

Extending REBEKAH-IP with Central Port Allocations for Un-Ambiguous IPv4 Address Expansion

Sanchai Rattananon, Björn Landfeldt and Aruna Seneviratne

Abstract—In recent times, the imminent lack of public IPv4 addresses has attracted the attention of both the research community and the industry. The problem becomes even more pressing with the introduction of third generation (3G) cellular networks which are predicted to introduce hundred of millions of new IP capable terminals over the next few years. The cellular industry has decided to combat this problem by using IPv6 for the new terminals, but unfortunately since the vast majority of services will reside in the IPv4 address space the problem will remain for the foreseeable future.

There have been many proposals on how to address the issue by extending the IPv4 address space so that the introduction of IPv6 is given more time for a smooth transition. One of these proposals, REBEKAH-IP is promising in that it is scalable and does not limit the services that can be supported. However, the scheme is limited in that there is a possibility for call blocking as the system gets loaded. In this paper, we introduce an improvement to REBEKAH-IP in which this side effect is removed, creating a robust and fully scalable system.

Index Terms—IPv4, IPv6, Address space extension, NAT

I. INTRODUCTION

One of the most pressing issues in the Internet today is the shortage of IPv4 addresses. The header format of IPv4 has a fixed size of 32 bit for addresses and due to this limitation it is predicted that all addresses will be occupied by year 2008 [1]. In order to combat this situation, the IETF has decided to introduce the next generation of IP, version 6 [2]. However, even though there are mechanisms in place for the transition to the new protocol [3], the process is very slow due to increased complexity to the individual networks as well as the lack of financial incentives for the migration. This has resulted in an almost non-existent IPv6 deployment in the Internet today.

The cellular industry has identified this problem and therefore specified IPv6 as the primary network protocol in the third generation cellular networks. The incentive is the vast

This work was partially sponsored by Ericsson Research, Networks and Systems in Kista, Sweden.

S. Rattananon and A. Seneviratne are with the School of Electrical Engineering and Telecommunications, The University of New South Wales, Sydney 2052 Australia, (san@mobqos.ee.unsw.edu.au, a.seneviratne@unsw.edu.au).

B. Landfeldt was with Ericsson Research, now with School of Information Technologies and School of Electrical and Information Engineering, University of Sydney NSW 2006 Australia, (bjornl@staff.usyd.edu.au).

number of new terminals expected to populate these networks and their need to access data services from the Internet. However, it has been shown that IPv6 will not solve the immediate problem because these terminals will access fixed Internet services, which reside in the IPv4 address space. Thus the transition mechanisms still require the usage of IPv4 addresses to enable IPv6 terminals to communicate with hosts in the IPv4 address space [4].

There have been a number of proposals from the research community, which attempt to address this problem.

One of these schemes, Realm Base Kluge Address Heuristic-IP, REBEKAH-IP [4, 6] is the most promising as it maintains IP transparency for applications while having a minimal impact on deployment. However, with this method there is a possibility that there will be irresolvable ambiguities that lead to failure of communication attempts. In this paper, we extend on the REBEKAH-IP scheme to resolve this issue and show how the revised system will exhibit the necessary properties to extend the life span of IPv4 until IPv6 gains enough momentum to be attractive enough to be widely deployed in the fixed Internet.

The rest of this paper is organized as follows: In section 2 we introduce related work, section 3 details the original REBEKAH-IP proposal. Then, in section 4 we present our extension to REBEKAH-IP, we detail a prototype implementation and experimental results and conclude and discuss future work in sections 5 and 6 respectively.

II. RELATED WORK

Over the past few years there have been a number of proposals for expanding the IPv4 address space to accommodate for more hosts in the Internet. The proposals can be divided into two major groups.

A. Network Address Translators

The first group of NATs [5, 7] uses different forms of address translation methods to enable traffic traversal between different address realms. NATs are typically deployed at the border of private networks and operate as middle boxes that are transparent to hosts in the public realm. However, no proposed NAT methods are transparent to the applications and impose different restrictions on the type of applications that can be used [8]. In [4] it is noted that an address expansion mechanism has to have the following properties:

- It should be scalable (being able to support millions

of new 2.5 and 3G terminals that are currently being added to the Internet)

- It should be application friendly (it should not restrict the applications that can be used together with the scheme)
- It should allow network initiated communication (make private terminals reachable from the public Internet) and
- It should have relatively low impact on the current infrastructure (so it can be deployed).

None of the NAT schemes meet all these criteria [4].

Another proposal, AVES [10], makes use of distributed address translators to which public hosts make connections. The scheme uses a DNS to dynamically allocate such translators to act on behalf of private hosts in order to allow network-initiated communication. This scheme shares many properties with other NAT schemes and also fail to meet the above-mentioned criteria. Most notably, it requires the use of Application Layer Gateways ALGs to be deployed in order to maintain application transparency and as discussed in [4], this poses a severe limitation for deployment of large-scale public access networks.

B. Tunneling Mechanisms

The second group of extension methods use different forms of tunnelling mechanisms to forward data between two different address realms. The ngrans working group within the IETF has proposed a number of methods for transition from IPv4 to IPv6 based networks. The work has primarily concerned itself with methods for co-existence of the two protocols and not interoperability. However, through the use of a combination of tunnelling mechanisms [11] it is possible to interconnect IPv6 and IPv4 domains. Even so, the tunnelling mechanism does not extend the public IPv4 address space and still requires public IPv4 addresses for the communication between hosts in the public realm and private realms. Therefore, the mechanism does not solve the immediate need for address expansion until IPv6 gains enough momentum to start easing the burden on IPv4 addressing.

The Comet group at Columbia proposes IP4+4 [9] a method that uses IP-IP tunnels to expand the v4 address space by adding a 32 bit private address. The mechanism appends a minimal “additional” IP header containing a private address after the original IP header. For packet originating from the public realm, special border routers at the private realms can easily read the extra header, remove the original IP header and forward the packet using the private address within the private network. Similarly, packets originating in the private domain can easily be pre-pended with a header containing the IP address of the border router before being forwarded into the public realm.

This proposal has two important advantages. Firstly, all intermediate routers will see a standard IPv4 header and ignore the additional header and therefore there is no need to upgrade these routers to comply with the scheme. Secondly, the mechanism provides hosts with unique end-to-end addresses and transparency towards applications. However,

the scheme requires all existing end hosts to be upgraded to be able to communicate using the scheme. This poses a significant problem of the same magnitude as a global upgrade to IPv6 through transition methods.

III. REBEKAH-IP

REBEKAH-IP integrates two existing NAT proposals [5], RSIP and Bi-directional NAT and adds a set of dispatch mechanisms that allows public IPv4 addresses to be reused. RSIP was chosen because it eliminates the need for ALGs for each application. In order to enable public realm initiated communication some functionality from Bi-directional NAT were chosen. Thus, the scheme consists of a single ALG that allows a DNS to separate queries from a private and a public realm.

The scheme is illustrated in figure 1. Similar to other NAT based solutions, a border gateway is deployed at the edge of the private realm which is capable of routing packets in-between the two realms. The gateway is further extended with the DNS/ALG function, which is responsible for managing the translation process, including the configuration of the private hosts. Finally, the private hosts are extended with the capability of being configured with IP addresses that they have been assigned by the DNS.

Thus, the infrastructure changes necessary when using REBEKAH-IP are the deployment of the gateway/DNS node and a simple upgrade of the private hosts to comply with the scheme. This makes the scheme highly deployable since the hosts belong to a small controlled set and the scheme can be deployed separately from public IPv4 services and not parallel to public IPv4 services. It is also worth noting that 3G terminals are yet to be widely deployed and therefore it will be easy to incorporate the support for REBEKAH-IP during the roll out of new terminals.

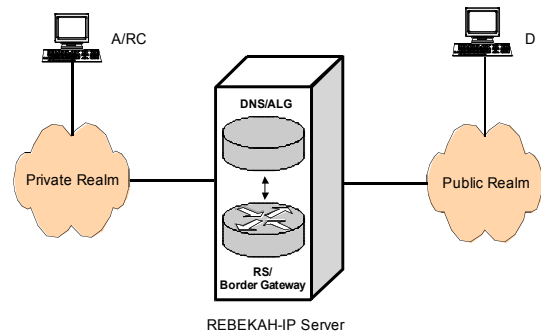


Figure 1: An overall architecture of REBEKAH-IP.

REBEKAH-IP uses a set of public IPv4 addresses that are used to configure the private terminals. A main difference from previous proposals is that REBEKAH-IP allows public IP addresses to be reused, i.e. more than one private host is configured with a single public address. In doing so, in effect the scheme moves IP address multiplexing from the private host to the border gateway. The separation of flows is done by looking at all four socket identifiers (sender and receiver IP address and port numbers) rather than the IP addresses alone.

The downside of this is that there is a small possibility that more than one private host will try to open connections to

public hosts using the same four parameters. In this case it will be impossible to distinguish between the different flows. To prevent this from happening, only the initial flow is allowed and subsequent connections are blocked. Even if this should occur very infrequently [4], the very occurrence is an undesirable property that can potentially lead to poor user experience.

IV. REBEKAH-IP WITH PORT EXTENSION (RPX)

The problem with REBEKAH-IP stems from the usage of ephemeral ports when applications open sockets. Since there is no control over the port allocation it is impossible for REBEKAH-IP to predict the sender port a host will use for a certain flow. In order to overcome this problem we propose to modify the REBEKAH-IP scheme to incorporate centralised management of both IP addresses and port numbers. Thus, instead of querying the DNS for a public IP address only when setting up a connection as in RSIP, in our proposal, the host will obtain the sender port number to use for the socket as well as the public IP address to use. This way, RPX will be able to unambiguously extend the address space as follows:

A. Scalability

The port range for an individual host is 2^{16} - the first 1024 ports reserved by the IANA [13]. In a worst case scenario, all private hosts in a network will try to connect to the same public IP address on the same port number. In this case, RPX will unambiguously allow $(2^{16} - 1024) \times N$ (the number of available public IPv4 addresses to the DNS) simultaneous flows. In the best-case scenario, all connections from private hosts will be to separate combinations of public IP addresses and port numbers.

In this case, RPX will unambiguously support $(2^{16} - 1024) \times N \times (2^{32} - N) \times (2^{16} - 1024)$ simultaneous flows. A partial derivative in respect to N to find the maximum yields a theoretical value of $\sim 2^{62}$ simultaneous flows through a single RPX gateway. However, this is an unrealistic value since it occurs when a single RPX gateway has half the IPv4 32 bit address space. From figure 2 we can see that for a more realistic value of 1000 addresses (for a cellular 3G operator) the outcome is $\sim 1.8 \times 10^{22}$ simultaneous uniquely distinguishable flows.

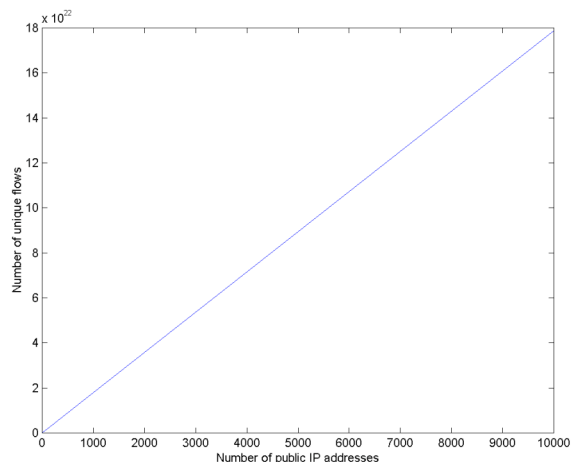


Figure 2. Number of unique flows depending on number of public IP addresses

B. Signalling

The signalling to obtain the configuration can be made using different techniques. Explicit signalling can be used as is done in RSIP, or the DNS can be used to directly relay information to the querying host. In our prototype implementation we opted for the second option since it is a much simpler and cleaner solution.

The design of the DNS is specified in [14, 15]. The basic function of DNS is to provide a distributed database that maps and convert host names into IP address. In our realisation of RPX we rely on an extended DNS that is able to distinguish between queries from the private and public realms.

Today, when a host queries a DNS, it usually asks for a type A (IPv4) or type AAAA (IPv6) record. This record is well defined and its format cannot be modified in any way. However, the SRV record type [12] can be modified to contain any information; hence we can add port information to this record. We have used the simple function of assuming that hosts in the public realm will query the DNS for A or AAAA records while the private realm hosts by default query for the SRV record of a host. Using this mechanism we allow the DNS to construct a reply to the host that consists of the standard A (AAAA) information plus the addition of IP address and port number to use for the socket.

The steps taken by the DNS/ALG function upon receipt of a query are illustrated in Figure 3. The chart shows how the steps taken to configure the private host and updating the border gateway are transparent to public hosts who receive a standard A or AAAA record as a result. In the case of a query from a private host, the DNS performs a normal query to the DNS in the public host's domain before assigning an address and a port number to the private host.

The algorithms for assigning addresses and port numbers can be implemented in a number of different ways. In our realisation we used a simple form by which we are arranging the addresses in a linked list and step through the list sequentially with each address allocation. If there is no free port for a specific address, the next address in the list is selected instead.

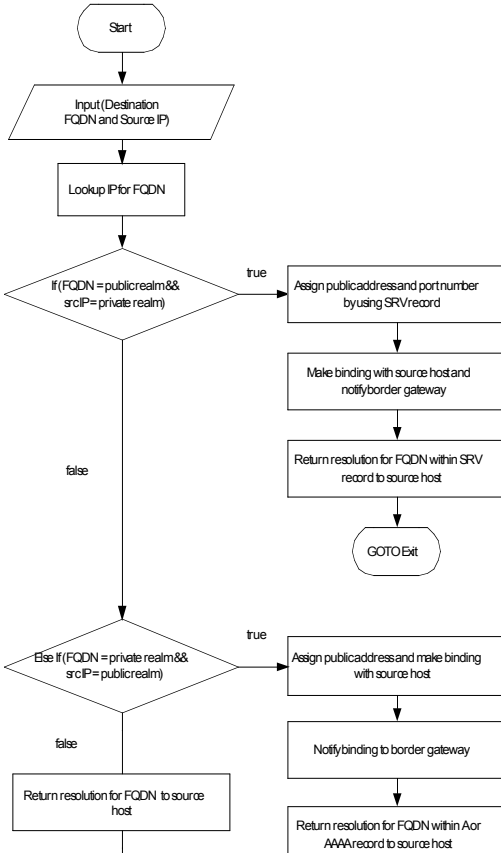


Figure 3. Flowchart of DNS/ALG function.

This simple algorithm spreads the usage of addresses evenly. However, it does not regard the relative occupation of port numbers for each address and therefore the algorithm is not optimized for the system. We intend to investigate how this algorithm should be optimized in future work.

C. Terminal implementation

The private realm terminals will have to support RPX by special functions that are not implemented in current operating systems. Even though this is a negative aspect of the scheme, we argue that it does not have a major impact on the deployment of the scheme for the following reasons:

- Firstly, in the new 3G cellular networks all terminals will be new and it would be easy to include the functions from the beginning.
- Secondly, if RPX is deployed in a small-scale domain, it is possible to upgrade existing hosts within the domain while shifting them from their current environment into an RPX environment.

The great advantage is that existing services and network infrastructure such as routers in the public Internet do not have to be modified in any way in order for RPX to be deployed.

The terminals have to be able to signal the DNS and expect configuration information in return. They also have to be able to configure themselves with the returned information and possibly also override an application's attempt to specify sender port for a socket.

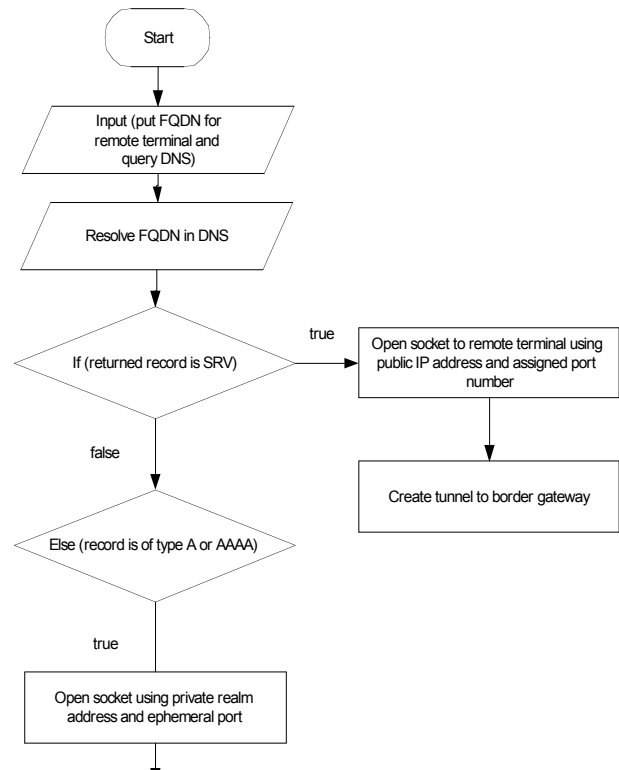


Figure 4 Flow chart of terminal signaling function.

Figure 4 shows a flow chart of the steps taken by a private domain terminal when it wants to open a connection to another terminal in either the private or public realm. After the terminal gets a reply from the DNS the record type determines how the terminal is configured. If the record is an SRV record, the connection is destined for a public domain terminal and the RPX scheme comes into play. If on the other hand the returned record is a standard A or AAAA record, the terminal uses a standard socket creation process using its private address and a randomly assigned ephemeral port number. This way, RPX only comes into play when traversing the border between the private and public realms.

The usage of the SRV record should be treated with care. In our implementation we have defined a format for the return of information whereby we have placed the sender IP address and port number in the ANSWER section. However, this is not a standard representation and it is possible that different implementations can interpret the returned record differently. We therefore believe that it is better to define a new reserved record type for RPX which can be standardized and therefore guarantee correct interpretation.

V. EXPERIMENTAL RESULTS

We made the prototype implementation of RPX in order to validate that the scheme operates according to the theoretical assumptions when designing the scheme. An overview of the prototype and experimental set up is shown in figure 5. The nodes are all 500 Mhz Pentium III PCs running LINUX.

Even though our primary goal was to validate the function of the scheme we were also interested in obtaining crude values of the associated signaling and configuration

overheads. In an optimized implementation the delay values will become significantly lower than the ones we obtained through our prototype implementation. In future work we intend to carry out extensive simulations to obtain more precise values and to optimize the algorithms used in the scheme.

We wanted to obtain measurements of the delay the scheme would introduce on top of the normal DNS query and socket opening values. We therefore made several measurements of the signaling time associated with opening sockets from a private realm terminal to a public realm terminal and timed the different steps individually.

We also carried out these tests both in a local area and in a wide area context to obtain an estimate of the spread of the delay overhead.

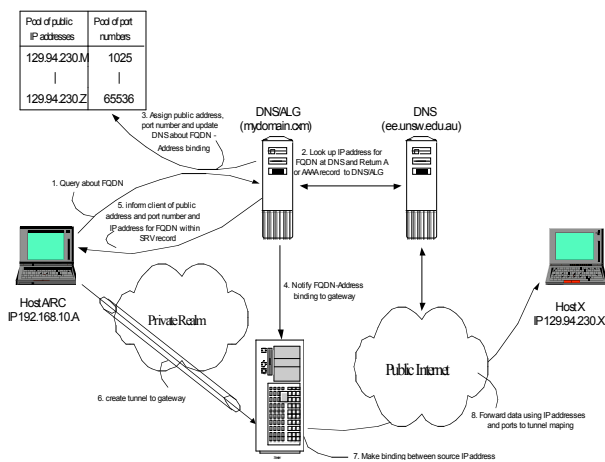


Figure 5 Our experimental set up.

Table 1 shows the results of initiating communication from a private realm terminal located at the University of New South Wales, to a public realm terminal over a single hop. The values in the columns represent different subsets of the configuration delays according to:

- T1 is the time taken for the client to query and receive an SRV record from the DNS/ALG.
- T2 is the time taken for the DNS/ALG to relay the query to the foreign DNS and to obtain the IP address reply for the FQDN.
- T3 is the time taken to configure the client with the obtained parameters.
- T4 = T1 + T3, is total processing time for the prototype implementation of RPX.

Table 1 call setup delay for opening a socket to a host on a local LAN

Experiment	T1(msec)	T2 (msec)	T3 (msec)	T4(msec)
1	924	1301	123	1047
2	938	987	110	1048
3	932	827	101	1033
4	977	737	93	1070
5	956	748	101	1057

6	966	800	100	1066
7	947	966	110	1057
8	994	75	134	1128
9	873	731	161	1034
10	879	1175	129	1008
Average	938.6	902.4	116.2	1054.8

Table 2 shows the same experiments but in this case the remote DNS resided at the University of Sydney, which is located within the same MAN as the DNS/ALG.

Table 2 call setup delay for opening a socket to a host on a MAN

Experiment	T1(msec)	T2 (msec)	T3 (msec)	T4(msec)
1	865	881	156	1021
2	886	812	148	1034
3	909	956	120	1029
4	957	726	114	1071
5	907	1079	133	1040
6	972	755	144	1116
7	951	814	138	1089
8	938	879	145	1083
9	956	1076	87	1043
10	958	1168	72	1030
Average	929.9	914.6	125.7	1055.6

Finally, Table 3 shows the same experiments when the remote DNS resided at the University of Berkeley, California which is located across an intercontinental WAN link from the DNS/ALG.

Table 3 call setup delay for opening a socket to a host across a WAN

Experiment	T1(msec)	T2 (msec)	T3 (msec)	T4(msec)
1	958	1840	108	1066
2	894	1568	130	1024
3	927	2385	125	1052
4	888	1737	130	1018
5	922	2305	134	1056
6	930	2284	112	1042
7	911	1916	130	1041
8	890	1560	100	990
9	912	1514	117	1029
10	945	1618	124	1069
Average	917.7	1872.7	121	1038.7

Figure 6 shows a comparison of the different delay overheads. The figure shows the relative overheads introduced by the scheme in comparison with a normal DNS query according to:

$$T = \frac{T4}{T4 + T2} \times 100\%$$

We have also included in the figure the introduced overhead when the RPX DNS uses caching of resolved addresses as a reference value since this value represents the processing delay in the RPX DNS alone (no query is sent to the remote DNS).

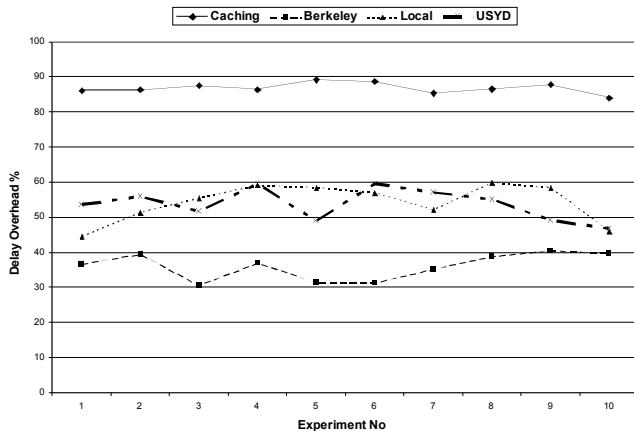


Figure 6 Delay Overhead.

The results from these experiments allow us to draw some conclusions about the usability of the scheme. Firstly, the delay overhead introduced by RPX is in the order of one second for our un-optimised implementation. Even though this value is not excessive, it is large enough to potentially negatively impact the total call setup delay budget in wireless networks and it is therefore necessary to investigate ways of minimizing this figure.

Secondly, we notice that the standard DNS signaling delay is significantly higher than that introduced by RPX. We therefore believe that the DNS signaling delay should be looked at in more detail in order to find out ways of minimizing impact on overall call setup latency.

Thirdly, the delay in processing the request in the DNS/ALG makes up the majority of the RPX delay budget. We therefore conclude that optimization of legacy terminal operating systems is of secondary interest for minimizing this delay and that the optimization effort should be concentrated on hardware and software issues for the DNS/ALG.

VI. CONCLUSION

The problem with the limited IPv4 address space is imminent for the cellular industry. If this problem is not addressed appropriately in the near future, there is a potential threat to the success of 3G-network deployment, as the success will depend on the service offering to the end users.

In this paper we have introduced RPX, and extended and improved version of REBEKAH-IP. RPX will allow operators to deploy 3G networks without limiting the services end-users can access from the public IPv4 based Internet. Furthermore, RPX will scale enough to carry the cellular terminal deployment over a period long enough to allow a smooth transition to IPv6 without the call blocking probability that

occurs in the scheme it extends, REBEKAH-IP.

We have detailed the extended scheme and reported on our prototype implementation, which allowed us to verify the function of the scheme. We have also reported on some early measurements which allow us to conclude that an optimized implementation will not introduce significant call setup delay into the system and therefore not negatively affect the end-users.

In the next phase of our work, we intend to focus on optimizing the algorithms we use in the DNS/ALG in terms of delay. We will carry out simulations of different alternatives to find out how the call setup delay can be minimized. In addition, we will investigate how the algorithms for distributing addresses and port numbers to terminals should be constructed in order to spread combinations evenly.

REFERENCES

- [1] Minutes of the Address Lifetime Expectations working group, proceedings of 29th ETF meeting, Seattle, April 1994.
- [2] S. Deering, R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," Internet RFC 2460, December 1998.
- [3] Work of The IETF Next Generation Transition (ngtrans) working group, <http://www.ietf.org>
- [4] B.Landfeldt, S.Rattananon and A.Seneviratne, "Providing Scalable and Deployable Addressing in Third Generation Cellular Networks", IEEE Wireless Communications Magazine, special issue on 3G Networks, vol.10 no.1, February 2003.
- [5] P.Srisuresh, M.Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", IETF RFC 2663, August 1999.
- [6] B.Landfeldt, S.Rattananon and A.Seneviratne, "Expanding the Address space through REBEKAH-IP: An Architectural View", ICITA 2002, November 2002.
- [7] K.Egevang, P.Francis, "The IP Network Address Translator (NAT)", IETF RFC 1631, MAY 1994.
- [8] D.Senie, "Network Address Translator (NAT)-Friendly Application Design Guidelines", IETF RFC 3235, January 2002.
- [9] Zoltán Turányi, András Valkó, 'IPV4+4,' 10th International Conference on Network Protocols (ICNP 2002), Paris, November 2002.
- [10] T. S. Eugene Ng, Ion Stoica, Hui Zhang, "A Waypoint Service Approach to Connect Heterogeneous Internet Address Spaces", USENIX Annual Technical Conference 2001, Boston, MA, June 2001
- [11] F. Templin, T. Gleeson, M. Lehman "ISATAP Transition Scenario for Enterprise/Managed Networks" IETF draft-ietf-ngtrans-isatap-scenario-01.txt
- [12] A.Gulbrandsen, P.Vixie, L.Esibov, "A DNS RR for specifying the location of service (DNS SRV)", IETF RFC 2782, February 2000.
- [13] <http://www.iana.org/numbers.html>
- [14] P.Mockapetris, "Domain Names-Implementation and Specification", IETF RFC 1035, November 1987.
- [15] M.Lottor, "Domain Administrators Operations Guide", IETF RFC 1033, November 1987.