Publication VIII

G. Camarillo, J. Mäenpää, A. Keränen, and V. Andersson. Reducing Delays Related to NAT Traversal in P2PSIP Session Establishments. In *Proceedings of the IEEE Consumer Communications and Networking Conference (CCNC)*, Pages 549-553, January 2011.

Reducing Delays Related to NAT Traversal in P2PSIP Session Establishments

Gonzalo Camarillo, Jouni Mäenpää, Ari Keränen, and Veera Andersson
NomadicLab, Ericsson Research
Jorvas, Finland
Email: Gonzalo.Camarillo@ericsson.com, Jouni.Maenpaa@ericsson.com
Ari.Keranen@ericsson.com, and Veera.Andersson@ericsson.com

Abstract— **This paper focuses on reducing the Network Address Translator (NAT) traversal-related components of the session establishment delay in peer-to-peer Session Initiation Protocol (P2PSIP) overlays. To reduce the delay, we propose to group the management of different connections so that the (time-consuming) NAT traversal procedures performed for one connection can be reused when establishing other connections. To do this, we propose to use the Host Identity Protocol (HIP) to perform connection management in P2PSIP overlays. In order to evaluate the performance gains resulting from this approach, we have implemented a P2PSIP system whose modular design allows us to build overlay networks with and without HIP and measure their differences in performance. Our experiments show that grouping the management of different connections results in a significant reduction in the session establishment delay in the presence of NATs.**

I. Introduction

Session establishment delays are an important factor when evaluating interpersonal communication systems. Longer delays reduce the perceived quality of a particular system. Our previous work [1] shows that peer-to-peer Session Initiation Protocol (P2PSIP) systems introduce higher delays than client/server SIP [2] systems. Additionally, our work [3] also shows that the presence of NATs significantly increases those delays making them unacceptably high in some cases. Therefore, one of the main challenges when designing interpersonal communication systems based on P2PSIP is to minimize session establishment delays, which is the focus of this paper. In particular, this paper focuses on reducing the components of the session establishment delay that relate to NAT traversal.

NAT traversal in P2PSIP systems is based on Interactive Connectivity Establishment (ICE) [4]. Our experiments show that ICE-related delays are by far the most significant components of session establishment delays in P2PSIP [3]. In particular, the fact that ICE procedures are executed several times, one for SIP and one for each media stream being established (e.g., audio and video), is often the most important source of delay. The goal of this paper is to minimize ICE-related delays in P2PSIP. In order to do this, we propose an architecture based on the Host Identity Protocol (HIP) [5] to manage connections between nodes as a group instead of individually. We have implemented the architecture and evaluated the performance gains associated with it.

We have chosen to use HIP to perform connection management in P2PSIP for several reasons. NAT traversal in HIP is also based on ICE. Therefore, the P2PSIP procedures for Session Traversal Utilities for NAT (STUN) and Traversal Using Relays around NAT (TURN) server selection can be easily reused. Being able to reuse these procedures simplifies the integration of HIP and P2PSIP. Also, since both HIP and P2PSIP use the same NAT traversal mechanism (i.e., ICE), performance differences between P2PSIP with and without HIP are solely a result of the way HIP handles connection management. The use of a different NAT traversal mechanism in our experiments would have made it less straightforward to measure performance gains due to grouping the management of several connections (comparing ICE with other NAT traversal mechanisms falls outside the scope of this paper). HIP implements a so-called identifier/locator split by which a node's identifier remains constant while its locator can change. This allows a natural separation between routing based on identifiers and routing based on locators, which is at the core of building overlay networks such as P2PSIP overlays.

Sections II and III provide background information on HIP and ICE, respectively. Section IV presents the architecture used to combine HIP and P2PSIP. Section V describes our implementation. Section VI evaluates the performance of the system. Section VII contains the conclusions of the paper.

II. Background on HIP

The Host Identity Protocol provides a secure mobility and multihoming solution by introducing a new *Host Identity* layer in the Internet Protocol stack. The new layer, located between the transport and internetworking layers, replaces a host's IP address with the host's identifier as the connection identity, as shown in Figure 1. Since transport layer connections are no longer statically bound to network interface addresses, the connections can be dynamically bound to different IP addresses at different times (enabling mobility) or to multiple addresses at the same time (enabling multihoming). Applications using HIP do not need to concern themselves with mobility and multihoming since the HIP implementations at the endpoints automatically discover and negotiate new addresses and change between them automatically, when necessary.

When two HIP hosts wish to communicate, they perform a four-way handshake, called HIP base exchange (BEX), which

978-1-4244-8790-5/11/$26.00 ©2011 IEEE

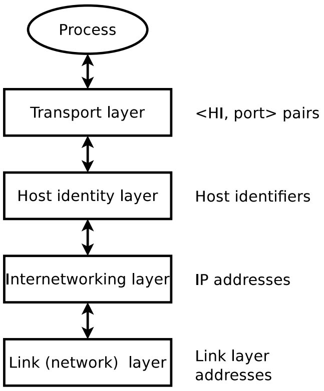

Fig. 1. TCP/IP stack with HIP

provides protection against Denial of Service (DoS) attacks and allows hosts to prove their identity. The identity used by HIP is the public key of an asymmetric cryptographic key pair. Hosts can prove that they own their identity during the base exchange using the private key of the key pair.

Since using a cryptographic key of an arbitrary length instead of an IP address is not feasible for most (especially legacy) applications and APIs, applications usually use a fixed-length IPv6-address-format compatible presentation of the identity called Host Identity Tag (HIT). HITs are a form of Overlay Routable Cryptographic Hash Identifiers (ORCHIDs) [6] and they are commonly generated by hashing the public key, and concatenating the ORCHID prefix and the result of the hash. Using a secure hash creates a secure binding between the identity and HIT and, thus, hashing is a common method in non-overlay environments. Nevertheless, any ORCHID (e.g., one generated randomly) can be bound to an identity using certificates.

During the HIP base exchange hosts perform a secure Diffie-Hellman key exchange. The key exchange enables the hosts to establish an IP Security (IPsec) Encapsulating Security Payload (ESP) connection that provides confidentiality and integrity protection for the data traffic between the hosts. Together the asymmetric key pair and IPsec provide the security for HIP's mobility and multihoming mechanisms.

III. Background on ICE

Interactive Connectivity Establishment (ICE) is a protocol that provides a complete NAT traversal solution by utilizing the functionality of Session Traversal Utilities for NAT (STUN) and Traversal Using Relays around NAT (TURN) protocols. A host using ICE needs to know the address of a STUN or TURN server and to have a signaling path with the host it wishes to communicate with. Connectivity between two hosts is tested with connectivity tests that also create NAT bindings (i.e., state required for sending and receiving packets through a NAT) in the NATs between the hosts.

In the beginning of the ICE processing a host gathers a set of transport addresses that can be possibly used to contact it. In addition to addresses from local interfaces, the host uses STUN to learn the address assigned to it by a NAT (if any) and TURN for allocating an address on a TURN server that can be used for relaying data. The signaling path is used for exchanging the gathered transport addresses, which are known as candidates. A host pairs up its own transport addresses with the transport addresses of the peer, which are called peer candidates, and arranges them in priority order, forming a list of possibly working candidate pairs. The pairs are prioritized so that the pairs that have a more direct path (i.e., less middleboxes between hosts) are preferred.

After forming the check list the hosts start performing connectivity checks. Checks are done by sending STUN requests on each pair in priority order and at the same time replying to checks sent by the peer. When a check succeeds on a candidate pair, that pair is considered valid for communications. The host that takes the controlling role of ICE processing chooses one of the valid pairs to be used for sending and receiving media.

If more than one transport layer port is needed (e.g., one for audio and one for video data), connectivity checks between the hosts need to be executed separately on each port. ICE uses an optimization for this case so that checks are first performed using one of the ports, and only when a check succeeds on some candidate pair, candidates using the same path with different port are tested.

IV. Architecture

In order to use HIP to perform connection management in P2PSIP, we needed to integrate HIP and P2PSIP in a coherent architecture. Our architecture provides three main functions: overlay maintenance, data storage and retrieval, and connection management. The first two functions are implemented using protocols different than HIP. These protocols are referred to as peer protocols. The connection management function is implemented using HIP.

Overlay maintenance consists of building and maintaining the routing tables of the overlay. The data storage and retrieval function allows storing and searching for objects in the overlay. The architecture supports any peer protocol that supports these two functions. Two examples of peer protocols are the Peer-to-Peer Protocol (P2PP) [7] and REsource LOcation And Discovery (RELOAD) [8] (both protocols can be used to build P2PSIP overlays, RELOAD being partially based on P2PP). The connection management function includes the functionality provided by HIP (which is discussed in Section II); that is, multihoming, mobility, security (authentication, DoS protection, integrity, and confidentiality), and NAT traversal.

Figure 2 shows the layers of the architecture. HIP signaling, and data connections established and managed by HIP form the base of the architecture. HIP signaling is used to establish connections between nodes in the overlay. These connections form the data transport layer in Figure 2.

Peer protocol messages are transported by HIP signaling messages. HIP signaling messages carry their destination

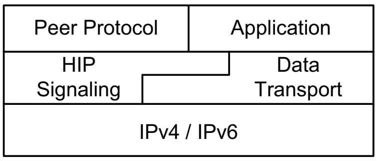

Fig. 2. Architecture layering

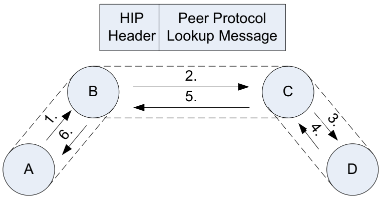

Fig. 3. Lookup procedure

ORCHID so that the nodes in the overlay can route the message towards its destination. HIP signaling messages carrying peer protocol messages are generally sent over an already-established data connection as part of the overlay's routing procedures. Application messages can be carried by a HIP signaling message (in the same way as peer protocol messages) or, more generally, straight over a direct data connection to its destination. The overlay's routing tables used to route HIP messages are built by the peer protocol.

Figures 3 and 4 illustrate how different messages are routed in an overlay. Let us assume that the application using this overlay is a Session Initiation Protocol (SIP) [2] telephony application. Node A wants to establish a voice call with node G. In order to do that, Node A needs to discover Node G's location and perform a SIP exchange with it. Such a SIP exchange, which establishes a voice call, involves three or more SIP messages.

Figure 3 shows how Node A uses the overlay to perform a lookup for Node G's location. The lookup message consists of a HIP header followed by a peer protocol message. The overlay's routing system routes the message to Node D, which is the node responsible for storing Node G's location. The dashed lines in Figures 3 and 4 represent existing connections between nodes. Those connections were established as part of the overlay's maintenance procedures. The lookup message is sent over those already-existing connections.

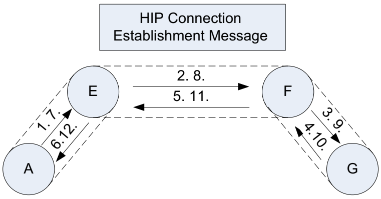

Fig. 4. Connection establishment procedure

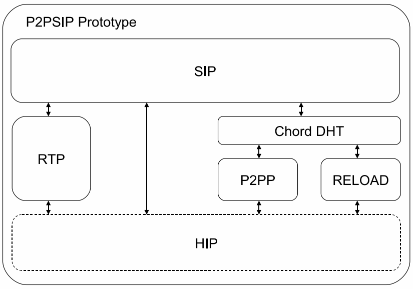

Fig. 5. Architecture of the P2PSIP prototype

Once Node A has Node G's location, Node A uses the overlay network to send a HIP connection establishment message to Node G, as shown in Figure 4. The HIP connection procedure consists of a four-way handshake, as described in Section II. These HIP messages are also sent using already-existing connections between nodes. The HIP handshake will establish a direct connection between Nodes A and G so that they can exchange messages directly, without using the overlay (i.e., messages will go from Node A to Node G directly without traversing Nodes E and F). Once such a direct connection is established, Nodes A and G perform the SIP exchange over it.

V. Implementation

In order to evaluate the proposed architecture, we have chosen to use a distributed voice over IP (VoIP) application that uses SIP to establish a single audio stream between two nodes. We have built a prototype of a P2PSIP system using the HIP-based architecture described in Section IV. The VoIP application chosen for our experiments includes media transfers using the real-time transport protocol (RTP).

Our P2PSIP prototype has been implemented in the Java programming language. It can be used to run existing SIP clients in a distributed mode by configuring them to use the P2PSIP prototype as a SIP outbound proxy. The P2PSIP prototype will then act as a SIP Back-to-Back User Agent (B2BUA) for the SIP client, relaying all SIP messages and RTP packets destined to and originating from the SIP client. In a typical setting, the SIP client and the P2PSIP prototype are running on the same host and communicate with each other via the local loopback interface.

The architecture of the P2PSIP prototype is illustrated in Figure 5. The prototype consists of different modules, including SIP, RTP, Chord, P2PP, RELOAD, and HIP modules. The SIP module consists of a SIP stack and SIP B2BUA implementation. The RTP module is a simple media proxy that relays RTP packets to and from the SIP client. The Chord module implements the Chord DHT algorithm [9] used to organize the topology of interconnections amongst peers in the P2PSIP overlay and to maintain the overlay. The Chord module can use two different peer protocols, P2PP and

TABLE I

IMPLEMENTATION COSTS

Work item	With HIP		Without HIP	
	LoC	**Hours**	**LoC**	**Hours**
ICE	8000	960	7000	840
P2PSIP-HIP interface	1300	160	0	0
ICE-HIP integration	1000	320	0	0
ICE-SIP integration	0	0	1500	180
ICE-RTP integration	0	0	1200	160
ICE-P2PP integration	0	0	1300	160
ICE-Chord integration	0	0	4600	550
Total	10300	1440	15600	1890

RELOAD. These peer protocols are implemented by the P2PP and RELOAD modules. In our evaluation, which is presented in Section VI, we have used the P2PP module.

The HIP module runs as a Linux operating system service. The HIP module has been implemented using the C programming language. Other modules interact with the HIP module through a Java Native Interface (JNI) based Application Programming Interface (API). When the HIP module is enabled, the prototype uses HIP to establish IPsec connections for the SIP, RTP, and P2PP protocols. NAT traversal functionality is provided by HIP, meaning that there is no need to implement NAT traversal in the other protocol modules. Additional benefits of using HIP include support for mobility and multihoming.

When the HIP module is disabled, each protocol module needs to implement NAT traversal functionality separately. For this, we implemented an ICE library in Java that was integrated into the P2PP, SIP, and RTP modules. In addition, extra logic was added to the Chord module to handle among other things STUN and TURN server discovery, connection management, exchange of ICE candidates across the overlay, triggering of ICE connectivity checks, and the possibility to use the peer protocol to establish ICE-negotiated connections for SIP.

Table I lists our implementation costs in terms of lines of code and person hours of the different work items when we integrated HIP and ICE in the whole system. Unsurprisingly, the difference in cost of implementing the ICE library for HIP in the C programming language compared to implementing the ICE library in Java was fairly small (the cost of implementing the former was slightly higher), as Table I shows.

VI. EVALUATION

Our implementation allows us to build overlay networks with and without HIP. Using the same peer protocol with and without HIP allows us to analyze the effects of using our architecture in terms of efficiency and performance. When HIP is used for NAT traversal, ICE is run once between two hosts, an IPsec ESP tunnel is established on the best working path, and all subsequent communication between the hosts uses the IPsec ESP tunnel. On the other hand, without HIP, ICE needs to be run separately for each data and signaling flow, and each of the flows needs to be secured independently.

We built P2PSIP overlays with different types of NATs and measured the resulting session establishment delays [3].

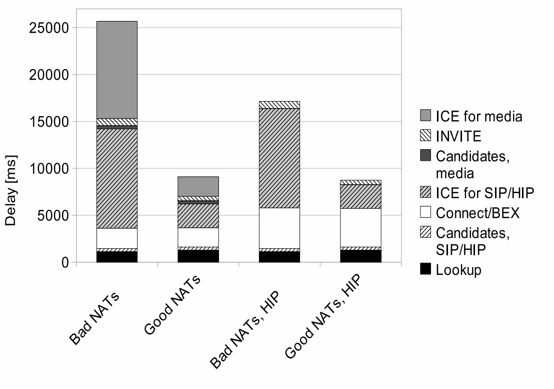

Fig. 6. Components of the session establishment delay in P2PSIP with and without HIP

Figure 6 shows the different components of the session establishment delay. In the figure, Bad NATs refer to NATs whose filtering characteristics force nodes to use a TURN relay for communication. Traversing Good NATs does not require the use of TURN relays (see [3] for a more thorough analysis on P2PSIP session establishment delays in the presence of NATs and full details on our experimental setup).

Without HIP (two left columns in Figure 6), when a session is set up between two nodes, the caller uses P2PP for retrieving the location of the callee ("Lookup" component) and establishing a direct connection with the callee ("Connect" component). The nodes will use this direct connection to exchange SIP messages. In order to establish the direct connection, the nodes need to run the ICE procedures between them, which include gathering candidates ("Candidates, SIP" component) and exchanging connectivity checks ("ICE for SIP" component). Once the direct connection is established, the nodes use a SIP INVITE transaction ("INVITE" component) to establish media streams. The establishment of media streams also involves running the ICE procedures between the nodes ("Candidates, media" and "ICE for media" components).

Using ICE to establish an RTP-based media stream involves finding connectivity for an RTP stream and an RTP Control Protocol (RTCP) stream, each of them running on a separate transport-layer port. Depending on the types of the NATs between the hosts and how persistently ICE is configured to try to find the best path, both hosts end up sending from three up to more than ten messages, and spend from the time of two RTTs to more than ten seconds[1] for each protocol [11].

The use of multiple transport layer ports in a non-HIP case (one for SIP plus several for media) increases the amount of messages and time needed for the ICE processing compared to the single-port approach used by HIP. We built P2PSIP overlays in presence of NATs using HIP. Figure 6 shows that when HIP is used in a P2PSIP overlay (two right columns

[1] For example, the Microsoft ICE implementation [10] uses by default 10 second timer for ending the checks if no good paths are available

in the figure), the "Candidates, media" and "ICE for media" (see Figure 6) components completely disappear. As can be observed in the figure, removing these two components yields a significant reduction in the ICE-related part of the session establishment delay (we discuss the increase in the "Connect" component later in this section). Additionally, HIP's single port approach reduces the number of messages needed to establish and maintain the session.

In the candidate gathering phase without HIP, a connection to a STUN or TURN server needs to be created separately for each of the ports, requiring one additional message exchange per port. Furthermore, each of the server connections needs to be kept alive with regular keepalive messages for the duration of the connectivity checks. During the connectivity establishment phase one to three additional connectivity checks for each additional port are needed, despite of the optimization ICE uses for the checks in case of multiple ports for a single media stream. The amount of additional checks per port needed is not fixed but it depends, e.g., on the timing of the checks the peers are performing and on the type of the working candidate pair. After the connections have been established, each of them needs to be kept alive using regular keepalive messages.

In addition to saving bandwidth and time, because HIP provides security for all the connections, there is no need for applications to use other security protocols such as Transport Layer Security (TLS) or Datagram Transport Layer Security (DTLS). Without HIP one would create separate TLS or DTLS connections for each flow, requiring as many additional TLS handshakes and authentication cryptographic operations as there are connections. The HIP handshake is performed only once between the hosts and all communication can re-use the created security association.

In our experiments, we did not use any security for SIP or for the media stream in the non-HIP case, though. Therefore, we saw a 100% increase in the "Connect" component. The reason for this increase in the delay is that while P2PP uses a two-way handshake to establish direct connections, HIP uses a four-way handshake (BEX) in order to prevent DoS attacks against the receiver of the connection. As stated before, this extra delay (which consists of a round trip between the nodes through the overlay) is generally lower than the time needed to establish TLS and DTLS connections for SIP and the media streams in the non-HIP case. Nevertheless, in scenarios where SIP and the media streams do not need to be secured for some reason, this extra delay could be considered a performance problem. As Figure 6 shows, in our experiments this extra delay was similar to the performance gain brought by HIP's single port approach in Good NATs scenarios. Therefore, overall, HIP's effect was negligible in the session establishment delay. In Bad NATs scenarios, the performance gain brought by HIP's single port approach was significantly larger than the extra delay resulting from the HIP four-way handshake. Therefore, HIP's effect resulted of a significant reduction in the session establishment delay.

Since HIP tunnels all the data over UDP and IPsec ESP, it slightly increases the amount of bandwidth required for each packet. However, with Bound End-to-End Tunnel (BEET) mode [12] the overhead is similar to that of TLS and thus does not increase bandwidth requirements in practice.

VII. Conclusions

Our experiments show that HIP can be used to manage groups of connections in P2PSIP overlays reducing the ICE-related components of the session establishment delay. The main advantage of such grouping is that ICE procedures performed for one connection can be reused when establishing other connections. On the other hand, HIP's four-way handshake adds delay to the "Connect" component of session establishments. In environments that do not require security for SIP or the media streams, and have NATs with good filtering properties, the overall effect of HIP in the session establishment delay is negligible. In all other scenarios, the use of HIP results in a significant reduction in the session establishment delay.

References

[1] J. Mäenpää and G. Camarillo, "Analysis of Delays in a Peer-to-Peer Session Initiation Protocol Overlay Network," in *Consumer Communications and Networking Conference (CCNC), 2010 7th IEEE*, Jan 2010.

[2] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session Initiation Protocol," RFC 3261 (Proposed Standard), Internet Engineering Task Force, Jun. 2002.

[3] J. Mäenpää, V. Andersson, G. Camarillo, and A. Keränen, "Impact of Network Address Translator Traversal on Delays in Peer-to-Peer Session Initiation Protocol," in *Accepted to IEEE Global Communications Conference (GLOBECOM), 2010*, Dec 2010.

[4] J. Rosenberg, "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols," RFC 5245 (Proposed Standard), Internet Engineering Task Force, Apr. 2010.

[5] R. Moskowitz, P. Nikander, P. Jokela, and T. Henderson, "Host Identity Protocol," RFC 5201 (Experimental), Internet Engineering Task Force, Apr. 2008.

[6] P. Nikander, J. Laganier, and F. Dupont, "An IPv6 Prefix for Overlay Routable Cryptographic Hash Identifiers (ORCHID)," RFC 4843 (Experimental), Internet Engineering Task Force, Apr. 2007.

[7] S. Baset, H. Schulzrinne, and M. Matuszewski, "Peer-to-Peer Protocol (P2PP)," Internet Engineering Task Force, Internet-Draft draft-baset-p2psip-p2pp-01, Nov. 2007, work in progress.

[8] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset, and H. Schulzrinne, "REsource LOcation And Discovery (RELOAD) Base Protocol," Internet Engineering Task Force, Internet-Draft draft-ietf-p2psip-base-08, Mar. 2010, work in progress.

[9] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications," *IEEE/ACM Trans. Netw.*, vol. 11, no. 1, pp. 17–32, 2003.

[10] "[MS-ICE]: Interactive Connectivity Establishment (ICE) Extensions," http://msdn.microsoft.com/en-us/library/cc431495.aspx, Referenced on 28.6.2010.

[11] A. Keränen, "Host Identity Protocol-based Network Address Translator Traversal in Peer-to-Peer Environments," Master's thesis, Helsinki University of Technology, Sep 2008.

[12] P. Nikander and J. Melen, "A Bound End-to-End Tunnel (BEET) mode for ESP," Internet Engineering Task Force, Internet-Draft draft-nikander-esp-beet-mode-09, Aug. 2008, work in progress.