Traversing Middleboxes with the Host Identity Protocol

Hannes Tschofenig[1], Andrei Gurtov[2], Jukka Ylitalo[3], Arthi Nagarajan[4], and Murugaraj Shanmugam[4]

[1] Siemens, Germany, hannes.tschofenig@siemens.com,
[2] Helsinki Institute for Information Technology, Finland, gurtov@cs.helsinki.fi,
[3] Ericsson Research NomadicLab, Finland, jukka.ylitalo@nomadiclab.com,
[4] Technical University Hamburg-Harburg, Germany,
{murugaraj.shanmugam}@tu-harburg.de

Abstract. The limited flexibility of the Internet to support mobility has motivated many researchers to look for alternative architectures. One such effort that combines security and multihoming together is the Host Identity Protocol (HIP). HIP is a signaling protocol that adds a new protocol layer to the Internet stack between the transport and the network layer. HIP establishes IPsec associations to protect subsequent data traffic. Though the security associations are established solely between the communicating end hosts, HIP also aims to interwork with middleboxes such as NATs and firewalls. This paper investigates this interworking aspect and proposes a solution for secure middlebox traversal.

Keywords: Identifier-Locator Split, Host Identity Protocol, Middlebox, Network Address Translators (NATs), Firewalls, Authentication, Authorization.

1 Introduction

In the classical Internet architecture, an IP address serves as an address for packet delivery and as an identifier for the communicating end points. These roles are known as the locator and identifier respectively. The dual use of an IP address, although originally intended, nowadays limits the flexibility with regard to mobility and multihoming. In recent years, there have been many efforts to overcome this limitation through different approaches at different layers in the protocol stack. Existing solutions propose new indirection infrastructures, transport layer enhancements to support multiple locators, or adding new shim protocol layers. This paper looks at the compatibility issues of the Host Identity Protocol with NATs or firewalls and proposes a generic middlebox security solution.

The Host Identity Protocol (HIP) [1] is being developed by the IETF HIP working group. It is an identifier-locator separation mechanism that operates between the transport layer and the network layer. The Host Identity Protocol heavily relies on public key cryptography where every host generates a pair of keys: a private key and a public key. The public key is called the Host Identity (HI). A Host Identity Tag (HIT) is a 128-bit hash of the host's public key. The interface to the transport layer uses Host Identity Tags in place of IP addresses, while the interface to the Internet layer uses conventional IP addresses. In simple terms, transport connections and security associations are bound to HITs that do not change with changes of IP addresses. HIP is initialized with a base exchange mechanism that is used to quickly authenticate the hosts, exchange the keys to protect the rest of the base exchange and to form the required security associations to protect the payload.

HIP [1] starts with one of the hosts looking up the HI and IP of the peer in the DNS. The host then sends an initial I1 message requesting a state to be established with the peer. Messages R1, I2 and R2 are exchanged successively in order to create an association.

Once the base exchange is completed, the data traffic between the communicating hosts is protected using IPsec. When one of the hosts changes its IP address, the new address needs to be updated with the peer. For this purpose, HIP uses a readdressing procedure. Additionally, readdressing can be accompanied with a new SPI value and/or new keys for the existing security association.

All packets except the base exchange and readdressing messages are protected using IPsec ESP. IPsec has traditionally been known to be a Network Address Translation (NAT) sensitive protocol. To allow IPsec protected traffic to traverse a NAT, it is either possible to provide UDP encapsulation [5] or to allow the NAT to participate in the signaling message exchange. A mechanism to detect a NAT along the path between two IPsec endpoints has be provided for IKEv1 [4] and has been incorporated into IKEv2 [6]. Additionally, firewall traversal faces routing asymmetry problems. A number of IETF working groups such as the MIDCOM, PANA and NSIS [13] have encountered this problem.

2 Problem Statement

Most networks today still use IPv4 addresses even though IPv6 is ready for deployment. Apart from the communicating end hosts, many middleboxes are also present between the hosts on the network, each meant for a specific functionality. For instance, to combat the IPv4 address depletion problem, private networks use NATs [12] to reuse and share global IPv4 addresses. For security reasons, firewalls are placed at the border of a network. When HIP is deployed into an existing network, NATs need to be retained for the sake of already existing IPv4 applications. For security reasons, HIP will need to deal with firewalls as well.

In the current Internet, IP addresses are used both for identifying hosts and identifying their topological locations. This semantic overloading is deeply related to most of well-known NAT problems [9]. IPsec is an example of a protocol that suffers from the related NAT traversal problems [10]. UDP encapsulation of IPsec packets allows a NAPT[12] to modify the UDP header and to perform the demultiplexing [4]. Unfortunately, the approach unnecessarily increases the packet size and may cause configuration difficulties, e.g., in firewalls.

In this proposal we try to address the following functionalities that are expected of a HIP aware NAT or firewall:

1. Interception : IPsec use <Destination IP, Destination SPI, Protocol >to identify a particular security association. Middleboxes can also be thought to use the same flow identifier information for a flow. This can be achieved by making the NAT/FW HIP aware and to intercept the SPI values carried within HIP signaling messages.
2. Authentication : Many middlebox traversal mechanisms do not have any security at all. A HIP aware NAT/FW must be able to authenticate the requesting HIP nodes before creating a NAT binding or a firewall pinhole.
3. Authorization : A HIP aware NAT/FW must be able to authorize the requesting HIP nodes using identity dependent or identity independent methods. A potential solution must respect the property of the middleboxes before roaming outside the network.
4. Denial of Service attack resistance : The authentication and authorization mechanisms should not introduce new DoS attacks at the middlebox.
5. Registration Procedure - A firewall might require authentication and authorization of one of the end points prior to allowing signaling (and data traffic) to bypass. Depending on the architecture and environment, this protocol step might be required.
6. Avoiding unwanted traffic : In the wireless environment an end host might want to stop receiving unwanted traffic. A signaling protocol is needed to indicate what traffic to receive and what traffic to drop. It must also be assumed that end-to-end communication is not always possible prior to the interaction of the end hosts.
7. Soft-state Nature : To deal with failures and route changes, it is important to design a protocol in such a way that the state allocated at middleboxes times out after a certain period of time. Periodic transmission of refresh messages is therefore required. SPI multiplexed NAT (SPINAT) is an example of a HIP aware NAT that uses HIP to establish a NAT binding and to establish the security state [7].

3 HIP and NAT/FW Traversal

This section describes our proposal for traversing middleboxes with HIP. We use HIP as a protocol to communicate with middleboxes.

3.1 HIP base exchange and NAT

A HIP aware NAT/FW needs to inspect the HIP base exchange to learn the <Destination IP, Destination SPI, Protocol>triplet for a specific host. The HIT values are also required and can subsequently be used to verify future signaling messages. The approach presented in [7] is also relevant here which requires the usage of hash chains to update the binding in a HIP aware NAT device.

All HIP messages carry a standard HIP header with the HIT of the initiator and the HIT of the receiver. It must be noted that IPsec SAs are unidirectional and hence two SPI values (for the Initiator and for the Responder) need to be negotiated. Subsequently, message I2 carries the SPI value of the Initiator, SPI(I), and message R2 carries the SPI value of the Responder, SPI(R). For authorization, SPKI certificates [2] or SAML assertions [3] may turn out to be useful since the Host Identities might be ephemeral and anonymity for the end hosts is an important aspect. Providing authorization based on information in the SPKI certificates or SAML assertions can be used to enable the middlebox to execute the necessary protocol actions (e.g., opening a pinhole) without the need for authentication.

3.2 HIP base exchange and firewalls

NATs establish state and modify IP address information and thereby force IP packets to flow through also in the reverse direction. This makes the interception mechanism for NATs much easier compared to that of the firewalls. In the presence of a generic middlebox (or firewalls in particular) or a topology with a mixture of NATs and firewalls, routing asymmetry needs to be considered. Figure 1 shows a HIP exchange through a firewall. In firewalls, forward paths may differ from the reverse paths. Then, messages I1 and I2 from the initiator to the receiver take a different path from messages R1 and R2 sent from the receiver to the initiator. For instance, the Initiator generates its SPI(I) and sends it to the Responder in a message I2 through FW(R). However, FW(I) needs this information to create the state for the Initiator. Similarly, the Responder generates its SPI(R) and sends it to I in the R2 message through FW(I). However, FW(R) needs to create the flow identifier information for R as shown in Figure 1.

Hence, new solutions need to be provided for tackling the routing asymmetry problem with respect to the firewalls and flow identifier interception. These solutions have to be handled without changing the existing HIP base exchange significantly.

3.3 HIP readdressing, re-keying and NAT/FW

Even after the HIP base exchange is finished, a NAT/FW still needs to keep updating its state for the flow identifier in case an IP address or an SPI value changes for an end host. For example, whenever a HIP end point is mobile and informs its peer about the new IP address, the states at FW(I) and FW(R) also need to be updated. Additionally, if the hosts decide to choose a new SPI value for the same security association or a new pair of keys along with the readdressing, routing asymmetry may cause additional complications. Middleboxes must authorize state modifications to avoid a number of attacks including redirection, black holing or third party flooding. A desired property in this case is sender invariance, which states: “A party is assured that the source of the communication has remained the same as the one that started the communication, although the actual identity of the source is not important to the recipient.”(Section 3 of [8]).

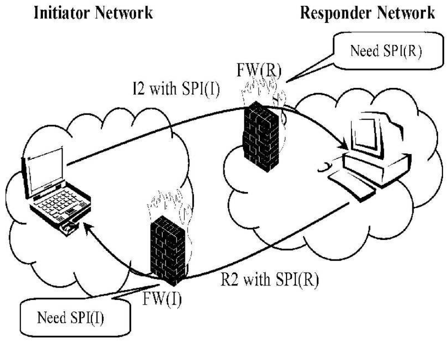

Fig. 1. Routing asymmetry with firewalls.

4 HIP aware NAT/FW

Many middleboxes today do not support any security. State is created based on data traffic without authentication, authorization or DoS protection. The complexity to support different types of NAT/FWs influences the design of the protocol to a certain extent. The middlebox could fall into some of the following categories:

1. A NAT/FW could support only the present Internet Protocol and can be completely incompatible with HIP. These falls into the category of "HIP-unaware NAT/FW "that does not require security capabilities.
2. A "Transparent NAT/FW "could need weak authentication techniques security for simple state establishment, for instance, using the SPINAT functionality. However, here the base exchange becomes vulnerable to a DoS attack because the initiator's HI is encrypted in the I2 packet and the NAT/FW box is unable to verify the I2 message. As a consequence, an attacker may send a spoofed I2 message before the authentic initiator does that. The spoofed I2 message may contain a spoofed SPI value resulting in an inconsistent state at NAT/FW. The problem can be solved, either by including the initiator's SPI value both to the I1 and I2 messages or sending the initiator's HI as plain text in I2 packet. While the former solution creates a state at the NAT/FW and the peer host even before the puzzle is solved, the later interferes with anonymity. Fortunately, the NAT/FW may verify the responder's SPI in R2 packet with signature, because responder's HI is sent in plain text.
3. A third set of NAT/FW may opt to complete authentication and authorization before establishing state for a host. These are the "Registration Requiring NAT/FW "that run a registration protocol, a variant of the HIP base exchange between the end host and the middlebox.

4.1 The HIP registration protocol

To introduce a new registration protocol, it is necessary to deal with the general protocol design issues such as mutual authentication capability, Denial of Service attack resistance and efficiency in the number of roundtrips. Furthermore, it is helpful if the end-to-end protocol and the registration protocol support the same credentials. These requirements motivate to reuse the HIP protocol for the purpose of authentication, authorization and the establishment of a security association. However, it should be noted that the establishment of an IPsec security association is not necessary here.

To deal with mobility it is necessary to periodically refresh the state at the firewall. The update of packet filters can either be sent directly to the firewall or indirectly with the help of an end-to-end HIP exchange. The former might be necessary for a data receiver installing packet filters to prevent unwanted traffic from consuming an expensive wireless resource where the data receiver might get charged for.

Factors giving an advantage to the HIP registration protocol are follows:

1. Reuses the same puzzle mechanism to prevent Denial of Service attacks.
2. The Initiator has to solve the puzzle in order to prove its interest in a successful protocol exchange. This allows the Responder to delay state creation until receiving I2. The puzzle is made up of the corresponding HITs and a random number; the difficulty of the puzzle can be increased based on the trust of the Initiator. This cookie mechanism prevents the Responder from some Denial of Service attacks.
3. Provides an end-to-end authentication, using signature verifications.
4. Both the Initiator and the Responder can authenticate each other; Initiator authenticates Responder in the R1 packet by verifying the signature using HI(R) and the Responder authenticates the Initiator by verifying the signature of the I2 packet using HI (I).
5. Uses HMAC to protect the integrity of the messages and prevents DoS using signature verifications.
6. Both the peers obtain the shared secret key and calculate the corresponding derived keys using the authenticated Diffie-Hellmann exchange. Responder uses one of the keys to calculate HMAC in the R2 packet in order to prove the key confirmation.
7. Uses SPKI certificates (or SAML assertions) for authorization.
8. The Initiator may send the authorization certificate immediately after the I2 message, to be authorized by the middlebox. This is a significant improvement in design of the middleboxes, as currently most middleboxes do not provide authorization.

4.2 SPISIG message

The generic registration protocol that we have introduced can be used for all middleboxes that require authentication and authorization for a host-middlebox binding. This is mostly the case for NATs and firewalls at network borders for outgoing traffic. However, the firewall for the incoming traffic needs to maintain state information for the host to forward its packets. The registration protocol can be reused here between the incoming traffic firewall and the host to make sure that the firewall maintains the proper state for the legitimate host. Even after the registration, the state is still not complete as FW(R) is unable to intercept SPI(R) sent in R2 and FW(I) is unable to intercept SPI(I) sent in I2 as was shown in Figure 1.

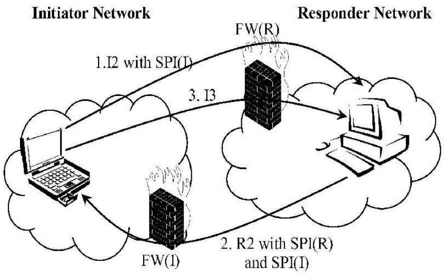

Fig. 2. Extending the base exchange with I3.

One possible solution to this problem could be following. Once the Responder receives the SPI (I) in message I2, it could resend the SPI (I) along with SPI(R) in message R2. This could help the FW (I) intercept the SPI (I) information. Since the receiver R has to remain stateless until the solution in I2 is verified, the SPI(R) cannot be sent in R1 and hence not resent in I2. The only other option would be to create a new message I3 as that carries the SPI(R) from the Initiator to the Responder such that all middleboxes in the path can intercept and form the flow identifier information for the receiver. However, such a solution of changing the base exchange messages for the sake of firewall traversal is unsatisfactory and undesired.

$I \rightarrow FW(I) \rightarrow R : I1 \subset Trigger\ exchange$
$I \leftarrow FW(R) \leftarrow R : R1 \subset$
$Puzzle, \{DH(R), HI(R), HIP_{Transform}, ESP_{Transforms}\}SIG$
$I \rightarrow FW(I) \rightarrow R : I2 \subset$
$\{Solution, SPI(I), DH(I), HIP_{Transform}, ESP_{Transform}, \{H(I)\}\}SIG$
$I \leftarrow FW(R) \leftarrow R : R2 \subset \{SPI(R), SPI(I), HMAC\}SIG$
$I \rightarrow FW(I) \rightarrow R : I3 \subset \{SPI(R), HMAC\}SIG$

An alternative solution could be that once the base exchange is complete and a state is established at the communicating HIP hosts, the local host could signal its firewall in a SPISIG message about the SPI value that it has chosen for the particular security association. The firewall would have already intercepted the IP and HIT values from the initial messages of the base exchange. It can then create the flow identifier information using the SPI value that it obtains from the hosts within the private network.

5 Formal Analysis

The protocol has been analyzed by means of formal method analysis using the High Level Protocol Specification Language (HLPSL) - an expressive language for modeling communication and security protocols.

We used the tool OFMC[1] (On-the-Fly Model-Checker), from the AVISPA project [15] (Automated Validation of Internet Security Protocols and Applications"), which uses a rich specification language for formalizing protocols, security goals, and threat models of industrial complexity.

The HIPSL file was then translated into an Intermediate format using another tool named HLPSL2IF, which is a translator, which maps security protocol specifications into rewriting systems. This intermediate format can be executed to analyze the threats of the protocol. From the results, which we got, no attacks were found for the following attacks:

- Man in the Middle Attack (MitM)
- Denial of Service attack (DoS)
- Replay Attack
- Server Authentication to the client (server spoofing)
- Client Authentication to the server (client spoofing)

5.1 Informal Analysis

The HIP registration protocol uses an authenticated Diffie-Hellmann Key Exchange and generates session keys to defend against the Man-in-the-Middle attacks. The Initiator provides key confirmation in the I2 packet by encrypting the Host Identity and the Responder performs key confirmation by sending the HMAC in the R2 packet. When the Initiator chooses anonymous HIs, the protocol suffers from the Man-in-the-Middle attacks. The usage of authorization certificates provides a solution for this purpose but the formal modeling tool will produce an error.

This protocol also provides some protection for the Initiator, since the messages from the Responder are signed. One potential problem could be the following case: R1 message contains the signature and the Initiator, first, has to verify it. Here the Intruder might try some DoS attacks. But in order to launch this attack the Intruder has to act as a Man-in-the-Middle adversary and act quickly to send spoofed R1 packets.

[1] The tool is available on-line at http://www.avispa-project.org/web-interface/

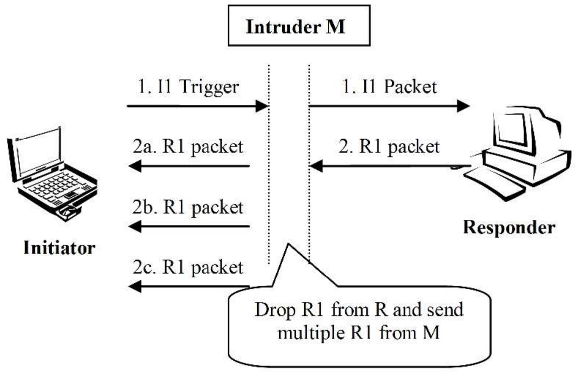

Fig. 3. Sending multiple bogus R1 packets.

After the I2 message, the Responder may wait for the certificates. Here the Intruder can send some bogus certificates with signatures and forcing the Responder to verify, this might cause DoS attacks. This kind of attack can be resisted, if the responder is designed not to accept more than one certificate during the base exchange.

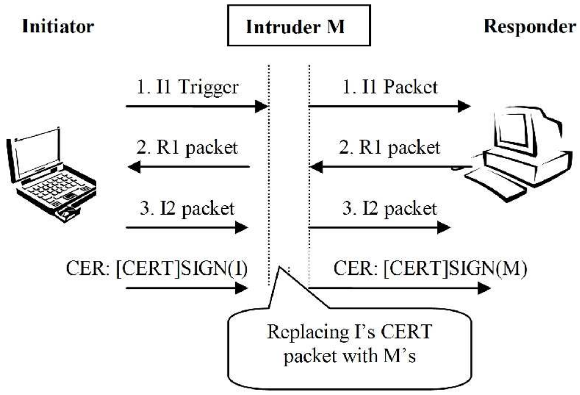

Fig. 4. Sending bogus certificates.

This protocol uses an R1 counter to protect against replay attacks. The R1 generation counter is a monotonically increasing 64-bit value and this counter indicates the current generation of puzzles. The system can avoid replay attacks by simply increasing the value of the counter to show the validity of the packet.

The Initiator can check the counter to determine whether it received a new high counter value or not. The server authenticates the client in the R1 packet by sending his HI in clear text and also signs the message. The Initiator can verify the HI and signature as it knows the Responder's public key/HI from the DNS look up.

Client authentication to the server can be done because the server verifies the Initiator's Public key/HI with the received HIT. Since HIT is the Hash of the HI, after the receiving the I2 packet, the Responder can verify the Initiator's identity by cross checking the HIT and HI.

Thus, the registration protocol provides enough resistance to protect against the above listed attacks.

5.2 Implementation

We have implemented a prototype for the registration protocol [5]. For simplicity the current implementation assumes that the Initiator obtained the SPKI certificate using an out-of-band mechanism [6].

We found out that the minimum memory needed for storing the state information at a middle-box is 2286 bytes. The approximate time taken for each packet is summarized in table below. Computing the Diffie-Hellman derived session key takes almost 80% of the time and signature verification takes 10% of the time.

Packets	I1 (ms)	R1 (ms)	I2 (ms)	R2 (ms)
Creation	0.030	15	95	25
Processing	0.007	300	75	15

Table 1. Time taken for the packets.

A more detailed performance investigation is in progress. To establish an arbitrary number of HIP sessions and to check the throughput and packet loss requires some protocol enhancements. The current implementation establishes a state, if there is a change in the IP address or in the HIT. Changing the IP address or HIT for high-performance tests does not seem to be adequate. seems really difficult in a short interval of time. A different session identification (added for testing purpose to the HIP registration protocol) allows creating an arbitrary number of concurrent exchanges.

[5] We used two Pentium II 266 Mhz Linux based machines as an Initiator and the Responder (Middlebox), both of them residing in a single LAN.

[6] The throughput between the Initiator and Responder, (measured by using ttcp) was 8.5 Mbps, the round trip time was 0.16 ms (measured with ping) and the average time taken to complete the registration was approximately 0.94 seconds.

6 Conclusions

For a long time the focus of HIP was on solving problems affecting mainly the endpoints. In future, the IETF HIP research group [17] will also address the middlebox traversal problem for HIP. To avoid including a HIT into every data packet and to provide end-to-end protection of data traffic, IPsec ESP is used between the end points. Unfortunately, IPsec protected data traffic is known to cause problems with middleboxes (particularly with regard to NAT traversal). Middleboxes need to participate in the HIP signaling exchange to allow these devices to perform their function. This interaction requires certain security goals to be met. A solution can be complicated by a number of factors including routing asymmetry, combination of different types of middleboxes and state updates due to mobility. Our proposal tries to raise the attention of the community based on a simple protocol proposal.

To enable HIP-aware middleboxes, we use a registration procedure. The registration procedure reuses the common base exchange mechanism, removing the ESP transforms and SPI fields. Authorization functionality is added using SPKI certificates or SAML assertions. It is a first step toward deployment of HIP friendly NATs and firewalls that performs their functionality with enhanced security.

7 Acknowledgements

This document is a by-product of the Ambient Networks Project, partially funded by the European Commission under its Sixth Framework Programme. It is provided “as is ”and without any express or implied warranties, including, without limitation, the implied warranties of fitness for a particular purpose. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Ambient Networks Project or the European Commission.

References

1. Moskowitz R., Nikander P., Jokela P. and Henderson T., Host Identity Protocol draft-ietf-hip-base-01.txt (work in progress), October 2004.
2. Ellison C., Frantz B., Lampson B., Rivest R., Thomas B. and Ylnen T., SPKI Certificate Theory . RFC 2693, September 1999.
3. Maler, E., Philpott R., and Mishra P., Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V1.1 , September 2003.
4. Kivinen, T., Swander, B., Huttunen, A. and Volpe, V., Negotiation of NAT-Traversal in the IKE, RFC 3947, January 2005.
5. Huttunen A., Swander, B., Volpe, V., DiBurro, L. and Stenberg M., UDP Encapsulation of IPsec ESP Packets . RFC 3948, January 2005.
6. Kaufman, C., Internet Key Exchange (IKEv2) Protocol. draft-ietf-ipsec-ikev2-17.txt (work in progress), September 2004.

7. Ylitalo, J., Melen, J., Nikander, P. and V. Torvinen Re-thinking Security in IP based Micro-Mobility 7th Information Security Conference (ISC-04), Palo Alto, September 2004.
8. Automated Validation of Internet Security Protocols and Applications (AVISPA) IST-2001-39252, Deliverable v1.0, November, 2003.
9. Moore K., Things that NATs break Unpublished, http://www.cs.utk.edu/ moore/what-nats-break.html, October. 2003.
10. Aboba B. and Dixon W., IPsec-Network Address Translation (NAT) Compatibility Requirements RFC 3715, March 2004.
11. Ylitalo, J., P. Jokela, J. Wall and P. Nikander., End-point Identifiers in Secure Multi- Homed Mobility in Proc. of the 6th International Conference On Principles Of DIstributed Systems (OPODIS 02), pp. 17-28, France, Dec., 2002.
12. Giving K. and Francis P., Network Address Translator RFC 1631, May 1994.
13. Next Steps in Signaling (nsis) Working Group Charter http://www.ietf.org/html.charters/ nsis-charter.html (February 2005).
14. Kent S. and Atkinson R., IP Encapsulating Security Payload, RFC2406, November 1998.
15. Automated Validation of Internet Security Protocols and Applications Webpage, http://www.avispa-project.org/, (February 2005).
16. Kent, S. and Seo K., Security Architecture for the Internet Protocol, draft-ietf-ipsec-rfc2401bis-05.txt, (work in progress), December 2004.
17. Host Identity Protocol (HIP) IRTF Research Group, http://www.irtf.org/charters/hip.html (February 2005)
18. Jokela P, Moskowitz R, Nikander P, Using ESP format with HIP draft-jokela-hip-esp-00.txt (work in progress), Febrauary 2005.