.10. MABUS - Update Stream specific to one MAB

This is a stream of data by which the real-time updates to the mapping data for any one IMAB are conveyed. For the purposes of discussion, the RUASes and the Launch system are assumed to work in a synchronized fashion, generating a body of updates for each MAB once a second. (Probably the case of no updates will be codified specifically in the update stream, rather than just resulting in no mention of the MAB.)

Each RUAS will generate one MABUS for each of its MABs. So each second, the RUASes collectively generate a variable length body of update information for every MAB in the Ivip system.

The MABUS consist primarily of mapping updates: micronet starting address, length and mapping address. These are all covered by a common authentication system for this MAB, so that ITRDs and QSDs can verify that the updates are genuine.

The MABUS also periodically contains other messages for the ITRDs and QSDs. At present, the only such message is to the effect that at the snapshot of the mapping database for this MAB has been made, and is available with a particular filename from multiple servers

The RUASes work together with the Launch system and the Replicator network to deliver every one second body of the MABUS, for every MAB, to every ITRD and QSD in the Net.

6.11. Launch server

A small (such as 8) number of widely dispersed Launch servers are operated by the RUASes and work together to generate, every second, multiple identical streams of packets to Replicators in the first level (1) of the Replicator system. The Launch server receives its input in the previous second from the RUASes.

6.12. Replicator

A cross-linked, tree-like, system of Replicators form a redundant, reliable, high-speed distribution system for delivering mapping updates to full database ITRs and Query Servers all over the Net.

Each Replicator receives one or more (typically two) streams of update packets from an upstream Replicator or Launch server. These two source streams should come from widely topologically separated sources, ideally over two separate physical links. For instance a Replicator in Berlin might receive its update streams from London and Berlin, two sources in Berlin which are in different ISP networks, or in any combination which minimises the likelihood that both sources will be disrupted by any one fault.

The Replicator identifies the packets in each input stream by a simple sequence number in the start of the payload. It expects a particular set of packet numbers, and for each number, the first packet to arrive is replicated to its multiple output streams.

In this way, unless the same numbered packet is lost from both input streams, each Replicator receives the full set of mapping update packets for this second, and sends them to tens or perhaps hundreds of downstream devices, which are other Replicators, or full database ITRs and Query Servers.

The receive and send links use UDP packets which are encrypted separately for each link, as discussed below. This prevents an attacker from spoofing these packets and so altering the behavior of ITRs.

Replicators could be implemented in routers, but are probably best implemented in ordinary software on a GNU-Linux/BSD etc. server. They do not cache information and they don't need hard drive storage. A full database ITRD or Query Server could also operate as a Replicator.

6.13. QSD - Query Server with full Database

Like ITRDs, QSDs get a full feed of updates from one or more Replicators. Like ITRDs, when they boot, they download individual snapshot files for each MAB in the Ivip system. This is discussed further in a later section. Query Servers, ITRs and ETRs will be are discussed in greater detail in future Ivip IDs, and are discussed in ivip-arch-01.

QSDs respond immediately to queries from nearby caching ITRs and from caching Query Servers - and send notifications to these if mapping

data changes for a micronet which was the subject of a recent query.

QSDs have no routing or traffic handling functions. They need a lot of memory, so the best way to implement a QSD is probably on an ordinary server with one or more gigabit Ethernet interfaces. No hard drive is required, except perhaps for logging purposes. A QSD could be integrated with a Replicator function, and perhaps an ITRD function - or for that matter an ETR function too.

6.14. QSC - Query Server with Cache

A QSC could be implemented in a router. It does not route packets, but its memory and computational requirements are likely to be modest compared to those of a QSD. There is no need for a full feed of updates from the Replicator system. However, each QSD must be able to get mapping information from one or more upstream QSDs - or perhaps via QSCs which themselves access upstream QSDs.

The easiest way to implement this would be software on a modest server, which would only need a hard drive for logging purposes.

6.15. ITR - Ingress Tunnel Router

"ITR" is a general term for a router or server which accepts packets with Destination Address = a Mapped address (that is, an address managed by Ivip, and not delivered directly by conventional BGP routers). The ITR determines the mapping for the micronet which encompasses the destination address, and encapsulates the packet with an outer header, to that address - where it will presumably be decapsulated by an ETR.

ITRs need not be located on RLOC addresses. However, it is likely that the larger ITRs will be. ITRs can be on Mapped addresses, but cannot be behind NAT.

6.16. ITRD - Ingress Tunnel Router with full Database

An ITRD is an ITR with a full copy of the current mapping database. When it boots, it downloads snapshots and then brings the data up-to-date, and maintains it in this state, with updates received from one - or ideally two or more - Replicators.

Consequently, an ITRD is able to tunnel every packet addressed to Mapped address space to the appropriate ETR.

ITRDs can be implemented in a suitable router with lots of RAM, CPU power and high capacity dedicated FIB hardware. Lower traffic rates could be handled by a suitably powerful server, without any hardware

FIB.

An ITRD might also implement the Replicator, QSD and/or ETR functions.

6.17. ITRC - Ingress Tunnel Router with Cache

An ITR without a full copy of the mapping database - and so not requiring a constant stream of updates from one or more Replicators.

The ITRC gains mapping information from a nearby QSD, perhaps via one or more intermediate QSCs. It may buffer every packet it needs to map, but is awaiting mapping information for, until it requests and receives mapping information. Since the QSD is local (within metres, kilometres or at most a few hundred km), the maximum buffering time should be milliseconds or tens of milliseconds. Subsequent packets can be tunnelled immediately. Alternatively, rather than buffering the packet, it may be passed on to where it will enter a full database ITR, or perhaps another ITRC which already has the mapping information for the relevant micronet.

Like an ITRD, an ITRC could be implemented in a conventional router with high-speed FIB - assuming the FIB is capable of the tunnelling function - or in a server without any specialised FIB hardware. While an ITRD requires large memory capacity and a constant stream of updates from two or more Replicators, an ITRC requires memory only according to the number of micronets for which it is currently handling traffic. This makes the ITRC function much more practical to implement in "hardware routers", which have generally smaller and more expensive memories than whatever is possible with commonplace PC-like servers.

An ITRC might also implement the QSC and/or ETR function.

6.18. ITFH - Ingress Tunneling Function in Host

A host which is not behind a NAT could have additional software in its TCP/IP stack to perform the ITRC functions described above. It needs a good link to a nearby QSD/QSC system - so this would not be suitable over a dialup modem or radio link.

Host software, CPU power and RAM is generally free of incremental cost in this setting. This would greatly reduce the load on any ITRCs and perhaps ITRDs in the rest of the network. An ITFH function would be desirable in every web server in a hosting company, assuming the servers had sufficient CPU and RAM resources.

A host performing NAT functions for some hosts on a private network

is a good place to implement ITFH, as long as this host is not behind NAT itself. The most common NAT situation is a DSL or cable modem or an optical home/SOHO adaptor. Technically these are routers, but they are inexpensive and purely software based, and therefore might be thought of as "hosts".

ITRCs and ITFHs could be overwhelmed by a large number of different micronets inside the caching period, so they need to be able to drop old cached mapping data when their RAM or FIB can't handle it. Then, they need to be in a network position where an upstream ITRD will always find the packets they emit which they cannot encapsulate. With Ivip, this is always the case, depending on how congested the nearest "anycast ITR in the DFZ" is.

6.19. ETR - Egress Tunnel Router

An ETR is a router or a server which receives encapsulated packets on one of its one or more RLOC addresses, strips off the outer IP header, copying its hop-count to the internal packet, and then by some means ensures the resulting packet is delivered to the destination host or network.

Unlike in other schemes, Ivip ETRs are not involved in reachability testing by ITRs. However ITRs need to do some probing for PMTUD and Fragmentation management purposes. ETRs will also generally need to respond to probing by other systems such as a multihoming management system, which is independent of the Ivip system, and which decides how mapping for a micronet should be changed to ensure continued service via alternative ETRs.

6.20. TTR - Translating Tunnel Router for Mobile-IP

A TTR behaves, in part, as an ETR - a device with an RLOC address to which packets are tunnelled so that they will be decapsulated and delivered to the destination host or network, which in this case is a Mobile Node (MN). The MN establishes a two-way tunnel to the TTR from its care-of address, which can be behind NAT. The MN may have such tunnels to other TTRs, including via different edge networks.

A TTR is also a means by which the MN can send packet out to the Internet at large. The TTR may simply emit the packets, or may integrate an ITRD or ITRC function within itself.

7. Update Authorities and User Interfaces

We now commence a detailed discussion of the fast push mapping distribution system itself, starting with the systems which accept commands from end-users (or their authorised representatives or systems) and prepare the information for the Launch system.

This is the early stage of an ambitious design, so a number of options are contemplated.

The final authority to control mapping information is fully devolved to end-users, who by means of a username and password or some other authentication method, are able to issue commands to define micronets within their UAS, and to map each micronet to any ETR.

However the physical authority to control the mapping of all Mapped space within a single MAB rests with a single RUAS. That RUAS may be acting for a UAS who is the assignee of the MAB. The RUAS may be the assignee and may delegate control to one or more UASes. The RUAS may have relationships directly to the end-users of this MAB, through its own UAS. Here we discuss the flow of information and trust between these various entities, in real-time, so that every second (for example, the actual time period will need to be carefully considered) each RUAS assembles a body of update information for each of its MABs.

In the diagrams below, each RUAS or UAS is depicted as a single entity. Each such entity acts as a single functional block, but will typically be implemented as a redundant system over several servers.

7.1. RUAS Outputs

7.1.1. Updates every second

Every second, for each MAB the RUAS is authoritative for, the RUAS generates a set of mapping updates, and works with other RUASes to integrate this into the next second's output from the Launch system.

As previously mentioned, these updates are primarily actual mapping updates for individual micronets within the MAB, but also contain occasional messages to the effect that a snapshot of this MAB's full mapping database has been made and is, or soon will be, available via various servers.

7.1.2. MAB snapshots

Every few minutes (or some other time period, as chosen by the RUAS, but with some reasonable maximum defined by a BCP) the RUAS makes a

copy of the complete mapping information for a MAB. Snapshots for each MAB are independent of each other, and so can be done with different frequencies.

The snapshot is in a format which needs to be standardized, so it can be downloaded and understood by any ITRD or QSD, now and in the future. This data format needs to be extensible to cover new kinds of mapping information and other functions not yet anticipated - which will be ignored by devices which are not capable of these functions.

The exact format for this is for future work, but for instance would begin with some identifying information about the MAB, a block defining that the following data concerns IPv4 micronet mapping information (and snapshot announcements), with the possibility of other blocks containing different kinds of data. Binary format would probably be best, and the file could be gzipped for distribution.

Each such file will be given a distinctive name, according to a standardised format, which indicates at least the MAB starting address and length, and the time of the snapshot.

The snapshot process will take a second or two to complete from the time it is initiated, and the resulting file will be copied to a number of servers, ideally located in a variety of locations around the Net.

Each such server would be run by the RUAS directly, or as part of all RUASes working together. The servers can probably be conventional HTTP servers, so that ITRDs and QSDs can download the snapshots when needed. There is scope for some careful design with DNS so that there is an automatic structure in the domain names of these servers, enabling an expandable system to be automatically used by ITRDs and QSDs without manual configuration.

These files will be publicly available, and need to be made available for somewhat longer than the cycle time of snapshots. So with a ten minute snapshot cycle, the previous snapshot should be available for a while - probably 10 minutes or so - after the new one is available.

Snapshots are downloaded by ITRDs and QSDs when they boot, and if they suffer a disruption in mapping updates which necessitates a reload of this part of the complete mapping database. To facilitate this, MABs should not be too large - or at least contain so many micronets - as to make individual snapshot files excessively large.

At boot time, or when resynching, the ITRD or QSD will monitor the update streams for each MAB until a snapshot announcement is found.

It will then buffer all subsequent updates and download the snapshot as soon as it is available. Once the snapshot has arrived, and been unpacked to RAM, the buffered updates are applied to it. Then, this MAB's part of the mapping database is up-to-date and the ITR can begin advertising this MAB, and therefore tunnelling all packets which are addressed to this MAB.

In order to reduce total path lengths, it would be desirable if an ITRD or QSD in a given location could access a nearby snapshot server. It may be desirable to have every snapshot of ever MAB in a single server, or a single set of servers which are accessed by geographically close ITRDs and QSCs. Anycast is not a good technology for this, since file retrieval is best done via TCP sessions. The ITR system itself can't be used, to avoid circular dependencies - so the servers must be on RLOC addresses. Likewise, any DNS servers involved in this server system need to be strictly on RLOC addresses.

Each ITRD or QSD needs to be configured with, or to automatically discover, two or more such servers which are relatively close, so the data can be found despite one server being down.

Perhaps these servers could be identified in a carefully structured DNS hierarchy:

xxxxx.yyyy.ipv4.ivipservers.net

Where xxxxx is one of an extendable list of localities and where yyyy uniquely identifies the RUAS. If snapshots from all RUASes were pooled into a single server, the latter would not be necessary. However, it may be better to let each RUAS run its own network of servers, which may involve a choice to use the same servers in some or many instances as are used by other RUASes.

Initially, an RUAS may have a single update server for Australia, and some standardised list of xxxxx locations defines "au" as being the value to be used by any ITRD or QSD which seeks this RUASes server which is closest to Australia. Later, the list could be extended for more specific locations, such as "syd-au", "mel-au" etc. Then, every RUAS would need to generate DNS entries for these as well, and point them to whatever server was appropriate. In the event they had no server in Melbourne, they could make that FQDN resolve to the same IP address as their only Australian server, in Sydney.

From the point of view of the ITRD or QSC, seeking an update for a given MAB of a particular RUAS, the address to request the file from could be made up from the RUAS identifier yyyy which is contained in the snapshot announcement (in the stream of mapping updates),

concatenated with a locally configured "xxxxx" and "ipv4.ivipservers.net". In the event that this server was unavailable one or more locally configured alternatives to this initial "xxxxx" value could be tried - including one or more for nearby countries.

The most significant 24 bits of the MAB's starting address (probably 48 bits for IPv6, assuming this is the granularity of BGP advertisements) for would be transformed into a text string such as 150.101.072. A similar transformation of the precise time of the snapshot would result in a second text string, and these would be used to reliably identify the appropriate directory and file in the server.

7.1.3. Missing packet servers

The cross-linked tree-structured Launch and Replicator systems should provide a robust method of delivering the complete set of MAB updates every second, to every ITRD and QSD. There may be more subtle and efficient methods than this somewhat brute-force approach, which involves typically a doubling of the amount of update traffic in the pursuit of robustness. However, the rate of updates will only be problematic by current standards at a date so far in the future that the technology of the day will render the task far less daunting that it would now be.

In the event that an ITRD or QSD misses one or more packets, it will be able to easily identify which are missing, due to the sequence numbers built into their payloads. This will transform easily into an address to use by which the missing one or more packets can be retrieved, probably via HTTP. Similar arrangements - probably the same servers to those just mentioned - would be used to locate the missing packet and download it.

7.2. Authentication of RUAS-generated data

Careful consideration must be given to how ITRDs and ITRCs can quickly and reliably ensure that the information they receive ostensibly from each RUAS is genuine. At this early stage of development, the model is pretty simple.

7.2.1. Snapshot and missing packet files

Each RUAS has a key pair and signs the MAB snapshot and missing packet files. ITRDs and ITRCs can verify the signature by reference to certificates signed by some higher authority, or by some alternative arrangements.

Both these types of files are only handled occasionally, so the overhead in performing crypto operations is insignificant.

7.2.2. Mapping updates

This principle does not apply to the update information contained in packets received from the Replicator system. It would be onerous to individually authenticate each packet, or each body of updates from each RUAS contained in potentially multiple packets. Instead, at the current early stage of development, a different model is proposed. No doubt this can be improved upon.

The Launch system servers will receive signed information, each second, from all the RUASes. Only when all such servers agree that the information they received is authenticated will any of them send that RUAS's updates to the Replicator network.

The first level (1) of the Replicator network involves manually configured, encrypted, links to Launch servers, with each Replicator receiving a full stream of update packets from two or more widely distributed Launch servers. Those links will involve encrypted UDP packets so that each stream can be known to have originated at a specific Launch server. The destination device will establish the encrypted link with the source device.

It is proposed that the subsequent levels of Replicators use the same techniques, so that there is implicit trust in the data received from the two (or perhaps more) upstream Replicators. This would be a fragile arrangement with a single upstream source, but since there are two sources, with identical contents, it will be a simple matter in each Replicator to detect a condition in which one stream differs from another. That will not prove which stream is correct, but it would be enough to show that an attacker has gained control of one upstream Replicator - enabling the current Replicator to shut down and so not propagate bogus mapping information.

Loss of a single Replicator will generally not affect the reliable delivery of updates, due to the cross-linked nature of the network. However, there remains a chance that an attacker's packet could be replicated all the way to an ITRD or QSD. There, it could cause traffic packets to be tunnelled to the attacker's chosen location.

One approach to preventing this is to have each ITRD and QSD authenticate every packet, or multi-packet body of update information, from each RUAS, by each packet carrying a digital signature. This seems expensive, but perhaps it would be practical.

Another approach would be to have the Launch system add one or more

packets to the stream, containing MD5 (or some other hash function) "checksums" of either each packet, or each body of update information from each RUAS. It would be trivial to have a checksum for the entire second's worth of updates, but then a single missing packet would make it impossible to check the rest.

The MD5 checksums could be sent twice, for robustness, and some care would be needed in deciding on their granularity. A separate checksum for every packet would be conceptually simple and enable individual packets to be accepted immediately, even if another packet was not received and so required a "missing packet" request. However, this increases the number of MD5 checksums to transmit.

The current proposal is to have an MD5 checksum for each MAB for which updates are received, which may be less than a packet, or perhaps more.

7.3. RUAS - UAS interconnection

This section depicts a single tree of delegated responsibility for the user control of mapping of one MAB. The Root UAS at the base of the tree is run by Company X - RUAS-X. RUAS-X could be authoritative for other MABs, and each such tree of delegation may have the same set of other UAS systems, or it could be different. Each delegation tree is separate from the delegation trees of other MABs, even if they look similar, because the tree includes specific subsets of the whole MAB address range as one of the defining characteristics of its branches and leaves.

The initial action which leads to the database being changed is a user generated (manually or by the user's equipment or by a system authorised by the user) UMUC (User Mapping Update Command).

For authorising and feeding UMUCs to the RUAS-X, there is a tree as depicted in Figure 1. Delegation of authority flows up the tree as the total address range of the MAB is split at each branching junction. This tree structure involves data, in the form of SUMUCs (Signed User Mapping Updated Commands) flowing down towards the root of the tree. (Data would also flow up the tree so each user-interface leaf could tell end-users what their current mapping was, could test their requests against constraints etc.) The idea is that RUAS-X could delegate control of one or more subsets of the MAB's total range of addresses to some other system, which in turn could delegate control to other systems. There would be no absolute limit on the height (usually called depth) of these hierarchies.

The servers which handle the end-user interaction needs to be one of the leaves of this tree structure, so as not to burden the RUAS-X

database servers themselves with details of user interaction. This enables various companies to give different kinds of control for the Mapping of the IP addresses their branch of the tree controls. Figure 1 does not show RUAS-X having any user interface servers, but it could. The simplest arrangement would be the RUAS having simply a user-interface server and no tree of other UASes.

There would need to be IETF standardised methods by which some server could execute a UMAC with the user-interface servers of any of these UASes. This standardisation would be especially important for multihoming, because some reasonably trusted company could run an automated monitoring system, and have the credentials (username, password, key etc.) stored in their system so their system can change the mapping of one or more micronets the moment one link was detected to be faulty. Also, the company (such as X, Y or Z in Figure 1) which controls a particular range of the Mapped space may offer such a multihoming monitoring system itself.

The tree in this example controls an MAB with the address range 20.0.0.0 to 20.3.255.255. In this example, company X has been assigned by an RIR the entire range 20.0.0.0 to 20.3.255.255. Company X sublets to Y a quarter of this: 20.1.0.0 to 20.1.255.255. These divisions are on binary boundaries, but they need not be. It would be just as possible for X to delegate to Y an arbitrary subset of the whole range, or the entire range - or just one IPv4 address or IPv6 /64.

X's Root Update Authorisation Server (RUAS) has a private key for signing all the MAB snapshot files it periodically creates and makes available.

In this example, company Y delegates control of some of its space to company Z, and Z has an end-user U, who needs to control the mapping of a UAB containing one or more micronets in Z's range.

Z has various interfaces by which U can do this, with its own arrangements for authentication, for monitoring a multihoming system and making changes automatically etc. Ideally there might be one or more automated, host-to-server, IETF-standardised protocols so all end users could have standardised software for talking to whichever company's servers they use to control the mapping of their IP address(es).

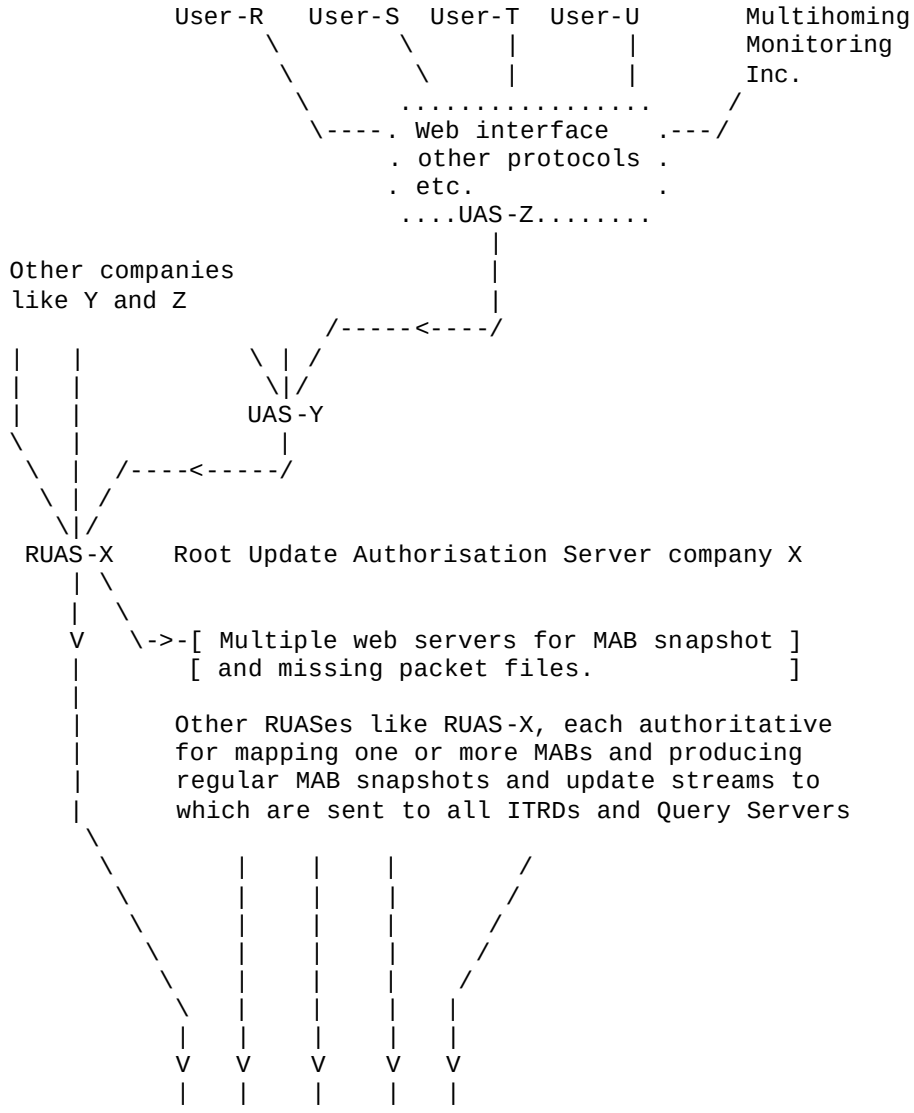


Figure 1: Delegation tree of UASes above one RUAS.

When user-U (or a device or system with user-U's credentials) changes the mapping of their micronet via a web interface this is achieved via Z's website, authenticating him-, her- or it-self, by whatever means Z requires. This causes UAS-Z to generate a signed copy of this update command (a SUMUC) and to send it to UAS-Y.

The SUMUC consists of three items (assuming IPv4 for simplicity): A starting address for which micronet this update covers, a range (>=1), and a new mapping value (ETR address), which will also be a 32 bit integer. The SUMAC could also consist of a time in the future the update should be executed.

UAS-Y trusts this SUMUC because it can authenticate UAS-Z's signature. It strips off the signature and adds its own, before passing the SUMUC down to the next level: RUAS-X.

RUAS-X likewise has a copy of UAS-Y's public key and within a fraction of a second of U initiating the UMUC, the master copy of this MAB's database, in RUAS-X is altered accordingly. (This would be a distributed, redundant, database system.)

Authority is delegated up the tree, because UAS-Y will only accept update commands if they are signed by one of its branch UASes, and for the particular address range that UAS has been authorised to control.

User-U may have given their username and password etc. to Multihoming Monitoring Inc. so this company can monitor their multihoming links and change the mapping as soon as one link goes down. UAS-Z doesn't know or care who actually makes the change - as long as they can authenticate themselves for whatever micronet they want to change the mapping of.

8. The Launch system

In this discussion 8 Launch servers will be assumed. The exact number could be varied over time. Initial introduction could no-doubt be done with a simpler system, but the purpose of this discussion is to explore how a the system could scale to very large numbers of micronets and updates per second.

The exact logic of the Launch system remains to be determined. The following is a rough guide to how it might be done.

The task of the Launch system is every cycle - in this example every second - to collate the update information from all the RUASes, agree on what has been collected, and then to generate multiple streams of packets containing that information, from multiple locations, to the widely geographically dispersed level 1 Replicators. Links between the Launch servers might best be done via private links to avoid packet flooding attacks. Likewise the links to level 1 Replicators.

Each Launch server has a link to every other Launch server, and every RUAS has a link to every Launch server. This may seem rather over-engineered, but the system will be robust in the event of failure of quite a few of these links, and the task at hand is a momentous one, deserving considerable effort to make it fast and reliable.

The exact details of how packets are handled, information combined into packets etc. remains for future work.

Each Launch server may be a single physical server, with a live backup at the same address, or a redundant cluster of servers which behaves as one.

While the Launch servers are sending out the update packets for one second, they are comparing notes about updates to be sent in the next second and collecting updates to be sent in the second after that. Perhaps this one second timing clock will prove to be too ambitious, or the operations may be broken into four phases, rather than three.

8.1. Phase 1 - collecting updates from RUASes

In phase 1, all RUASes attempt to send their complete set of updates to every Launch server, where they are buffered in readiness for Phase 2. The Launch server authenticates this information, by standard cryptographic means based on the public key of each RUAS.

The contents of each RUAS's updates are then collected, and an MD5 (or some other hash algorithm) checksum (actually a digest) is created for each one.

8.2. Phase 2 - checksum comparison

Each Launch server sends to every other Launch server its record of the checksums of the updates received from each RUAS.

This enables each Launch server to identify its state as one of the following:

- Normal: no received set of checksums includes updates from more or different RUASes than where received by this RUAS and all the checksums agree with the local values. Therefore, this Launch server has established that it correctly received the complete set of updates.

- Missing updates: One (maybe some higher figure) or more received lists contained checksums from an RUAS for which this Launch server did not correctly receive any updates. Therefore, this Launch server has established that it has missed out on updates from one or more RUASes.

- Invalid updates: The local checksum value for one or more RUAS sets of updates does not equate to two or more checksums from other Launch servers, which themselves are equal. The Launch server has established that it received an erroneous copy of at least one RUAS's set of updates.

Each Launch server now sends a signed message to the other Launch servers, containing the state determined above: Normal, invalid updates or missing updates.

Those Launch servers which are in the Normal state count how many others are also in this state. If the number is above some "quorum" constant, say 4 in an 8 server system, then each such Launch server is ready to send the collected updates in phase 3. These Launch servers independently process the same update data into a series of packets, with sequence numbers which can easily be identified by the recipient devices - initially level 1 Replicators. Those packets are stored, ready for transmission in phase 3.

Normally, all 8 Launch servers will receive the same information correctly, and so will participate in phase 3. The purpose of this constant is to ensure that there will not be a condition in which only one or two Launch servers participate in phase 3. The idea is that the updates will be launched into the Replicator network robustly, or not at all.

With further development work, it should be possible to fine-tune this system to adequately guard against single or multiple points of

failure, but also to ensure that the system only sends out data when it can send from at least three, or four, or some constant number of Launch servers. Careful analysis will be required to anticipate various failure modes.

RUASes monitor the output of the Launch system, and if a particular second's worth of updates are not sent, then the RUAS will send them again soon.

This raises some potential ordering difficulties, where one second contains a command to map a micronet to zero, and the next second contains a command to map part of it to some valid address. While these could be combined in the one second, if they were not, and the first second was not sent, then the second second's command would fail in the ITR, because it would be defining a new smaller micronet in part of a micronet which was not at the time mapped to zero. Further work required, but the RUAS can predict the problems which the ITR would have, and generate suitable updates to make the same results occur.

The above algorithm will need to be extended so that a flaky RUAS, which only transmits to a few Launch servers, will not cause the quorum test to fail, due for instance to two Launch servers getting its updates, and the rest recognising that they didn't.

8.3. Phase 3 - identical update streams

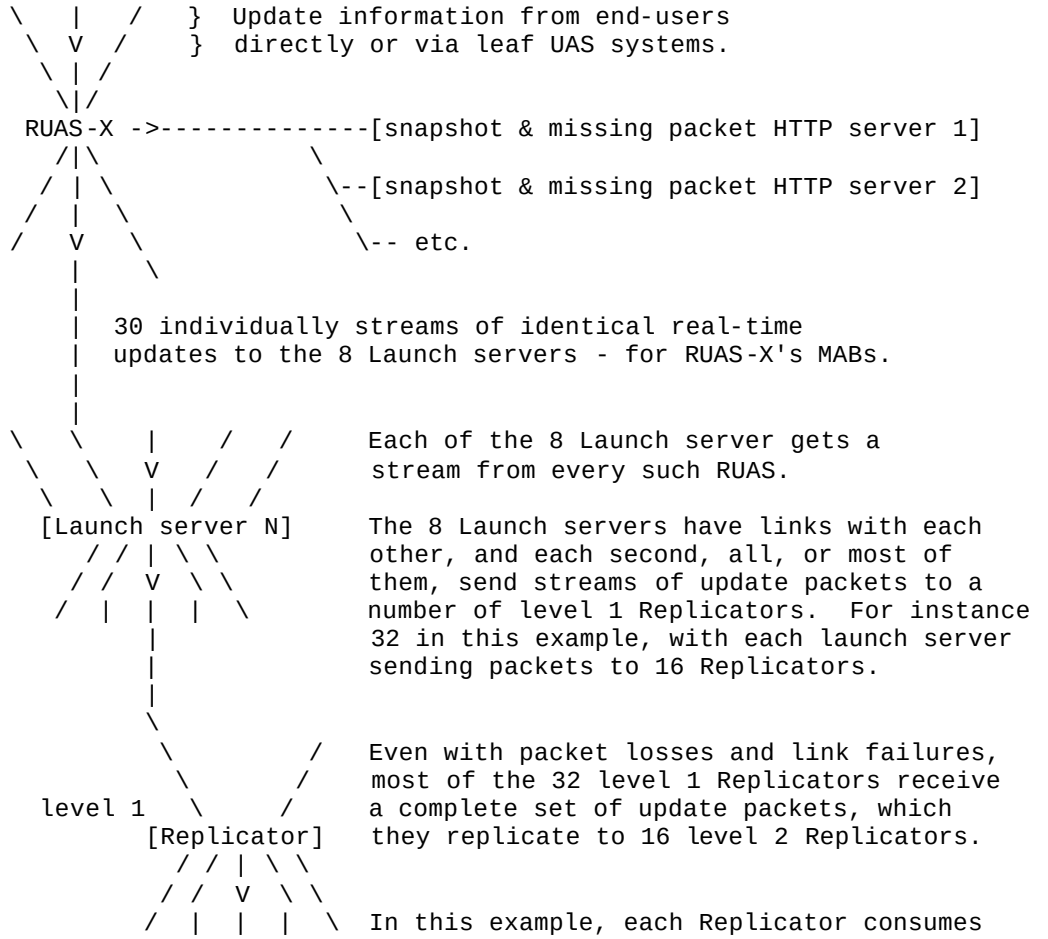
Those Launch servers which have the full set of update data now send the packets they generated, in separate encrypted streams, to level 1 Replicators. It would probably be best if the packets are sent in numeric sequence, with sending times decided to spread the packets over the whole second. Exactly how many level 1 Replicators there are, and how many are driven by each Launch server, will be a matter for further work.

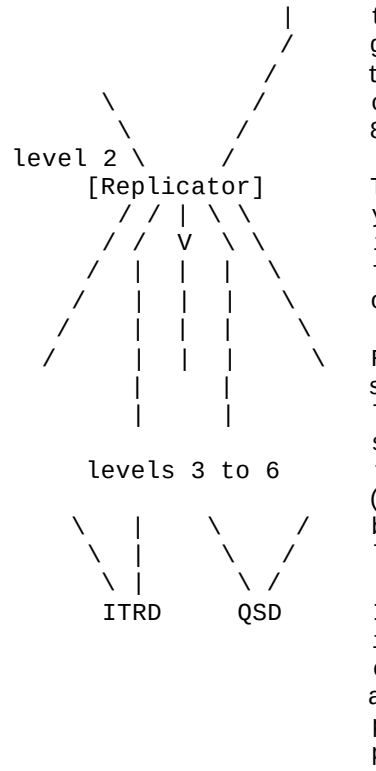
The result will be in each cycle that either the full set of updates are sent out, robustly, by all or almost all level 1 Replicators. Even if there is a relatively high packet loss from some or many of these, and some broken links, all, or almost all level 2 Replicators will receive a full set of packets.

9. Replicators

Further work is required to reach a more precise description of how the update information is placed in packets, and signed in such a way that ITRDs and QSDs can be sure they have received the correct information. If we assume that this problem can be solved, then the following description of the functionality of individual Replicators and the way they are arranged will lead to an understanding of how they form a robust, packet amplifying, global network for delivering the output of the Launch system to a million or more ITRDs and QSDs.

(See "Figure 2 Tree of UASes above one RUAS".)





two feeds from the upstream level, and generates 16 feeds to Replicators in the level below (numbered one above the current level). So each level involves 8 times the number of Replicators.

These figures might be typical of later years with a billion micronets, however in the first five or ten years, with fewer updates, the amplification ratio of each level could be much higher.

Replicators are cheap diskless Linux/BSD servers with one or two gigabit Ethernet links. They would ideally be located on stub connections to transit routers, though the Level 5 and 6 Replicators (32,000 and 128,000 respectively) might be at the border of, or inside, provider larger end-user networks.

ITRDs and QSDs get two or more ideally identical full feeds of updates - so occasional packets missing from one are no problem, since the other stream provides a packet with an identical payload.

Figure 2: Multiple levels of Replicators drive hundreds of thousands of ITRDs and QSDs.

9.1. Scaling limits

The Replicator system is scalable to any size simply by adding Replicators. Assuming two input streams for each Replicator, N output streams gives an N/2 amplification of stream numbers per level. N could be quite high in the early years of introduction, when the number of micronets and updates is small by comparison with the design target of one to ten billion micronets, with accompanying update rates driven by their use for handheld mobile devices.

First, a maximal IPv4 example will be considered. Assume a billion micronets, most of them for single IP addresses. Presumably most of these will be for individual end-users, at home or with mobile devices. The update rate will be relatively low for multihoming the home and office-based micronets, but the update rate for mobile devices could be much higher. Half a billion mobile micronets, each with an update every 3 hours, involves 47k updates a second, on

average. The raw data of each IPv4 mapping update is about 12 bytes, so adding 50% protocol overhead, this is 846k bytes a second - about 10Mbps on average. Peak data rates would be higher.

By the time such large update rates eventuate, Replicators based on commodity PCs will be able to handle such rates, and the bandwidth involved will not seem as frightening as it is today.

While a pure pull system can scale effortlessly to any number of micronets, with any rate of change to the mapping, it can't support mobility - which is the only reason there would ever be such large numbers of micronets or updates. Any initially "pure pull" system which could support mobility would require either short caching times and so massive volumes of queries and responses, or would require a "notification" system rivalling the fast push system described here.

IPv6 could theoretically involve tens of billions of micronets - and the mapping data would be more voluminous due to the long addresses involved. Still, a system based on principles such as described in this ID would be well placed to be the most scalable solution to the problem.

In a system such as this, there needs to be some financial charge for each update - which need not be so high as to deter the majority of end-users.

At some point, with extremely large numbers of micronets and updates, the fast push system would become unwieldy, even with the technology of the day. However realistic projections are impossible to make at this stage of development. The question is whether a system such as this is practical and desirable, considering the benefits it provides over a pull and cache, or pull with notify system. A "pull with notify" system on a global scale is likely to be more complex and insecure than a fast push system.

Ivip involves a fast push system to some depth in the network, as chosen by operators given all the local conditions, update rates, bandwidth costs, technological capabilities of servers etc. Beyond that, Ivip uses query and cache with notify - but only over short distances where the delay times are short, the path lengths are a small fraction of the distance around the planet, and where costs are low and reliability high, compared to a global query server system.

It is difficult to quantify the limits of a system such as this, or the tasks it will need to perform in the future. However, if an architecture such as this seems feasible, its design should be developed further so that more concrete estimates can be made of its short-term cost and worth, and of its long-term potential to scale to

very large sizes.

9.2. Managing Replicators

Replicators should be easy to create and deploy. Any substantial server with the requisite software, in a suitable location, will do the job. However a successful system will require some mechanisms which ensure reliable operation with a minimal amount of configuration and ongoing management.

In the current model, each Replicator normally receives feeds from two upstream Replicators, and generates some figure N feeds for downstream devices. Each Replicator should be able to request and quickly gain a replacement feed from another upstream Replicator if one of those it is using becomes unavailable, or unreliable.

This requires that Replicators in general be operating below capacity, so that when others in their level fail, they can take up the slack. This needs to be locally configured beforehand, with upstream Replicators of organisations which have agreed to provide the feeds, and with downstream Replicators of organisations who have requested them.

It is possible to imagine a sophisticated, distributed, management system for the Replicator network. This could be developed over time, since for initial deployment, considerable manual configuration and less automation would probably be acceptable.

10. Security Considerations

There are many potential security problems with any bold new architectural addition to the Internet. This ID mentions some authentication and security issues and possible solutions to them, but the full consideration of security will occur as the proposal is fleshed out in greater detail.

11. IANA Considerations

[To do as more detail is developed about data formats and communication protocols.]

12. Informative References

[I-D.farinacci-lisp]
Farinacci, D., "Locator/ID Separation Protocol (LISP)", draft-farinacci-lisp-05 (work in progress), November 2007.

[I-D.fuller-lisp-alt]
Farinacci, D., "LISP Alternative Topology (LISP-ALT)", draft-fuller-lisp-alt-01 (work in progress), November 2007.

[I-D.irtf-rrg-design-goals]
Li, T., "Design Goals for Scalable Internet Routing", draft-irtf-rrg-design-goals-01 (work in progress), July 2007.

[I-D.jen-apt]
Jen, D., Meisel, M., Massey, D., Wang, L., Zhang, B., and L. Zhang, "APT: A Practical Transit Mapping Service", draft-jen-apt-01 (work in progress), November 2007.

[I-D.lear-lisp-nerd]
Lear, E., "NERD: A Not-so-novel EID to RLOC Database", draft-lear-lisp-nerd-03 (work in progress), January 2008.

[I-D.vogt-rrg-six-one]
Vogt, C., "Six/One: A Solution for Routing and Addressing in IPv6", draft-vogt-rrg-six-one-01 (work in progress), November 2007.

[I-D.whittle-ivip-arch]
Whittle, R., "Ivip (Internet Vastly Improved Plumbing) Architecture", draft-whittle-ivip-arch-01 (work in progress), January 2008.

[TRRP] Herrin, W., "TRRP", February 2008.

Appendix A. Acknowledgements

[I-D.whittle-ivip-arch] includes a list of people who have helped in some way with this project. Some have helped a great deal and I thank them all. This is not to say that any of these people necessarily support Ivip as currently described.

Author's Address

Robin Whittle
First Principles

Email: rw@firstpr.com.au
URI: http://www.firstpr.com.au/ip/ivip/

Full Copyright Statement

Intellectual Property

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).