Internet Engineering Task Force (IETF)
Request for Comments: 6281
Category: Informational
ISSN: 2070-1721

S. Cheshire
Apple Inc.
Z. Zhu
UCLA
R. Wakikawa
Toyota ITC
L. Zhang
UCLA

June 2011

Understanding Apple's Back to My Mac (BTMM) Service

Abstract

This document describes the implementation of Apple Inc.'s Back to My Mac (BTMM) service. BTMM provides network connectivity between devices so that a user can perform file sharing and screen sharing among multiple computers at home, at work, or on the road. The implementation of BTMM addresses the issues of single sign-on authentication, secure data communication, service discovery, and end-to-end connectivity in the face of Network Address Translators (NATs) and mobility of devices.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6281.

Table of Contents

1. Introduction

Apple Inc.’s Back to My Mac (BTMM) service was first shipped with MAC OS X 10.5 release in October 2007; since then, it has been widely used. 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 instead of designing new protocols. Note that we generally refer to Network Address Port Translation (NAPT) as NAT in this document. This document describes the implementation of BTMM and describes how BTMM works in MAC OS X version 10.5.x; BTMM continues to evolve over time.

BTMM provides secure transport connections among a set of devices that may be located over a dynamic and heterogeneous network environment. Independent from whether a user is traveling and accessing the Internet via airport WiFi or staying at home behind a NAT, BTMM allows the user to connect to any Mac hosts with a click, after which the user can share files with remote computers or control the remote host through screen sharing. When a user changes locations and thus also changes the IP address of his computer (e.g., roaming around with a laptop and receiving dynamically allocated IP address), BTMM provides a means for the roaming host to update its reachability information to keep it reachable by the user’s other Mac devices. BTMM maintains end-to-end transport connections in the face of host IP address changes through the use of unique host identifiers. It also provides a means to reach devices behind a NAT.

BTMM achieves the above functions mainly by integrating a set of existing protocols and software tools. It uses DNS-based Service Discovery [DNS-SD] to announce host reachability information, dynamic DNS update [RFC2136] to refresh the DNS resource records (RRs) when a host detects network changes, and DNS Long-Lived Queries (LLQs) [DNS-LLQ] to notify hosts immediately when the answers to their earlier DNS queries have changed. BTMM uses the IPv6 Unique Local Address (ULA) [RFC4193] as the host identifier and employs the NAT Port Mapping Protocol (PMP) [NAT-PMP] to assist NAT traversal. It uses Kerberos [RFC4120] for end-to-end authentication and uses IPsec [RFC4301] to secure data communications between two end hosts.

2. An Overview of Back to My Mac

To keep an established TCP connection running while either of the two end hosts may change its IP address requires that the connection use unique and stable identifiers that do not change with the addresses, and that a mapping service exists between these stable identifiers and dynamically changing IP addresses. BTMM uses DNS to provide this mapping service. Figure 1 provides a sketch of the basic components in the BTMM implementation.

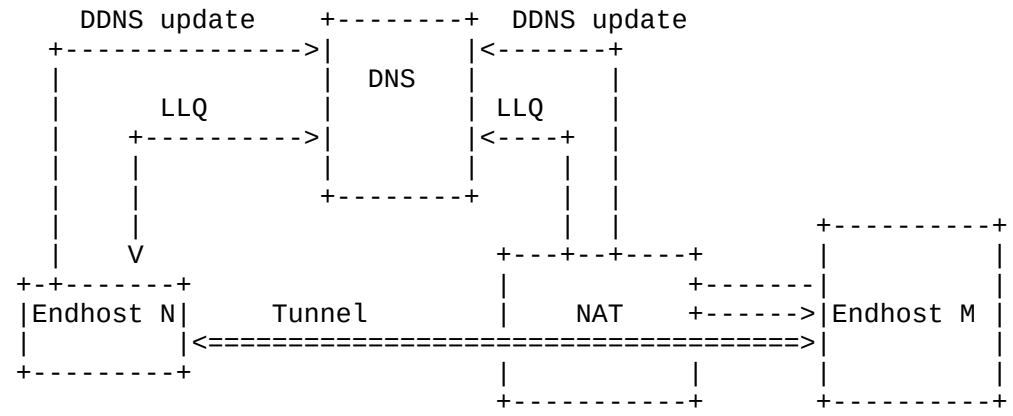


Figure 1

Apple Inc. operates a DNS domain called members.me.com and provides DNS name resolution services for all the subdomains underneath. Every BTMM user is assigned a DNS subdomain under members.me.com, e.g., alice.members.me.com. The user then assigns a DNS name for each of her computers, e.g., myMacPro.alice.members.me.com. The reachability information of each of the user's hosts is encoded in DNS resource records and published in the DNS. For example, if the host myMacPro.alice.members.me.com has a public IPv4 address P, P represents the reachability information to the host. On the other hand, if the host 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 to reach the internal host. In this case, both the public IP address of the NAT box and the port number are encoded into DNS using DNS SRV records [RFC2782], as we explain in the next section. When a user logs in from a host M, M starts updating the DNS server about its reachability information. If the user has multiple hosts, M also sets up LLQs with the DNS server for her other hosts, so that the DNS server can push any reachability changes of these other hosts to M immediately.

To obtain a unique identifier for each host, BTMM automatically generates an IPv6 ULA for each host as its identifier at machine boot time. This design choice allows BTMM to reuse all the existing code of applications and protocols that already support IPv6. To ensure end-to-end data security, BTMM leverages the existing IPsec to protect the communications and Kerberos to perform end-to-end authentication.

BTMM provides an IPv6 socket interface to user applications. It then wraps the IPv6 packets with IPsec Encapsulating Security Payload (ESP) [RFC4303] and encapsulates the packets in a UDP/IP tunnel, as illustrated in Figure 2. Note that this is the case even when both ends have public IPv4 addresses.

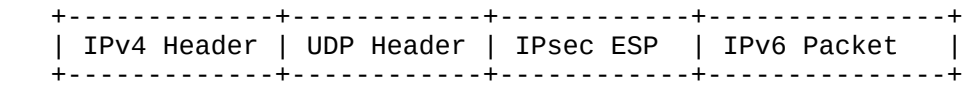


Figure 2

The following sections describe each of the basic components in BTMM. Since this document is intended to be an informal description of the BTMM implementation, it does not include all the details (e.g., packet format, error code, etc) of each component.

3. Encoding Host Information in DNS Resource Records

For each host, BTMM encodes into DNS both the host identifier and its current location information. BTMM stores the host identifier (IPv6 ULA) in a DNS AAAA RR and uses a DNS SRV RR [RFC2782] to represent the host's current location information. For hosts behind a NAT box, the use of a DNS 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, Time to Live (TTL), Class, Type, Priority, Weight, Port, and Target. BTMM uses SRV RRs in the following way.

Service is the symbolic name of the desired service. In the BTMM case, the service is named "autotunnel", which means that the information contained in the SRV RR is used by BTMM to automatically set up a tunnel between two end hosts.

Proto is the symbolic name of the desired protocol. In this document, the protocol is "_udp". BTMM uses "_udp" to tunnel packets between the two ends to achieve NAT traversal.

Name is the domain this RR refers to. When a user subscribes to BTMM service with the username "alice", a domain name "alice.members.me.com" is assigned to her. The user assigns a name, such as "myMacPro", to each host that is appended to the assigned domain name. Hence, the first part of the SRV record would look like this: "_autotunnel._udp.myMacPro.alice.members.me.com".

Priority and Weight are set to zero, since there is only one instance that provides autotunnel service for each name in BTMM.

Port is the port opened on the target host of the service. In BTMM, most likely it is the external port a NAT opened for the host behind it. Knowing the port number is the basic requirement for NAT traversal via UDP encapsulation. If the host is not behind a NAT, the port opened on the host for autotunnel service is placed here.

Target is the canonical hostname of the host that provides the service. In BTMM, it refers to a name constructed by appending the user's domain name to an autotunnel label, which identifies the host and is not generally user-visible. The autotunnel label is created by concatenating "AutoTunnel" with the IEEE EUI-64 identifier [EUI64] of the primary network interface. Hence, an example for the Target field would look like this: AutoTunnel-00-22-69-FF-FE-8E-34-2A.alice.members.me.com. After obtaining the SRV RR, the remote host can query the A RR for the Target and get the external tunnel address for the BTMM client during the NAT Traversal.

4. NAT Traversal

BTMM's NAT traversal function requires NAT router devices to support NAT-PMP or the Universal Plug and Play (UPnP) Internet Gateway Device (IGD). NAT-PMP is the alternative introduced by Apple Inc. to the more common IGD Standardized Device Control Protocol [IGD] as published in the UPnP Forum. Both NAT-PMP and IGD require the NAT devices to be able to open a port for inbound traffic to some host behind it and to inform the host about its public IP address. The differences between IGD and NAT-PMP can be found in [NAT-PMP]. This section focuses on NAT-PMP.

4.1. Introduction to NAT-PMP

NAT-PMP is a protocol that is designed specifically to handle the NAT traversal without manual configuration. When a host determines that its primary IPv4 address is in one of the private IP address ranges defined in "Address Allocation for Private Internets" [RFC1918], it invokes NAT-PMP to communicate with the NAT gateway to request the creation of inbound mappings on demand. Having created a NAT mapping to allow inbound traffic, the client host then publishes its NAT box's public IP address and external port number in a DNS server.

A host sends its Port Mapping Protocol request to the default gateway, which means that by default, this protocol is designed for small home networks where the host's default gateway is the NAT router. If the host finds that NAT-PMP or UPnP IGD is not available on its network, it would proceed under the assumption that the network is a public network.

4.2. Requesting/Removing a Port Mapping

To request a port mapping, the client host sends its request packet via UDP to port 5351 of its configured gateway address and waits 250 ms for a response [NAT-PMP]. If no response is received after 250 ms, the host repeats the process with exponential back-off.

While requesting the port mapping, the host can specify the desired external port (e.g., the port that is identical to the internal port opened on the host), but the NAT device is not obliged to allocate the desired one. If such a port is not available, the NAT device responds with another port. The primary reason for allowing the host to request a specific port is to help recovery from a NAT device crash by allowing the host to request the same port number used before the crash. This simple mechanism allows the end hosts (instead of the NAT box) to keep the mapping states, which turns hard state in the network into soft state, and enables automatic recovery whenever possible.

The default port-mapping lifetime is 3600 seconds. The host tries to renew the mapping every 1800 seconds. The renewal message sent by the client host, whether for the purpose of extending the lease or recreating mappings after the NAT device reboots, is the same as the message requesting a port mapping.

A mapping may be removed in a variety of ways. If a client host fails to renew a mapping, the mapping is automatically deleted when its lifetime expires. If the client host's DHCP address lease expires, the NAT device also automatically deletes the mapping. A client host can also send an explicit packet to request the deletion of a mapping that is no longer needed.

4.3. Obtaining NAT Box's Public IP Address

To determine the public IP address of the NAT, the client host also sends the query packet to port 5351 of the configured gateway address. The NAT device responds with a packet containing the public IP address of NAT.

In case the public IP address of the NAT changes, the NAT gateway sends a gratuitous response to the link-local multicast address 224.0.0.1, port 5350 to notify the clients about the new IP address, and the host can then update its DNS SRV record to reflect its new reachability as we describe in the next section.

4.4. Unsupported Scenarios

There are a number of situations where NAT-PMP (and consequently BTMM) does not work.

4.4.1. NAT behind NAT

Some people's primary IP address assigned by their ISPs may itself be a NAT address. In addition, some people may have a public IP address, but may put their hosts (perhaps unknowingly) behind multiple nested NAT boxes. NAT traversal cannot be achieved with NAT-PMP in such situations.

4.4.2. NATs and Routed Private Networks

In some cases, a site may run multiple subnets in the private network behind a NAT gateway. Such subnetting breaks the assumption of NAT-PMP protocol because a host's default router is not necessarily the device performing NAT.

5. Handling IP Address or Port Changes

This section describes how BTMM handles IP address or port number changes, so that the hosts of the same user can find each other and keep ongoing TCP connections even after the changes happen at one or both ends.

5.1. Updating Local Interfaces and Tunnels

After a BTMM client receives the notification about the network changes, it updates the list of active interfaces. Then, the client sends requests to the NAT device (if it is behind a NAT) in order to create a port mapping and obtain the new public IP address.

Next, the BTMM client makes changes to the local autotunnel interface, i.e., configures the IPv6 interface for the inner address of the tunnel. If there are established tunnels, it scans to find those whose local inner/outer addresses have been changed since the tunnel was set up, and then puts in the current addresses.

With all these done, the BTMM client publishes the changes to DNS.

5.2. Dynamically Updating Reachability Information

The mobile nature of BTMM clients implies that dynamic DNS updates are required if the location information of hosts are to be published via DNS.

However, a mobile host may have dynamically updated an RR but the updated value has not been propagated to the authoritative DNS server, leaving stale RRs in the server. Hence, Dynamic DNS Update Leases (DDULs) [DDUL] are employed by BTMM to minimize the chances of stale RRs. Note that DDUL controls the lifetime of dynamically updated RRs at the authoritative DNS servers, while the RRs' TTL values control the cache lifetime at caching resolvers.

In case of network changes, the RRs of a host are updated immediately after local interfaces are properly configured, and after the port mapping and the public IP address of the NAT are obtained. Usually there are 4 types of RRs involved: a AAAA RR for updating the new host identifier of the host (possibly the same as the old one); an SRV RR for updating the autotunnel service information, which includes the new external port; an A RR for updating the new public IP address; and a TXT RR for describing the autotunnel device information. The name for the SRV RR is discussed in Section 3, and the names for the A, AAAA, and TXT RRs are specified in the Target field of the SRV RR. The host then constructs and sends an SRV query for the dynamic DNS server to which it should send updates. Following our example for alice, it queries the SRV RR for _dns-update-tls._udp.alice.members.me.com. Then, the updates are sent to the dynamic DNS server returned in the Target field of query response.

In addition, periodic refreshes are also required by the DDUL even in the absence of any network changes. The update requests contain a signed 32-bit integer indicating the lease life in seconds. To reduce network and server load, a minimum lease of 30 minutes is required. On the other hand, to avoid stale information, a lease longer than 2 hours is not allowed in BTMM. The typical length is 90 minutes. The client host refreshes the RRs before the lease expires to prevent them from being deleted by the server.

5.3. Getting Up-to-Date DNS Resource Records without Polling

In dynamic environments, changes to DNS information can often be frequent. However, since a DNS query only retrieves the RR value available at that instance in time, one must continue to query DNS to learn the latest changes. This solution presents the dilemma of choosing a low polling rate that leaves the client with stale information or choosing a high polling rate that would have an adverse impact on the network and server.

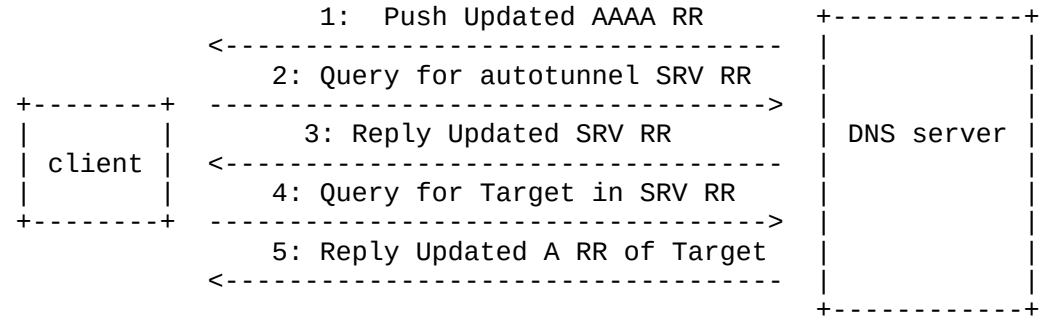
To let the hosts that care about particular DNS RRs learn the changes quickly and efficiently, BTMM uses DNS Long-Lived Queries (LLQs) [DNS-LLQ] to let the DNS server notify the client host once any changes are made to the concerned DNS data.

To obtain the LLQ server information, the client issues an SRV query. So alice's host issues a query for _dns-llq-tls._udp.alice.members.me.com and obtains the server that provides LLQ service. LLQs are initiated by a client and are completed via a four-way handshake: Initial Request, Challenge, Challenge Response, and ACK + Answers. During the Challenge phase, the DNS server provides a unique identifier for the request, and the client is required to echo this identifier in the Challenge Response phase. This handshake provides resilience to packet loss, demonstrates client reachability, and reduces denial-of-service attack opportunities.

LLQ lease is negotiated during the handshake. In BTMM, the minimum lease is 15 minutes, and the maximum lease is 2 hours. Leases are refreshed before they expire.

When a change ("event") occurs to a name server's domain, the server checks if the new or deleted RRs answer any LLQs. If so, the RRs are sent to the LLQ issuers in the form of a gratuitous DNS response. The client acknowledges the reception of the notification; otherwise, the server resends the response. If a total of 3 transmissions (with exponential backoff) fail, the client is considered unreachable, and the LLQ is deleted.

A BTMM client then updates its tunnels according to the query answers. The callback function for automatically updating tunnels is depicted Figure 3.



In Step 1: Client learns the inner IP address of the tunnel.
In Step 3: Client learns the port opened for UDP NAT traversal.
In Step 5: Client learns the public IP address of the remote NAT, i.e., the outer IP address of the tunnel.

Figure 3

6. IPv6 ULA as Host ID

6.1. The Need for a Host 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 survive network changes. Second, sometimes one needs a constant identifier to be associated with a key so that the Security Association can survive the location changes.

The above needs for a host identifier impose very little constraint on the properties of the identifier. In particular, one notes that this identifier does not need to be a permanent one as long as its lifetime is no shorter than the lifetime of any TCP connection or any Security Association that runs on the host.

6.2. What to Use as Host Identifiers

Much effort has been put into the development of host identifiers. Possible candidates for host identifiers include DNS name and Host Identity Tag (HIT) in the Host Identity Protocol (HIP) [RFC4423]. However, because the current protocol stack used IP as identifiers in TCP, other transport protocols, and some applications, if one does not wish to rewrite all the transport protocol and application code, then DNS is ruled out as infeasible because DNS names have variable lengths.

For HIP, although publickey-based HIT has the same length as an IPv6 address, we still lack a secure way to retrieve the public keys. Under this condition, using HIT would not bring us much benefit.

BTMM chooses to use IPv6 ULA as the host identifier so that all the existing IPv6 code can be used directly. Since the identifier does not need to stay constant over machine shutdown or crashes, each host creates an IPv6 ULA at boot time. Furthermore, since a host does not leak this ULA to the network, it would not cause any problem to the routing system.

6.3. IPv6 ULA Configuration

In BTMM, IPv6 ULA is advertised to be used in the autotunnel service of the host. Thus, the IPv6 address needs to be configured before BTMM starts its service.

When the machine boots up, the IPv6 address for autotunnel service is initialized as zeros, and the autotunnel interface is marked as inactive. During the process when BTMM updates the interfaces list

(which is performed every time the network changes), BTMM would randomly generate an IPv6 ULA according to [RFC4193] if the IPv6 address is found uninitialized. The first octet of the ULA is set to be "0xFD", and the following 7 octets are randomly selected from 0~255. Finally, the EUI-64 identifier fills up the remaining 8 octets. Since there are 56 random bits plus a theoretically unique EUI-64 identifier, it is unlikely for an IPv6 ULA collision between any two hosts of the same subscriber to occur.

This locally generated ULA remains unchanged when the machine is on, despite its location changes. Hence, the user can fully enjoy the benefits brought by topology-independent host identifiers. After the machine is turned off, this particular ULA is no longer kept.

7. Securing Communication

BTMM users often have to fetch their personal data via a network they don’t trust (or they do not know whether or not it’s trustworthy). Hence, it is important for BTMM to have an effective means to secure the communications.

7.1. Authentication for Connecting to Remote Host

Kerberos is a "single sign on" technology and has been supported in Apple’s products since MAC OS X 10.5. Each Mac OS X client maintains a local Key Distribution Center (KDC) for the use of Bonjour and peer-to-peer security.

When the user first signs in to MobileMe on a host, it automatically receives a digital certificate and private key for "Back to My Mac Encryption Certificate" from KDC. When the user connects to another system using BTMM, authentication is performed using the Public Key Cryptography for Initial Authentication in Kerberos (PKINIT) protocol [RFC4556] with that certificate. After that, the user is granted a "ticket" that permits it to continue to use the services on the remote host without re-authenticating until the ticket expires (a ticket usually has a 10-hour lifetime).

7.2. Authentication for DNS Exchanges

BTMM uses Transaction SIGnature (TSIG) to authenticate the user when dynamic DNS update is performed [RFC2845]. Also, to protect the subscriber’s privacy, LLQ is required to contain TSIG. This authentication mechanism is based on the shared secret key, which in BTMM’s case is derived from the subscriber’s MobileMe account password.

Every time a DNS request/response is going to be issued, a TSIG RR is dynamically computed with the HMAC-MD5 [RFC2104] message digest algorithm (and the TSIG RR will be discarded once its has been used). Inside the TSIG RR, the name of the shared secret key in the domain name syntax is included, so the receiver knows which key to use (this is especially useful if the receiver is the DNS server). This TSIG RR is appended to the additional data section before the message is sent out. The receiver of the message verifies the TSIG RR and proceeds only if the TSIG is valid.

Besides, the DNS messages are also protected by TLS [RFC5246] to prevent eavesdropping.

7.3. IPsec for Secure End-to-End Data Communication

7.3.1. Internet Key Exchange

Before the Security Association can be established between two end hosts, the Internet Key Exchange (IKE) [RFC5996] process needs to be accomplished.

BTMM calls Racoon [Racoon], the IKE daemon, to do the key exchange, after which the key is put into the Security Association Database (SAD). The exchange mode is set to be aggressive so that it will not take too long, and it uses a pre-shared key to do the user authentication. The subscriber's Fully Qualified Domain Name (FQDN) is used as both identifier and pre-shared key during the IKE process.

7.3.2. Discussion: End-to-End Encryption

When it comes time to set up Security Associations between two BTMM clients, we have two choices: put the other host's IPv4 address in the destination address field or put it in the IPv6 address of the remote end.

If the IPv4 address (which is the public address of a NAT) is chosen to associate with a Security Association, that means we set up a Security Association between one end host and the NAT of the other host. The IPv6 packet would then be wrapped by the UDP header and then get encrypted by ESP. After the encrypted packet arrives at the NAT, the NAT device decrypts the packet and sends it to the destination according to the port mapping. Although this approach seems viable, there are 3 drawbacks:

- First, the encryption is not really end-to-end, i.e., only the path between one end host and the NAT device of the other end is protected. The rest of the path, from the NAT device to the other BTMM client, is unprotected and vulnerable to attacks. If the NAT

device is not trustworthy, the communication is at high risk. Even if the NAT device is trustworthy (e.g., the user owns the NAT), it is not uncommon for the NAT to communicate with the host through a broadcast channel, which provides opportunities for an eavesdropper to sniff the sensitive data (consider the unlocked "free" WiFi access near your neighborhood).

- Second, quite a few BTMM clients are on the move very often. Every time they change their attachment points to the Internet, they will get different IPv4 addresses. As a result, the previously established Security Associations become obsoleted, and the two end hosts need to re-establish them again. This is a waste of time and resources.

- Third, this approach assumes that the NAT device is able and willing to do the IPsec ESP for the host behind it, which is not always the case.

Consequently, BTMM decides to put the IPv6 ULA into the destination field of IPsec Security Associations. In this way, the end-to-end path between the hosts is fully protected, and the Security Associations survive the network changes since the IPv6 ULA remains the same even if the BTMM client changes its location. Furthermore, the encryption is transparent to the NAT device, which means the NAT device is not required to interfere with the IPsec protection.

8. Security Considerations

The BTMM implementation utilizes existing security protocols to address the end-to-end security considerations. It uses Kerberos [RFC4120] for end-to-end authentication and uses IPsec [RFC4301] to secure data communications between two end hosts.

9. References

9.1. Normative Reference

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

[RFC2136] Vixie, P., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, April 1997.

[RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, February 2000.

[RFC2845] Vixie, P., Gudmundsson, O., Eastlake, D., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, May 2000.

[RFC4120] Neuman, C., Yu, T., Hartman, S., and K. Raeburn, "The Kerberos Network Authentication Service (V5)", RFC 4120, July 2005.

[RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, October 2005.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.

[RFC4556] Zhu, L. and B. Tung, "Public Key Cryptography for Initial Authentication in Kerberos (PKINIT)", RFC 4556, June 2006.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

9.2. Informative References

[DDUL] Sekar, K., "Dynamic DNS Update Leases", Work in Progress, August 2006.

[DNS-LLQ] Sekar, K., "DNS Long-Lived Queries", Work in Progess, August 2006.

[DNS-SD] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", Work in Progress, February 2011.

[EUI64] "Guidelines for 64-bit Global Identifier (EUI-64)", <http://standards.ieee.org/regauth/oui/tutorials/EUI64.html>.

[IGD] "Internet Gateway Device (IGD) Standard Device Control Protocol", <http://www.upnp.org>.

[NAT-PMP] Cheshire, S., "NAT Port Mapping Protocol (NAT-PMP)", Work in Progress, April 2008.

[RFC1918] Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, February 1996.

[RFC4423] Moskowitz, R. and P. Nikander, "Host Identity Protocol (HIP) Architecture", RFC 4423, May 2006.

[Racoon] "Racoon", <http://ipsec-tools.sourceforge.net>.

Authors' Addresses

Stuart Cheshire
Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
USA
Phone: +1 408 974 3207
EMail: cheshire@apple.com

Zhenkai Zhu
UCLA
4805 Boelter Hall, UCLA
Los Angeles, CA 90095
USA
Phone: +1 310 993 7128
EMail: zhenkai@cs.ucla.edu

Ryuji Wakikawa
Toyota ITC
465 Bernardo Avenue
Mountain View, CA 94043
USA
EMail: ryuji.wakikawa@gmail.com

Lixia Zhang
UCLA
3713 Boelter Hall, UCLA
Los Angeles, CA 90095
USA
Phone: +1 310 825 2695
EMail: lixia@cs.ucla.edu