Home As You Go: An Engineering Approach to Mobility-Capable Extended Home Networking

Zhenkai Zhu
UCLA
zhenkai@cs.ucla.edu

Ryuji Wakikawa
Toyota Infotechnology Center
ryuji@us.toyota-itc.com

Stuart Cheshire
Apple Inc.
stuart@apple.com

Lixia Zhang
UCLA
lixia@cs.ucla.edu

ABSTRACT

In this paper we describe and discuss the design choices of Apple Inc.'s Back to My Mac (BTMM) service. BTMM provides an integrated solution to host mobility support, NAT traversal, and secure end-to-end data delivery through a combination of several existing protocols and software tools. We discuss BTMM's design tradeoffs and examine its applicability as a general solution to mobility-capable extended home networks.

Categories and Subject Descriptors

C.2 [**Computer-Communication Networks**]: Local and Wide-Area Networks

General Terms

Design

Keywords

Home Networks, Mobility, NAT Traversal

1. INTRODUCTION

Thanks to rapid technology advances, the Internet is penetrating into more homes as well as becoming increasingly mobile. For homes that are connected to the Internet, their residents are most likely to own stationary desktops, portable computers such as laptops, as well as hand-held computing devices such as smartphones and tablets. Naturally one would like to have all her computing devices communicate with each other, for example to share photo albums or to synchronize calendars, and to be able to do so all the time, regardless of whether she is traveling with her laptop and accessing the Internet via airport WiFi, or sharing a photo from a smart phone which is connected through a cellular network with the home storage server that is behind a NAT. As a result, there is a soaring demand to provide users with ubiquitous home environments, in which a wide range of personal computing devices, stationary or mobile, seamlessly access and share content and services with each other in a secure way.

However, the Internet is yet to provide "always connected" home networking in general. For example, TCP, the dominant transport protocol utilized by almost all applications, uses the IP addresses of the two end hosts as a part of the connection identifier. Thus a TCP connection breaks if one end moves and changes its IP address. Besides, ISPs usually assign IP addresses to residential home networks dynamically, and home computers are typically behind NAT, further increasing the challenges for remote access and data sharing.

As an input to stimulate new solution exploration, in this paper we present the design and implementation of a mobility-capable home networking solution [1] named *Back to My Mac (BTMM)*. It was first shipped by Apple Inc. with MAC OS X 10.5 release in 2007 and has been widely used since then (BTMM continues to evolve over time; the description in this paper is based on MAC OS X version 10.5.x implementation). BTMM provides an integrated solution to host mobility support, NAT traversal, and secure end-to-end data delivery. What made BTMM unique is that it achieves the above functions mainly through a well engineered combination of existing protocols and software tools, instead of developing new protocols. It showcases how one may use existing tools in creative ways to provide new functions, achieving the goal of fast deployment.

The rest of this paper is organized as follows. In Section 2 we articulate the basic set of problems BTMM must resolve in order to support the required set of functions. In Sections 3 and 4 we explain in detail how BTMM implementation utilizes a set of existing protocols. In Sections 5 we discuss the reasons that lead to BTMM's success and how it can be further improved to serve as a general extended home networks

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
AINTEC'11 November 9–11, 2011, Bangkok, Thailand.
Copyright 2011 ACM 978-1-4503-1062-8/11/11 ...$10.00.

solution for users with heterogeneous devices.

2. PROBLEM OVERVIEW

In this section we provide an overview of extended home networks from 3000-foot height on its goals and the resulting functional requirements.

To build a ubiquitous home network environment for the users, the solution should keep track the reachability of each of the user's computers and provide secure communications among them, in the presence of mobility and NAT. The solution should also be transparent to end users in that no manual configuration is required. Finally, various application services such as file and screen sharing should be able to run on top of it with little or no extra effort required.

To fulfill these goals, an extended home networks solution must resolve the following issues.

- First, each device D needs a stable identifier I_D that is independent from the IP address used for its Internet connectivity, so that I_D can be used to establish TCP connections that survive IP address changes.
- Second, given that the IP addresses of devices may change at any time, there must be a stable piece of information for each device D that other nodes can use to look up the dynamically changing information on D's reachability.
- Third, home computers typically do not have public IP addresses. Instead the ISP may assign a single public IP address to the home network router, which also serves as a NAT box and a DHCP server. The router then assigns private IP addresses to all the computers behind it. The use of NAT makes home computers not directly reachable from the outside unless they initiate the communications first. Furthermore, ISPs tend to assign dynamic IP addresses to home users, thus a home router may obtain a different IP address after a reboot. Therefore effective means must be provided to track the home router address changes and to penetrate the NAT to reach home computers.
- Fourth, one must secure both dynamic updates of device reachability information changes and the lookup process for the updated information. Communications among devices should also be secured.

Solutions to design details such as what to use as a device identifier, how to store device identifiers and reachability information, how to traverse the NAT, and how to bootstrap and enforce security are explained in the next sections where we give a detailed description of the BTMM implementation.

3. BTMM DESIGN

BTMM addresses the issues raised in Section 2 by integrating a set of existing protocols and software tools. It leverages DNS to store and serve device identifiers as well as reachability information. Every BTMM user is assigned a DNS subdomain under `members.me.com` domain (*e.g.* `alice.members.me.com`), which is operated by Apple Inc. The user can then assign a DNS name to each of her devices under that subdomain (*e.g.* `myLaptop.alice.members.me.com`). This DNS name serves as the stable piece of information for each device D mentioned in the second issue. Dynamic DNS updates [23] are used to support dynamic reachability changes, and DNS service is used to look up D's identifier and reachability information. To achieve the security goals, BTMM secures DNS dynamic updates and lookups with Transaction SIGnature (TSIG) [22] and TLS, provides authentication with Kerberos [17] and uses IPsec ESP [14] to encrypt data communications among BTMM devices.

When a user enables BTMM service on a host M, M starts updating the DNS server about its reachability information. If M has a public IPv4 address P, P represents the reachability information of M. On the other hand, if M is behind a NAT, its reachability information is composed of the public IP address of the NAT box and the port number opened on the NAT for M. If the user has multiple hosts running BTMM, M also sets up long-lived queries (LLQs) [6] with the DNS server for her other hosts, so that the DNS server can push any reachability changes of these hosts to M immediately.

To obtain a unique identifier for each host, BTMM automatically generates an IPv6 Unique Local Address (ULA) [10] at the boot time. This design choice allows BTMM to reuse all the existing code of applications and protocols that already support IPv6. BTMM provides an IPv6 tunnel interface to user applications intended to run on top of BTMM. It then wraps the IPv6 application packets with IPsec ESP [14], and encapsulates the packets in a UDP/IPv4 tunnel for NAT traversal. Only the tunnel gets updated when either end moves and the transport layer is unaware of the change. Figure 1 depicts the overall design of BTMM and Figure 2 illustrates how the BTMM packet shows up in the wire.

The following text in this section describe the basic components of BTMM in detail.

3.1 Identifier

BTMM needs to assign a topology-independent identifier to each client host for the following reasons. First, two end hosts may wish to have the established TCP connections between them continue to work in face of their IP address changes. Second, sometimes one needs a constant identifier to be associated with a key so that the security association can stay intact despite the location, hence the IP address, changes.

The above needs for a host identifier impose very little constraint on the properties of the identifier. In particular this identifier does not need to be a permanent one, as long as its life time is no shorter than the life time of any TCP

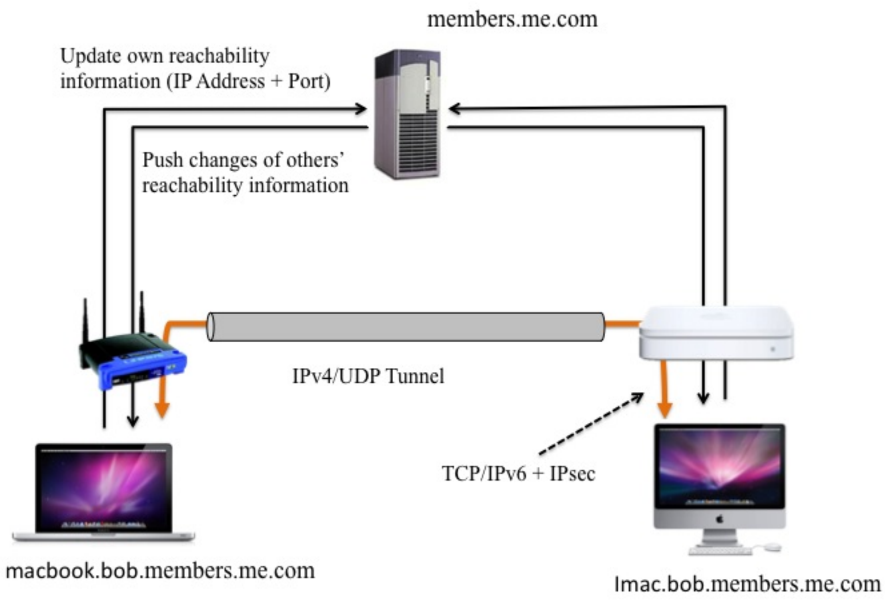
members.me.com
Update own reachability information (IP Address + Port)
Push changes of others' reachability information
IPv4/UDP Tunnel
TCP/IPv6 + IPsec
macbook.bob.members.me.com
Imac.bob.members.me.com

Figure 1: BTMM Overall Design

connection or any security association that runs on the host.

In BTMM, each host composes an IPv6 Unique Local Address (ULA) to be used as its identifier. When the machine boots up, the IPv6 ULA is initialized as zeros and the interface used by BTMM's autotunnel service is marked as inactive. Before activating the autotunnel interface, BTMM randomly generates an IPv6 ULA according to the specifications in [10]. As this ULA includes a 56-bit random string and the EUI-64 identifier [4] of the host's default interface, it is very unlikely to have IPv6 ULA collision between any two hosts of the same user.

3.2 Storing Host Information in DNS

BTMM uniquely identifies each host by its DNS name, and encodes into DNS both the host identifier and its current location information. BTMM stores the host identifier in a DNS AAAA Resource Record (RR), and stores the host location information in a DNS SRV RR [9]. Both the identifier and the location of a host may change from time to time, and these changes are readily handled through DNS dynamic updates.

For hosts behind a NAT box, the use of SRV RR allows BTMM to store both the public IP address of the NAT box and also the port opened for the host. The SRV RR consists of eight fields: *_Service._Proto.Name, TTL, Class, Type, Priority, Weight, Port* and *Target*. It is used in BTMM in the following way:

- *Service* is the symbolic name of the desired service. In BTMM, this field is named "_autotunnel", which means that BTMM automatically sets up a tunnel between two end hosts.

- BTMM uses "_udp" in the *Proto* field to indicate the UDP packets will be used inside the tunnel.

- *Port* is the port opened at the host that provides the service. In most cases, BTMM stores the external port of a NAT opened for the host behind it. If the host is not behind a NAT, the port opened on the host for autotunnel service is placed instead.

- *Target* specifies the canonical hostname of the host that provides the service. In BTMM, it refers to a name constructed by prepending an autotunnel label to the user's domain name. This autotunnel label identifies the interface used by BTMM on the remote host that published this RR. An example of *Target* would look like: `Autotunnel-<EUI-64>.alice.members.me.com`. After obtaining the SRV RR, a BTMM client can query for the A RR using the name in *Target* and get the external tunnel address of the remote host.

3.3 NAT Traversal

BTMM supports automatic NAT traversal as long as the NAT boxes support either NAT-PMP [8] or UPnP [2], a requirement that most commodity home routers satisfy.

Both NAT-PMP and UPnP require that the NAT device can inform the host behind it about its public IP address and also open a port for inbound connections when requested.

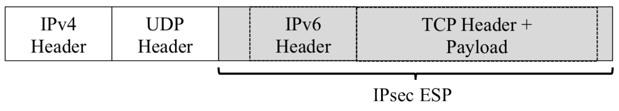

Figure 2: BTMM Packet

When a BTMM client determines that its primary IPv4 address is in one of the private IP address ranges defined in RFC 1918 [20], it sends a query packet to the configured gateway address, and the NAT device responses with a packet containing its public IP address. The client further requests a port mapping, with a desired external port specified. The NAT box honors the request if the port is available or otherwise responds with a different port. This IP address and port pair is then published to the DNS server by the BTMM client. With such information available at DNS, standard IPv4/UDP encapsulation [11] is then applied for NAT traversal when the communication between two BTMM clients happens.

3.4 Tracking Host Information Changes

None of DNS RRs published by a BTMM client is expected to be persistent. A host's identifier changes after a reboot, and its location information changes after movements. A BTMM client starts the procedures to obtain its new reachability information and/or host identifier whenever needed, and then updates the corresponding DNS records immediately through Dynamic DNS updates [23].

However, it is possible that some dynamic updates may fail to reach the authority DNS servers due to unexpected errors or failures. To purge all stale host information from DNS servers, BTMM employs Dynamic DNS Update Lease (DDUL) [7]. When a BTMM client sends a DNS update request for a RR, the update also specifies the RR's lease life time. The client is expected to refresh the RR before its lease expires, or otherwise the RR will be removed from the server. Note the DDUL controls the life time of dynamically updated RRs at the authoritative servers, while the RRs' TTL values control the cache life time at caching resolvers.

In a dynamic environment, changes to DNS RRs can be frequent. However, since a DNS query only retrieves the RR value available at that instance in time, to keep up with the latest changes one must continue to query DNS. This solution faces the dilemma between choosing a low polling rate and leaving the client with stale information, or a high polling rate which would have an adverse impact on the network and server. To let the hosts that care about particular DNS RRs learn the changes quickly and efficiently, BTMM uses DNS long-lived queries (LLQs) [6] which lets the DNS server notify the client host as soon as any changes are made to the concerned DNS data. Different from normal DNS query, LLQ requires a setup phase and persists until the client stops refreshing the LLQ lease. When a change ("event") occurs to a name server's domain, the server checks

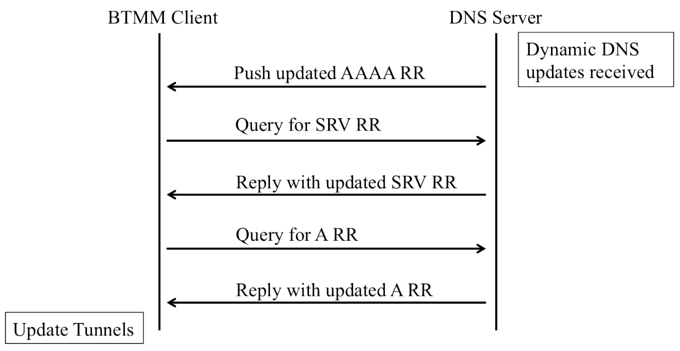

Figure 3: DNS Exchanges Triggered by LLQ

to see whether any existing LLQs are concerned with the affected RRs. If so, the RRs are sent to the LLQ issuers in the form of a gratuitous DNS response. To reduce the number of LLQs required, a BTMM client only sets up LLQ for AAAA RR. Any changes indicated by the LLQ will trigger the client to subsequently query for SRV RR and A RR of the remote host and updates the tunnel according to the query answers, as shown in Figure 3.

4. SECURITY AND PRIVACY IN BTMM

BTMM pays particular attention to security and privacy, which are enforced through three tactics. First, communications with DNS server are authenticated and protected, so that only legitimate users can update and query DNS RRs. Second, user-level authentication is required before one can access services on a remote machine. Third, end-to-end encryption is used so that BTMM users can safely fetch personal data via a network they do not trust (or have no idea whether it is trustworthy).

4.1 Secure DNS Exchanges

BTMM uses Transaction SIGnature (TSIG) to authenticate DNS messages, both dynamic DNS updates and DNS queries. TSIG uses a shared secret key, which is derived from the user's BTMM account password, to establish a trust relationship between the BTMM client and the DNS server. This secret key is never transmitted over the wire; instead, a message digest calculated using the secret key and a hash algorithm is transmitted along with each DNS message. The successful verification of the digest proves the message receiver that the sender is trustworthy (knows the shared secret key) and the message was not modified after it left the sender.

Besides the mutual authenticity provided by TSIG, the DNS messages are also protected by TLS to prevent eavesdropping.

4.2 Service Authentication

Password based authentication is not suitable for use on computer networks. First, passwords sent across the network can be intercepted and subsequently used by eavesdroppers to impersonate the user. Second, password based authentication is inconvenient: users do not want to enter a password each time they access a network service.

Consequently, BTMM relies on Kerberos [17] for the service authentication. A local Key Distribution Center (KDC) is built into the Mac OS X system, which consists of two logically separated parts: an Authentication Server (AS) and a Ticket Granting Server (TGS). Each BTMM application service daemon on a host has a trust relationship with the local KDC.

Assume a user on travel wants to access services (*e.g.* remote access, content sharing, *etc.*) on a desktop H at home from a laptop M. The BTMM client on M authenticates itself to the AS on H based on a secret key derived from the user's BTMM account password, and receives a time-stamped Ticket Granting Ticket (TGT). It then contacts the TGS on H, using TGT to demonstrate its identity and ask for access to services. The TGS sends another ticket to the BTMM client on M. The client then contacts the service daemon on H with the ticket to prove that it has been approved to receive the service.

Later, when the client wants to contact some other service daemons on H, it can reuse the TGT to get additional service tickets from TGS, without resorting to AS again, until the TGT expires.

4.3 End-to-End Data Security

BTMM leverages IPsec to protect the end-to-end communication. It calls Racoon [3], the Internet Key Exchange [13] daemon available on most platforms, to perform the key exchange in order to establish Security Association (SA) between the two end hosts. Afterwards, IPsec ESP in tunnel mode is used to encrypt all the communication data.

Note that BTMM uses the IPv6 ULAs rather than the IPv4 addresses inside the IPsec SAs, based on observations of three drawbacks if IPv4 addresses were used. First, the encryption is not end-to-end. The IPsec protect ends at the NAT and the path from the NAT device to the BTMM client is unprotected and vulnerable to attacks. If the NAT device is not trustworthy, the user's privacy is at high risk. Even if the NAT device is trustworthy (e.g. the user owns the NAT), it is not uncommon that the NAT communicates with the host through broadcast channel, providing opportunities for eavesdroppers to sniff the sensitive data. Second, quite a lot BTMM clients are mobile. Every time they change their attachment points to the Internet, they will acquire different IPv4 addresses. As a result, the previously established SAs become obsoleted. Third, this approach assumes that the NAT device is able and willing to do the IPsec ESP encryption and decryption for the host behind it, which is not always the case.

5. DISCUSSION

After presenting the implementation of BTMM, in this section we assess its pros and cons, and its applicability to mobility-capable extended home networking in general.

5.1 BTMM: An Engineering Solution

BTMM provides an effective solution to extending home networking beyond the traditional scope of home, without developing a new network architecture, demanding new devices or implementing any newly crafted protocols. Except LLQ, the solution is mainly gluing together a collection of existing protocols and services, that are already supported by off-the-shelf software and devices.

As a poor man's mobility support, LLQ is used to push the changes of reachability information of each BTMM client without requiring constant polling. It is easy to implement and good enough for the applications home users care about, although it remains to be seen whether it could support real time applications such as VoIP.

Similarly, BTMM's choice of host identifier is another example of engineering thinking. Among various efforts that have been devoted to developing host identifiers [16, 5], it is often the case to start with listing the properties a host identifier must satisfy, such as global uniqueness and permanency that are needed to globally identify a host. BTMM separates two things that other solutions bundled together: it uses DNS names to identify hosts, and then uses IPv6 ULAs (obtained through DNS lookup) as host identifiers in transport protocols. This solution provides a simple way to generate host identifiers that also allows the reuse of all existing application and transport protocol code that use IP addresses.

BTMM heavily relies on DNS to function. It uses dynamic DNS updates to keep track of moving hosts, and use DNS LLQs to promptly inform the others about the changes. However DNS is a highly distributed system, thus the load of dynamic updates of DNS RRs and LLQs can be distributed among a large set of DNS zones to address scalability concerns.

5.2 BTMM As A Home Networking Solution

Although BTMM is now operated by Apple Inc., and works for Apple products only, one can easily replicate its implementation on other platforms and devices to build one's own BTMM equivalent. Besides, it is easy to add any desired application service on top of BTMM. In this section, we discuss what needs to be done to build one's own mobility-capable extended home networks.

As we mentioned earlier, BTMM utilizes a collection of existing protocols and services, and consequently implementing BTMM on other platforms does not require substantial amount of coding. Once a user updates her own machines to run BTMM, she is able to benefit. The critical piece is obtaining a DNS domain and providing DNS servers that support LLQ. Since DNS is highly distributed, BTMM can be deployed by individual homes, with no requirement for permission from any centralized authority or coordination with other users. To the contrary, each user is free to arrange her own DNS service providers, or even to roll out DNS servers of her own.

If one chooses to implement BTMM on other platforms, one may consider the following change to enhance the performance. Instead of relying on LLQ, one may consider allowing a BTMM-capable device to directly update its reachability information to the other devices that are currently communicating with it. This requires no signification changes to the existing BTMM code, but only adding a few signaling packets between the already communicating devices. In fact, this is a "mature" technique used by other protocols such as MIP [12] and HIP [16]. In case of simultaneous movements at both ends, it is always safe to fall back to DNS resolution.

Another possible enhancement is to employ UDP hole punching techniques [21] for NAT traversal as a fallback solution in case the user is in an access network whose NAT supports neither NAT-PMP nor UPnP.

6. RELATED WORK

The Virtual Private Network (VPN) techniques allow a remote client to establish a secure communication channel with a network that is not open to public. Due to the need of a special gateway, which requires extra budget and configuration, as well as the non-trivial effort to make it work over NAT, VPN is mainly used by corporations for connecting different sites or allowing employees to remotely work. The usage of VPN in home networks is not common.

Although VPN client is supported on mobile devices, the VPN gateway at home network still requires a stable public IP address. And additional mobility protocols such as Mobile IP [12] or MOBIKE [15] must be supported in order to allow the VPN connections survive the mobility [18].

Port forwarding technique has been used to traverse the NAT and to allow users to connect to home computers. However, even with help of solutions such as UPnP IGD [2], the task of configuring port forwarding is still daunting for home users. And neither mobility nor security is considered.

Peer-to-Peer (P2P) based solutions have also been proposed for extending home networks [19]. Users can have access to indoor or outdoor devices using a P2P networking technique. The peers are connected to each other through rendezvous peers and relay peers, which are located outside the home to propagate queries and to relay communications if NAT exists. A managing peer is introduced to troubleshoot the rendezvous and relay peers. Special middleware is also required for the implementation. The resulting solution is complex, yet lacks effective security mechanisms and the capability to maintain seamless connections while mobile devices are roaming.

7. CONCLUSION

Two of the recent growth trends in the Internet is its penetration to homes and its increased mobility due to the soaring number of smartphone, tablets and other hand-held computing devices. These trends call for support of mobility-capable extended home networking. In this paper we showcase BTMM as a good engineering approach to an effective solution to the problem.

Although the existing BTMM implementation is for Mac users only, we believe that its basic design can be easily adapted for implementation on other platforms, hence to offer a general solution to extended home networking for users with heterogeneous devices.

8. REFERENCES

[1] `http://www.apple.com/mobileme/`.

[2] `www.upnp.org`.

[3] `http://ipsec-tools.sourceforge.net/`.

[4] Guidelines for 64-bit global identifier (eui-64) registration authority. `http://standards.ieee.org/develop/regauth/tut/eui64.pdf`.

[5] R. Atkinson, S. Bhatti, and S. Hailes. Evolving the internet architecture through naming. *IEEE Journal on Selected Areas in Communication (JSAC)*, October 2010.

[6] S. Cheshire, M. Krochmal, and K. Sekar. Dns long-lived queries. *IETF draft-sekar-dns-llq-01.txt*, August 2006.

[7] S. Cheshire, M. Krochmal, and K. Sekar. Dynamic dns update leases. *IETF draft-sekar-dns-ul-01.txt*, August 2006.

[8] S. Cheshire, M. Krochmal, and K. Sekar. Nat port mapping protocol (nat-pmp). *IETF draft-cheshire-nat-pmp-03.txt*, April 2008.

[9] A. Gulbrandsen, P. Vixie, and L. Esibov. A dns rr for specifying the location of services (dns srv). *IETF RFC 2782*, February 2000.

[10] R. Hinden and B. Haberman. Unique local ipv6 unicast address. *IETF RFC 4193*, October 2005.

[11] A. Huttunen, B. Swander, V. Volpe, L. Diburro, and M. Stenberg. Udp encapsulation of ipsec esp packets. *IETF RFC 3948*, January 2005.

[12] D. Johnson, C. Perkins, and J. Arkko. Mobility support in ipv6. *IETF RFC 3775*, June 2004.

[13] C. Kaufman. Internet key exchange (ikev2) protocol. *IETF RFC 4306*, December 2005.

[14] S. Kent. Ip encapsulation security payload (esp). *IETF RFC 4303*, December 2005.

[15] T. Kivinen and H. Tschofenig. Design of ikev2 mobility and multihoming (mobike) protocol. *IETF RFC 4621*, August 2006.

[16] R. Moskowitz, R. Nikander, and T. Henderson. Host identity protocol. *IETF RFC 5201*, April 2008.

[17] B. C. Neuman and T. Ts'o. Kerberos: An authentication service for computer networks. *IEEE Communications*, September 1994.

[18] Nokia. The evolution for mobile vpn and its implications for security. White Paper, June 2005.

[19] H. Park, I. Lee, T. Hwang, and N. Kim. Architecture of home gateway for device collaboration in extended home space. *IEEE Transactions on Consumer*

Electronic, November 2008.

[20] Y. Rekhter, B. Moskowtiz, D. Karrenberg, G. J. de Groot, and E. Lear. Address allocation for private internets. *IETF RFC 1918*, February 1996.

[21] J. Rosenberg, R. Mahy, P. Matthews, and D. Wing. Stun - simple traversal of user datagram protocol (udp) through network address translators (nats). *IETF RFC 5389*, October 2008.

[22] P. Vixie, O. Gudmundsson, D. E. 3rd, and B. Wellington. Security key transaction authentication for dns (tsig). *IETF RFC 2845*, May 2000.

[23] P. Vixie, S. Thomson, Y. Rehter, and J. Bound. Dynamic updates in the domain name system(dns update). *IETF RFC 2136*, April 1997.