Stream: Internet Engineering Task Force (IETF)
RFC: 9063
Obsoletes: 4423
Category: Informational
Published: July 2021
ISSN: 2070-1721
Authors: R. Moskowitz, Ed. *HTT Consulting*
M. Komu *Ericsson*

RFC 9063
Host Identity Protocol Architecture

Abstract

This memo describes the Host Identity (HI) namespace, which provides a cryptographic namespace to applications, and the associated protocol layer, the Host Identity Protocol, located between the internetworking and transport layers, that supports end-host mobility, multihoming, and NAT traversal. Herein are presented the basics of the current namespaces, their strengths and weaknesses, and how a HI namespace will add completeness to them. The roles of the HI namespace in the protocols are defined.

This document obsoletes RFC 4423 and addresses the concerns raised by the IESG, particularly that of crypto agility. The Security Considerations section also describes measures against flooding attacks, usage of identities in access control lists, weaker types of identifiers, and trust on first use. This document incorporates lessons learned from the implementations of RFC 7401 and goes further to explain how HIP works as a secure signaling channel.

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 candidates for any level of Internet Standard; see Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9063.

Copyright Notice

Table of Contents

1. Introduction

The Internet has two important global namespaces: Internet Protocol (IP) addresses and Domain Name Service (DNS) names. These two namespaces have a set of features and abstractions that have powered the Internet to what it is today. They also have a number of weaknesses. Basically, since they are all we have, we try to do too much with them. Semantic overloading and functionality extensions have greatly complicated these namespaces.

The proposed Host Identity namespace is also a global namespace, and it fills an important gap between the IP and DNS namespaces. A Host Identity conceptually refers to a computing platform, and there may be multiple such Host Identities per computing platform (because the platform may wish to present a different identity to different communicating peers). The Host Identity namespace consists of Host Identifiers (HI). There is exactly one Host Identifier for each Host Identity (although there may be transient periods of time such as key replacement when more than one identifier may be active). While this text later talks about non-cryptographic Host Identifiers, the architecture focuses on the case in which Host Identifiers are cryptographic in nature. Specifically, the Host Identifier is the public key of an asymmetric key pair. Each Host Identity uniquely identifies a single host, i.e., no two hosts have the same Host Identity. If two or more computing platforms have the same Host Identifier, then they are instantiating a distributed host. The Host Identifier can either be public (e.g., published in the DNS) or unpublished. Client systems will tend to have both public and unpublished Host Identifiers.

There is a subtle but important difference between Host Identities and Host Identifiers. An Identity refers to the abstract entity that is identified. An Identifier, on the other hand, refers to the concrete bit pattern that is used in the identification process.

Although the Host Identifiers could be used in many authentication systems, such as IKEv2 [RFC7296], the presented architecture introduces a new protocol, called the Host Identity Protocol (HIP), and a cryptographic exchange, called the HIP base exchange; see also Section 6. HIP provides for limited forms of trust between systems, enhances mobility, multihoming, and dynamic IP renumbering, aids in protocol translation and transition, and reduces certain types of denial-of-service (DoS) attacks.

When HIP is used, the actual payload traffic between two HIP hosts is typically, but not necessarily, protected with Encapsulating Security Payload (ESP) [RFC7402]. The Host Identities are used to create the needed ESP Security Associations (SAs) and to authenticate the hosts. When ESP is used, the actual payload IP packets do not differ in any way from standard ESP-protected IP packets.

Much has been learned about HIP [RFC6538] since [RFC4423] was published. This document expands Host Identities beyond their original use to enable IP connectivity and security to enable general interhost secure signaling at any protocol layer. The signal may establish a security association between the hosts or simply pass information within the channel.

2. Terminology

2.1. Terms Common to Other Documents

Term	Explanation
Public key	The public key of an asymmetric cryptographic key pair. Used as a publicly known identifier for cryptographic identity authentication. Public is a relative term here, ranging from "known to peers only" to "known to the world".
Private key	The private or secret key of an asymmetric cryptographic key pair. Assumed to be known only to the party identified by the corresponding public key. Used by the identified party to authenticate its identity to other parties.
Public key pair	An asymmetric cryptographic key pair consisting of public and private keys. For example, Rivest-Shamir-Adleman (RSA), Digital Signature Algorithm (DSA) and Elliptic Curve DSA (ECDSA) key pairs are such key pairs.
Endpoint	A communicating entity. For historical reasons, the term 'computing platform' is used in this document as a (rough) synonym for endpoint.

Table 1

2.2. Terms Specific to This and Other HIP Documents

It should be noted that many of the terms defined herein are tautologous, self-referential, or defined through circular reference to other terms. This is due to the succinct nature of the definitions. See the text elsewhere in this document and the base specification [RFC7401] for more elaborate explanations.

Term	Explanation
Computing platform	An entity capable of communicating and computing, for example, a computer. See the definition of 'Endpoint', above.
HIP base exchange	A cryptographic protocol; see also Section 6.

Term	Explanation
HIP packet	An IP packet that carries a 'Host Identity Protocol' message.
Host Identity	An abstract concept assigned to a 'computing platform'. See 'Host Identifier', below.
Host Identifier	A public key used as a name for a Host Identity.
Host Identity namespace	A name space formed by all possible Host Identifiers.
Host Identity Protocol	A protocol used to carry and authenticate Host Identifiers and other information.
Host Identity Hash	The cryptographic hash used in creating the Host Identity Tag from the Host Identifier.
Host Identity Tag	A 128-bit datum created by taking a cryptographic hash over a Host Identifier plus bits to identify which hash was used.
Local Scope Identifier	A 32-bit datum denoting a Host Identity.
Public Host Identifier and Identity	A published or publicly known Host Identifier used as a public name for a Host Identity, and the corresponding Identity.
Unpublished Host Identifier and Identity	A Host Identifier that is not placed in any public directory, and the corresponding Host Identity. Unpublished Host Identities are typically short lived in nature, being often replaced and possibly used just once.
Rendezvous Mechanism	A mechanism used to locate mobile hosts based on their HIT.

Table 2

3. Background

The Internet is built from three principal components: computing platforms (endpoints), packet transport (i.e., internetworking) infrastructure, and services (applications). The Internet exists to service two principal components: people and robotic services (silicon-based people, if you will). All these components need to be named in order to interact in a scalable manner. Here we concentrate on naming computing platforms and packet transport elements.

There are two principal namespaces in use in the Internet for these components: IP addresses, and Domain Names. Domain Names provide hierarchically assigned names for some computing platforms and some services. Each hierarchy is delegated from the level above; there is no anonymity in Domain Names. Email, HTTP, and SIP addresses all reference Domain Names.

The IP addressing namespace has been overloaded to name both interfaces (at Layer 3) and endpoints (for the endpoint-specific part of Layer 3 and for Layer 4). In their role as interface names, IP addresses are sometimes called "locators" and serve as an endpoint within a routing topology.

IP addresses are numbers that name networking interfaces, and typically only when the interface is connected to the network. Originally, IP addresses had long-term significance. Today, the vast number of interfaces use ephemeral and/or non-unique IP addresses. That is, every time an interface is connected to the network, it is assigned an IP address.

In the current Internet, the transport layers are coupled to the IP addresses. Neither can evolve separately from the other. IPng deliberations were strongly shaped by the decision that a corresponding TCPng would not be created.

There are three critical deficiencies with the current namespaces. First, the establishing of initial contact and the sustaining of data flows between two hosts can be challenging due to private address realms and the ephemeral nature of addresses. Second, confidentiality is not provided in a consistent, trustable manner. Finally, authentication for systems and datagrams is not provided. All of these deficiencies arise because computing platforms are not well named with the current namespaces.

3.1. A Desire for a Namespace for Computing Platforms

An independent namespace for computing platforms could be used in end-to-end operations independent of the evolution of the internetworking layer and across the many internetworking layers. This could support rapid readdressing of the internetworking layer because of mobility, rehoming, or renumbering.

If the namespace for computing platforms is based on public-key cryptography, it can also provide authentication services. If this namespace is locally created without requiring registration, it can provide anonymity.

Such a namespace (for computing platforms) and the names in it should have the following characteristics:

- The namespace should be applied to the IP 'kernel' or stack. The IP stack is the 'component' between applications and the packet transport infrastructure.
- The namespace should fully decouple the internetworking layer from the higher layers. The names should replace all occurrences of IP addresses within applications (like in the Transport Control Block, TCB). This replacement can be handled transparently for legacy applications as the Local Scope Identifiers (LSIs) and HITs are compatible with IPv4 and IPv6

addresses [RFC5338]. However, HIP-aware applications require some modifications from the developers, who may employ networking API extensions for HIP [RFC6317].

- The introduction of the namespace should not mandate any administrative infrastructure. Deployment must come from the bottom up, in a pairwise deployment.
- The names should have a fixed-length representation, for easy inclusion in datagram headers and existing programming interfaces (e.g., the TCB).
- Using the namespace should be affordable when used in protocols. This is primarily a packet size issue. There is also a computational concern in affordability.
- Name collisions should be avoided as much as possible. The mathematics of the birthday paradox can be used to estimate the chance of a collision in a given population and hash space. In general, for a random hash space of size n bits, we would expect to obtain a collision after approximately 1.2*sqrt(2^n) hashes were obtained. For 64 bits, this number is roughly 4 billion. A hash size of 64 bits may be too small to avoid collisions in a large population; for example, there is a 1% chance of collision in a population of 640M. For 100 bits (or more), we would not expect a collision until approximately 2^{50} (1 quadrillion) hashes were generated. With the currently used hash size of 96 bits [RFC7343], the figure is 2^{48} (281 trillions).
- The names should have a localized abstraction so that they can be used in existing protocols and APIs.
- It must be possible to create names locally. When such names are not published, this can provide anonymity at the cost of making resolvability very difficult.
- The namespace should provide authentication services.
- The names should be long-lived, but replaceable at any time. This impacts access control lists; short lifetimes will tend to result in tedious list maintenance or require a namespace infrastructure for central control of access lists.

In this document, the namespace approaching these ideas is called the Host Identity namespace. Using Host Identities requires its own protocol layer, the Host Identity Protocol, between the internetworking and transport layers. The names are based on public-key cryptography to supply authentication services. Properly designed, it can deliver all of the above-stated requirements.

4. Host Identity Namespace

A name in the Host Identity namespace, a Host Identifier (HI), represents a statistically globally unique name for naming any system with an IP stack. This identity is normally associated with, but not limited to, an IP stack. A system can have multiple identities, some 'well known', some unpublished or 'anonymous'. A system may self-assert its own identity, or may use a third-party authenticator like DNSSEC [RFC4033], Pretty Good Privacy (PGP), or X.509 to 'notarize' the identity assertion to another namespace.

In theory, any name that can claim to be 'statistically globally unique' may serve as a Host Identifier. In the HIP architecture, the public key of a private-public key pair has been chosen as the Host Identifier because it can be self-managed and it is computationally difficult to forge. As

specified in the Host Identity Protocol specification [RFC7401], a public-key-based HI can authenticate the HIP packets and protect them from man-in-the-middle (MitM) attacks. Since authenticated datagrams are mandatory to provide much of HIP's denial-of-service protection, the Diffie-Hellman exchange in HIP base exchange has to be authenticated. Thus, only public-key HI and authenticated HIP messages are supported in practice.

In this document, some non-cryptographic forms of HI and HIP are referenced, but cryptographic forms should be preferred because they are more secure than their non-cryptographic counterparts. There has been past research in challenge puzzles using non-cryptographic HI for Radio Frequency IDentification (RFID), in an HIP exchange tailored to the workings of such challenges (as described further in [urien-rfid] and [urien-rfid-draft]).

4.1. Host Identifiers

Host Identity adds two main features to Internet protocols. The first is a decoupling of the internetworking and transport layers; see Section 5. This decoupling will allow for independent evolution of the two layers. Additionally, it can provide end-to-end services over multiple internetworking realms. The second feature is host authentication. Because the Host Identifier is a public key, this key can be used for authentication in security protocols like ESP.

An identity is based on public-private key cryptography in HIP. The Host Identity is referred to by its public component, the public key. Thus, the name representing a Host Identity in the Host Identity namespace, i.e., the Host Identifier, is the public key. In a way, the possession of the private key defines the Identity itself. If the private key is possessed by more than one node, the Identity can be considered to be a distributed one.

Architecturally, any other Internet naming convention might form a usable base for Host Identifiers. However, non-cryptographic names should only be used in situations of high trust and/or low risk. That is any place where host authentication is not needed (no risk of host spoofing) and no use of ESP. However, at least for interconnected networks spanning several operational domains, the set of environments where the risk of host spoofing allowed by non-cryptographic Host Identifiers is acceptable is the null set. Hence, the current HIP documents do not specify how to use any other types of Host Identifiers but public keys. For instance, the Back to My Mac service [RFC6281] from Apple comes pretty close to the functionality of HIP, but unlike HIP, it is based on non-cryptographic identifiers.

The actual Host Identifiers are never directly used at the transport or network layers. The corresponding Host Identifiers (public keys) may be stored in various DNS or other directories as identified elsewhere in this document, and they are passed in the HIP base exchange. A Host Identity Tag (HIT) is used in other protocols to represent the Host Identity. Another representation of the Host Identities, the Local Scope Identifier (LSI), can also be used in protocols and APIs.

4.2. Host Identity Hash (HIH)

The Host Identity Hash (HIH) is the cryptographic hash algorithm used in producing the HIT from the HI. It is also the hash used throughout HIP for consistency and simplicity. It is possible for the two hosts in the HIP exchange to use different hash algorithms.

Multiple HIHs within HIP are needed to address the moving target of creation and eventual compromise of cryptographic hashes. This significantly complicates HIP and offers an attacker an additional downgrade attack that is mitigated in HIP [RFC7401].

4.3. Host Identity Tag (HIT)

A Host Identity Tag (HIT) is a 128-bit representation for a Host Identity. Due to its size, it is suitable for use in the existing sockets API in the place of IPv6 addresses (e.g., in sockaddr_in6 structure, sin6_addr member) without modifying applications. It is created from an HIH, an IPv6 prefix [RFC7343], and a hash identifier. There are two advantages of using the HIT over using the Host Identifier in protocols. First, its fixed length makes for easier protocol coding and also better manages the packet size cost of this technology. Second, it presents the identity in a consistent format to the protocol independent of the cryptographic algorithms used.

In essence, the HIT is a hash over the public key. As such, two algorithms affect the generation of a HIT: the public-key algorithm of the HI and the used HIH. The two algorithms are encoded in the bit presentation of the HIT. As the two communicating parties may support different algorithms, [RFC7401] defines the minimum set for interoperability. For further interoperability, the Responder may store its keys in DNS records, and thus the Initiator may have to couple destination HITs with appropriate source HITs according to matching HIH.

In the HIP packets, the HITs identify the sender and recipient of a packet. Consequently, a HIT should be unique in the whole IP universe as long as it is being used. In the extremely rare case of a single HIT mapping to more than one Host Identity, the Host Identifiers (public keys) will make the final difference. If there is more than one public key for a given node, the HIT acts as a hint for the correct public key to use.

Although it may be rare for an accidental collision to cause a single HIT mapping to more than one Host Identity, it may be the case that an attacker succeeds to find, by brute force or algorithmic weakness, a second Host Identity hashing to the same HIT. This type of attack is known as a preimage attack, and the resistance to finding a second Host Identifier (public key) that hashes to the same HIT is called second preimage resistance. Second preimage resistance in HIP is based on the hash algorithm strength and the length of the hash output used. Through HIPv2 [RFC7401], this resistance is 96 bits (less than the 128-bit width of an IPv6 address field due to the presence of the Overlay Routable Cryptographic Hash Identifiers (ORCHID) prefix [RFC7343]). 96 bits of resistance was considered acceptable strength during the design of HIP but may eventually be considered insufficient for the threat model of an envisioned deployment. One possible mitigation would be to augment the use of HITs in the deployment with the HIs themselves (and mechanisms to securely bind the HIs to the HITs), so that the HI becomes the final authority. It also may be possible to increase the difficulty of a brute force attack by making

the generation of the HI more computationally difficult, such as the hash extension approach of Secure Neighbor Discovery Cryptographically Generated Addresses (CGAs) [RFC3972], although the HIP specifications through HIPv2 do not provide such a mechanism. Finally, deployments that do not use ORCHIDs (such as certain types of overlay networks) might also use the full 128-bit width of an IPv6 address field for the HIT.

4.4. Local Scope Identifier (LSI)

An LSI is a 32-bit localized representation for a Host Identity. Due to its size, it is suitable for use in the existing sockets API in the place of IPv4 addresses (e.g., in sockaddr_in structure, sin_addr member) without modifying applications. The purpose of an LSI is to facilitate using Host Identities in existing APIs for IPv4-based applications. LSIs are never transmitted on the wire; when an application sends data using a pair of LSIs, the HIP layer (or sockets handler) translates the LSIs to the corresponding HITs, and vice versa for the receiving of data. Besides facilitating HIP-based connectivity for legacy IPv4 applications, the LSIs are beneficial in two other scenarios [RFC6538].

In the first scenario, two IPv4-only applications reside on two separate hosts connected by IPv6-only network. With HIP-based connectivity, the two applications are able to communicate despite the mismatch in the protocol families of the applications and the underlying network. The reason is that the HIP layer translates the LSIs originating from the upper layers into routable IPv6 locators before delivering the packets on the wire.

The second scenario is the same as the first one, but with the difference that one of the applications supports only IPv6. Now two obstacles hinder the communication between the applications: the addressing families of the two applications differ, and the application residing at the IPv4-only side is again unable to communicate because of the mismatch between addressing families of the application (IPv4) and network (IPv6). With HIP-based connectivity for applications, this scenario works; the HIP layer can choose whether to translate the locator of an incoming packet into an LSI or HIT.

Effectively, LSIs improve IPv6 interoperability at the network layer as described in the first scenario and at the application layer as depicted in the second example. The interoperability mechanism should not be used to avoid transition to IPv6; the authors firmly believe in IPv6 adoption and encourage developers to port existing IPv4-only applications to use IPv6. However, some proprietary, closed-source, IPv4-only applications may never see the daylight of IPv6, and the LSI mechanism is suitable for extending the lifetime of such applications even in IPv6-only networks.

The main disadvantage of an LSI is its local scope. Applications may violate layering principles and pass LSIs to each other in application-layer protocols. As the LSIs are valid only in the context of the local host, they may represent an entirely different host when passed to another host. However, it should be emphasized here that the LSI concept is effectively a host-based NAT and does not introduce any more issues than the prevalent middlebox-based NATs for IPv4. In other words, the applications violating layering principles are already broken by the NAT boxes that are ubiquitously deployed.

4.5. Storing Host Identifiers in Directories

The public Host Identifiers should be stored in DNS; the unpublished Host Identifiers should not be stored anywhere (besides the communicating hosts themselves). The (public) HI along with the supported HIHs are stored in a new Resource Record (RR) type. This RR type is defined in the HIP DNS extension [RFC8005].

Alternatively, or in addition to storing Host Identifiers in the DNS, they may be stored in various other directories. For instance, a directory based on the Lightweight Directory Access Protocol (LDAP) or a Public Key Infrastructure (PKI) [RFC8002] may be used. Alternatively, Distributed Hash Tables (DHTs) [RFC6537] have successfully been utilized [RFC6538]. Such a practice may allow them to be used for purposes other than pure host identification.

Some types of applications may cache and use Host Identifiers directly, while others may indirectly discover them through a symbolic host name (such as a Fully Qualified Domain Name (FQDN)) look up from a directory. Even though Host Identities can have a substantially longer lifetime associated with them than routable IP addresses, directories may be a better approach to manage the lifespan of Host Identities. For example, an LDAP-based directory or DHT can be used for locally published identities whereas DNS can be more suitable for public advertisement.

5. New Stack Architecture

One way to characterize Host Identity is to compare the proposed HI-based architecture with the current one. Using the terminology from the IRTF Name Space Research Group Report [nsrg-report] and, e.g., the document on "Endpoints and Endpoint Names" [chiappa-endpoints], the IP addresses currently embody the dual role of locators and endpoint identifiers. That is, each IP address names a topological location in the Internet, thereby acting as a routing direction vector, or locator. At the same time, the IP address names the physical network interface currently located at the point-of-attachment, thereby acting as an endpoint name.

In the HIP architecture, the endpoint names and locators are separated from each other. IP addresses continue to act as locators. The Host Identifiers take the role of endpoint identifiers. It is important to understand that the endpoint names based on Host Identities are slightly different from interface names; a Host Identity can be simultaneously reachable through several interfaces.

The difference between the bindings of the logical entities are illustrated in Figure 1. The left side illustrates the current TCP/IP architecture and the right side the HIP-based architecture.

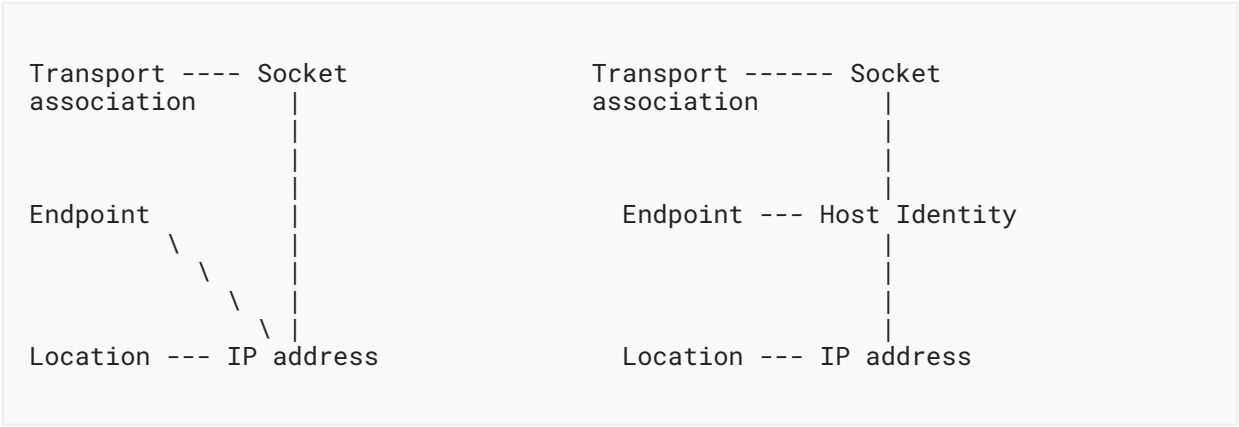

Figure 1

Architecturally, HIP provides for a different binding of transport-layer protocols. That is, the transport-layer associations, i.e., TCP connections and UDP associations, are no longer bound to IP addresses but rather to Host Identities. In practice, the Host Identities are exposed as LSIs and HITs for legacy applications and the transport layer to facilitate backward compatibility with existing networking APIs and stacks.

The HIP layer is logically located at Layer 3.5, between the transport and network layers, in the networking stack. It acts as shim layer for transport data utilizing LSIs or HITs but leaves other data intact. The HIP layer translates between the two forms of HIP identifiers originating from the transport layer into routable IPv4/IPv6 addresses for the network layer and vice versa for the reverse direction.

5.1. On the Multiplicity of Identities

A host may have multiple identities both at the client and server side. This raises some additional concerns that are addressed in this section.

For security reasons, it may be a bad idea to duplicate the same Host Identity on multiple hosts because the compromise of a single host taints the identities of the other hosts. Management of machines with identical Host Identities may also present other challenges and, therefore, it is advisable to have a unique identity for each host.

At the server side, utilizing DNS is a better alternative than a shared Host Identity to implement load balancing. A single FQDN entry can be configured to refer to multiple Host Identities. Each of the FQDN entries can be associated with the related locators or with a single shared locator in the case the servers are using the same HIP rendezvous server (Section 6.3) or HIP relay server (Section 6.4).

Instead of duplicating identities, HIP opportunistic mode can be employed, where the Initiator leaves out the identifier of the Responder when initiating the key exchange and learns it upon the completion of the exchange. The trade-offs are related to lowered security guarantees, but a benefit of the approach is to avoid the publishing of Host Identifiers in any directories [komu-

leap]. Since many public servers already employ DNS as their directory, opportunistic mode may be more suitable for, e.g., peer-to-peer connectivity. It is also worth noting that opportunistic mode is also required in practice when anycast IP addresses would be utilized as locators.

HIP opportunistic mode could be utilized in association with HIP rendezvous servers or HIP relay servers [komu-diss]. In such a scenario, the Initiator sends an I1 message with a wildcard destination HIT to the locator of a HIP rendezvous/relay server. When the receiving rendezvous/relay server is serving multiple registered Responders, the server can choose the ultimate destination HIT, thus acting as a HIP-based load balancer. However, this approach is still experimental and requires further investigation.

At the client side, a host may have multiple Host Identities, for instance, for privacy purposes. Another reason can be that the person utilizing the host employs different identities for different administrative domains as an extra security measure. If a HIP-aware middlebox, such as a HIP-based firewall, is on the path between the client and server, the user or the underlying system should carefully choose the correct identity to avoid the firewall unnecessarily dropping HIP-based connectivity [komu-diss].

Similarly, a server may have multiple Host Identities. For instance, a single web server may serve multiple different administrative domains. Typically, the distinction is accomplished based on the DNS name, but also the Host Identity could be used for this purpose. However, a more compelling reason to employ multiple identities is the HIP-aware firewall that is unable to see the HTTP traffic inside the encrypted IPsec tunnel. In such a case, each service could be configured with a separate identity, thus allowing the firewall to segregate the different services of the single web server from each other [lindqvist-enterprise].

6. Control Plane

HIP decouples the control and data planes from each other. Two end-hosts initialize the control plane using a key exchange procedure called the base exchange. The procedure can be assisted by HIP-specific infrastructural intermediaries called rendezvous or relay servers. In the event of IP address changes, the end-hosts sustain control plane connectivity with mobility and multihoming extensions. Eventually, the end-hosts terminate the control plane and remove the associated state.

6.1. Base Exchange

The base exchange is a key exchange procedure that authenticates the Initiator and Responder to each other using their public keys. Typically, the Initiator is the client-side host and the Responder is the server-side host. The roles are used by the state machine of a HIP implementation but then discarded upon successful completion.

The exchange consists of four messages during which the hosts also create symmetric keys to protect the control plane with Hash-based Message Authentication Codes (HMACs). The keys can be also used to protect the data plane, and IPsec ESP [RFC7402] is typically used as the data plane protocol, albeit HIP can also accommodate others. Both the control and data planes are terminated using a closing procedure consisting of two messages.

In addition, the base exchange also includes a computational puzzle [RFC7401] that the Initiator must solve. The Responder chooses the difficulty of the puzzle, which permits the Responder to delay new incoming Initiators according to local policies, for instance, when the Responder is under heavy load. The puzzle can offer some resiliency against DoS attacks because the design of the puzzle mechanism allows the Responder to remain stateless until the very end of the base exchange [aura-dos]. HIP puzzles have also been studied under steady-state DDoS attacks [beal-dos], on multiple adversary models with varying puzzle difficulties [tritilanunt-dos], and with ephemeral Host Identities [komu-mitigation].

6.2. End-Host Mobility and Multihoming

HIP decouples the transport from the internetworking layer and binds the transport associations to the Host Identities (actually through either the HIT or LSI). After the initial key exchange, the HIP layer maintains transport-layer connectivity and data flows using its extensions for mobility [RFC8046] and multihoming [RFC8047]. Consequently, HIP can provide for a degree of internetworking mobility and multihoming at a low infrastructure cost. HIP mobility includes IP address changes (via any method) to either party. Thus, a system is considered mobile if its IP address can change dynamically for any reason like PPP, DHCP, IPv6 prefix reassignments, or a NAT device remapping its translation. Likewise, a system is considered multihomed if it has more than one globally routable IP address at the same time. HIP links IP addresses together when multiple IP addresses correspond to the same Host Identity. If one address becomes unusable, or a more preferred address becomes available, existing transport associations can easily be moved to another address.

When a mobile node moves while communication is ongoing, address changes are rather straightforward. The mobile node sends a HIP UPDATE packet to inform the peer of the new address(es), and the peer then verifies that the mobile node is reachable through these addresses. This way, the peer can avoid flooding attacks as further discussed in Section 11.2.

6.3. Rendezvous Mechanism

Establishing a contact to a mobile, moving node is slightly more involved. In order to start the HIP exchange, the Initiator node has to know how to reach the mobile node. For instance, the mobile node can employ Dynamic DNS [RFC2136] to update its reachability information in the DNS. To avoid the dependency to DNS, HIP provides its own HIP-specific alternative: the HIP rendezvous mechanism as defined in the HIP rendezvous specification [RFC8004].

Using the HIP rendezvous extensions, the mobile node keeps the rendezvous infrastructure continuously updated with its current IP address(es). The mobile nodes trusts the rendezvous mechanism in order to properly maintain their HIT and IP address mappings.

The rendezvous mechanism is especially useful in scenarios where both of the nodes are expected to change their address at the same time. In such a case, the HIP UPDATE packets will cross each other in the network and never reach the peer node.

6.4. Relay Mechanism

The HIP relay mechanism [RFC9028] is an alternative to the HIP rendezvous mechanism. The HIP relay mechanism is more suitable for IPv4 networks with NATs because a HIP relay can forward all control and data plane communications in order to guarantee successful NAT traversal.

6.5. Termination of the Control Plane

The control plane between two hosts is terminated using a secure two-message exchange as specified in base exchange specification [RFC7401]. The related state (i.e., host associations) should be removed upon successful termination.

7. Data Plane

The encapsulation format for the data plane used for carrying the application-layer traffic can be dynamically negotiated during the key exchange. For instance, HICCUPS extensions [RFC6078] define one way to transport application-layer datagrams directly over the HIP control plane, protected by asymmetric key cryptography. Also, Secure Real-time Transport Protocol (SRTP) has been considered as the data encapsulation protocol [hip-srtp]. However, the most widely implemented method is the Encapsulated Security Payload (ESP) [RFC7402] that is protected by symmetric keys derived during the key exchange. ESP Security Associations (SAs) offer both confidentiality and integrity protection, of which the former can be disabled during the key exchange. In the future, other ways of transporting application-layer data may be defined.

The ESP SAs are established and terminated between the Initiator and the Responder hosts. Usually, the hosts create at least two SAs, one in each direction (Initiator-to-Responder SA and Responder-to-Initiator SA). If the IP addresses of either host changes, the HIP mobility extensions can be used to renegotiate the corresponding SAs.

On the wire, the difference in the use of identifiers between the HIP control and data planes is that the HITs are included in all control packets, but not in the data plane when ESP is employed. Instead, the ESP employs Security Parameter Index (SPI) numbers that act as compressed HITs. Any HIP-aware middlebox (for instance, a HIP-aware firewall) interested in the ESP-based data plane should keep track between the control and data plane identifiers in order to associate them with each other.

Since HIP does not negotiate any SA lifetimes, all lifetimes are subject to local policy. The only lifetimes a HIP implementation must support are sequence number rollover (for replay protection) and SA timeout. An SA times out if no packets are received using that SA. Implementations may support lifetimes for the various ESP transforms and other data plane protocols.

8. HIP and NATs

Passing packets between different IP addressing realms requires changing IP addresses in the packet header. This may occur, for example, when a packet is passed between the public Internet and a private address space, or between IPv4 and IPv6 networks. The address translation is usually implemented as Network Address Translation (NAT) [RFC3022] or the historic NAT Protocol Translation (NAT-PT) [RFC2766].

In a network environment where identification is based on the IP addresses, identifying the communicating nodes is difficult when NATs are employed because private address spaces are overlapping. In other words, two hosts cannot be distinguished from each other solely based on their IP addresses. With HIP, the transport-layer endpoints (i.e., applications) are bound to unique Host Identities rather than overlapping private addresses. This allows two endpoints to distinguish one other even when they are located in different private address realms. Thus, the IP addresses are used only for routing purposes and can be changed freely by NATs when a packet between two HIP-capable hosts traverses through multiple private address realms.

NAT traversal extensions for HIP [RFC9028] can be used to realize the actual end-to-end connectivity through NAT devices. To support basic backward compatibility with legacy NATs, the extensions encapsulate both HIP control and data planes in UDP. The extensions define mechanisms for forwarding the two planes through an intermediary host called HIP relay and procedures to establish direct end-to-end connectivity by penetrating NATs. Besides this "native" NAT traversal mode for HIP, other NAT traversal mechanisms have been successfully utilized, such as Teredo [RFC4380] (as described in further detail in [varjonen-split]).

Besides legacy NATs, a HIP-aware NAT has been designed and implemented [ylitalo-spinat]. For a HIP-based flow, a HIP-aware NAT or HIP-aware historic NAT-PT system tracks the mapping of HITs, and the corresponding ESP SPIs, to an IP address. The NAT system has to learn mappings both from HITs and from SPIs to IP addresses. Many HITs (and SPIs) can map to a single IP address on a NAT, simplifying connections on address-poor NAT interfaces. The NAT can gain much of its knowledge from the HIP packets themselves; however, some NAT configuration may be necessary.

8.1. HIP and Upper-Layer Checksums

There is no way for a host to know if any of the IP addresses in an IP header are the addresses used to calculate the TCP checksum. That is, it is not feasible to calculate the TCP checksum using the actual IP addresses in the pseudo header; the addresses received in the incoming packet are not necessarily the same as they were on the sending host. Furthermore, it is not possible to recompute the upper-layer checksums in the NAT/NAT-PT system, since the traffic is ESP protected. Consequently, the TCP and UDP checksums are calculated using the HITs in the place of the IP addresses in the pseudo header. Furthermore, only the IPv6 pseudo header format is used. This provides for IPv4 / IPv6 protocol translation.

9. Multicast

A number of studies investigating HIP-based multicast have been published (including [shields-hip], [zhu-hip], [amir-hip], [kovacshazi-host], and [zhu-secure]). In particular, so-called Bloom filters, which allow the compression of multiple labels into small data structures, may be a promising way forward [sarela-bloom]. However, the different schemes have not been adopted by the HIP working group (nor the HIP research group in the IRTF), so the details are not further elaborated here.

10. HIP Policies

There are a number of variables that influence the HIP exchange that each host must support. All HIP implementations should support at least two HIs, one to publish in DNS or a similar directory service and an unpublished one for anonymous usage (that should expect to be rotated frequently in order to disrupt linkability and/or trackability). Although unpublished HIs will rarely be used as Responder HIs, they are likely to be common for Initiators. As stated in [RFC7401], "all HIP implementations **MUST** support more than one simultaneous HI, at least one of which **SHOULD** be reserved for anonymous usage", and "support for more than two HIs is **RECOMMENDED**". This provides new challenges for systems or users to decide which type of HI to expose when they start a new session.

Opportunistic mode (where the Initiator starts a HIP exchange without prior knowledge of the Responder's HI) presents a security trade-off. At the expense of being subject to MitM attacks, the opportunistic mode allows the Initiator to learn the identity of the Responder during communication rather than from an external directory. Opportunistic mode can be used for registration to HIP-based services [RFC8003] (i.e., utilized by HIP for its own internal purposes) or by the application layer [komu-leap]. For security reasons, especially the latter requires some involvement from the user to accept the identity of the Responder similar to how the Secure Shell (SSH) protocol prompts the user when connecting to a server for the first time [pham-leap]. In practice, this can be realized in end-host-based firewalls in the case of legacy applications [karvonen-usable] or with native APIs for HIP APIs [RFC6317] in the case of HIP-aware applications.

As stated in [RFC7401]:

> Initiators **MAY** use a different HI for different Responders to provide basic privacy. Whether such private HIs are used repeatedly with the same Responder, and how long these HIs are used, are decided by local policy and depend on the privacy requirements of the Initiator.

According to [RFC7401]:

> Responders that only respond to selected Initiators require an Access Control List (ACL), representing for which hosts they accept HIP base exchanges, and the preferred transport format and local lifetimes. Wildcarding **SHOULD** be supported for such ACLs, and also for Responders that offer public or anonymous services.

11. Security Considerations

This section includes discussion on some issues and solutions related to security in the HIP architecture.

11.1. MitM Attacks

HIP takes advantage of the Host Identity paradigm to provide secure authentication of hosts and to provide a fast key exchange for ESP. HIP also attempts to limit the exposure of the host to various denial-of-service (DoS) and man-in-the-middle (MitM) attacks. In so doing, HIP itself is subject to its own DoS and MitM attacks that potentially could be more damaging to a host's ability to conduct business as usual.

Resource exhausting DoS attacks take advantage of the cost of setting up a state for a protocol on the Responder compared to the 'cheapness' on the Initiator. HIP allows a Responder to increase the cost of the start of state on the Initiator and makes an effort to reduce the cost to the Responder. This is done by having the Responder start the authenticated Diffie-Hellman exchange instead of the Initiator, making the HIP base exchange four packets long. The first packet sent by the Responder can be prebuilt to further mitigate the costs. This packet also includes a computational puzzle that can optionally be used to further delay the Initiator, for instance, when the Responder is overloaded. The details are explained in the base exchange specification [RFC7401].

MitM attacks are difficult to defend against without third-party authentication. A skillful MitM could easily handle all parts of the HIP base exchange, but HIP indirectly provides the following protection from a MitM attack. If the Responder's HI is retrieved from a signed DNS zone or securely obtained by some other means, the Initiator can use this to authenticate the signed HIP packets. Likewise, if the Initiator's HI is in a secure DNS zone, the Responder can retrieve it and validate the signed HIP packets. However, since an Initiator may choose to use an unpublished HI, it knowingly risks a MitM attack. The Responder may choose not to accept a HIP exchange with an Initiator using an unknown HI.

Other types of MitM attacks against HIP can be mounted using ICMP messages that can be used to signal about problems. As an overall guideline, the ICMP messages should be considered as unreliable "hints" and should be acted upon only after timeouts. The exact attack scenarios and countermeasures are described in full detail in the base exchange specification [RFC7401].

A MitM attacker could try to replay older I1 or R1 messages using weaker cryptographic algorithms as described in Section 4.1.4 of [RFC7401]. The base exchange has been augmented to deal with such an attack by restarting on the detection of the attack. At worst, this would only

lead to a situation in which the base exchange would never finish (or would be aborted after some retries). As a drawback, this leads to a six-way base exchange, which may seem bad at first. However, since this only occurs in an attack scenario and since the attack can be handled (so it is not interesting to mount anymore), we assume the subsequent messages do not represent a security threat. Since the MitM cannot be successful with a downgrade attack, these sorts of attacks will only occur as 'nuisance' attacks. So, the base exchange would still be usually just four packets even though implementations must be prepared to protect themselves against the downgrade attack.

In HIP, the Security Association for ESP is indexed by the SPI; the source address is always ignored, and the destination address may be ignored as well. Therefore, HIP-enabled ESP is IP address independent. This might seem to make attacking easier, but ESP with replay protection is already as well protected as possible, and the removal of the IP address as a check should not increase the exposure of ESP to DoS attacks.

11.2. Protection against Flooding Attacks

Although the idea of informing about address changes by simply sending packets with a new source address appears appealing, it is not secure enough. That is, even if HIP does not rely on the source address for anything (once the base exchange has been completed), it appears to be necessary to check a mobile node's reachability at the new address before actually sending any larger amounts of traffic to the new address.

Blindly accepting new addresses would potentially lead to flooding DoS attacks against third parties [RFC4225]. In a distributed flooding attack, an attacker opens high-volume HIP connections with a large number of hosts (using unpublished HIs) and then claims to all of these hosts that it has moved to a target node's IP address. If the peer hosts were to simply accept the move, the result would be a packet flood to the target node's address. To prevent this type of attack, HIP mobility extensions include a return routability check procedure where the reachability of a node is separately checked at each address before using the address for larger amounts of traffic.

A credit-based authorization approach for "Host Mobility with the Host Identity Protocol" [RFC8046] can be used between hosts for sending data prior to completing the address tests. Otherwise, if HIP is used between two hosts that fully trust each other, the hosts may optionally decide to skip the address tests. However, such performance optimization must be restricted to peers that are known to be trustworthy and capable of protecting themselves from malicious software.

11.3. HITs Used in ACLs

At end-hosts, HITs can be used in IP-based access control lists at the application and network layers. At middleboxes, HIP-aware firewalls [lindqvist-enterprise] can use HITs or public keys to control both ingress and egress access to networks or individual hosts, even in the presence of mobile devices because the HITs and public keys are topology independent. As discussed earlier in Section 7, once a HIP session has been established, the SPI value in an ESP packet may be used as an index, indicating the HITs. In practice, firewalls can inspect HIP packets to learn of the

bindings between HITs, SPI values, and IP addresses. They can even explicitly control ESP usage, dynamically opening ESP only for specific SPI values and IP addresses. The signatures in HIP packets allow a capable firewall to ensure that the HIP exchange is indeed occurring between two known hosts. This may increase firewall security.

A potential drawback of HITs in ACLs is their 'flatness', which means they cannot be aggregated, and this could potentially result in larger table searches in HIP-aware firewalls. A way to optimize this could be to utilize Bloom filters for grouping HITs [sarela-bloom]. However, it should be noted that it is also easier to exclude individual, misbehaving hosts when the firewall rules concern individual HITs rather than groups.

There has been considerable bad experience with distributed ACLs that contain material related to public keys, for example, with SSH. If the owner of a key needs to revoke it for any reason, the task of finding all locations where the key is held in an ACL may be impossible. If the reason for the revocation is due to private key theft, this could be a serious issue.

A host can keep track of all of its partners that might use its HIT in an ACL by logging all remote HITs. It should only be necessary to log Responder hosts. With this information, the host can notify the various hosts about the change to the HIT. There have been attempts to develop a secure method to issue the HIT revocation notice [zhang-revocation].

Some of the HIP-aware middleboxes, such as firewalls [lindqvist-enterprise] or NATs [ylitalo-spinat], may observe the on-path traffic passively. Such middleboxes are transparent by their nature and may not get a notification when a host moves to a different network. Thus, such middleboxes should maintain soft state and time out when the control and data planes between two HIP end-hosts have been idle too long. Correspondingly, the two end-hosts may send periodically keepalives, such as UPDATE packets or ICMP messages inside the ESP tunnel, to sustain state at the on-path middleboxes.

One general limitation related to end-to-end encryption is that middleboxes may not be able to participate in the protection of data flows. While the issue may also affect other protocols, Heer et al. [heer-end-host] have analyzed the problem in the context of HIP. More specifically, when ESP is used as the data plane protocol for HIP, the association between the control and data planes is weak and can be exploited under certain assumptions. In the scenario, the attacker has already gained access to the target network protected by a HIP-aware firewall, but wants to circumvent the HIP-based firewall. To achieve this, the attacker passively observes a base exchange between two HIP hosts and later replays it. This way, the attacker manages to penetrate the firewall and can use a fake ESP tunnel to transport its own data. This is possible because the firewall cannot distinguish when the ESP tunnel is valid. As a solution, HIP-aware middleboxes may participate in the control plane interaction by adding random nonce parameters to the control traffic, which the end-hosts have to sign to guarantee the freshness of the control traffic [heer-midauth]. As an alternative, extensions for transporting the data plane directly over the control plane can be used [RFC6078].

11.4. Alternative HI Considerations

The definition of the Host Identifier states that the HI need not be a public key. It implies that the HI could be any value, for example, a FQDN. This document does not describe how to support such a non-cryptographic HI, but examples of such protocol variants do exist ([urien-rfid], [urien-rfid-draft]). A non-cryptographic HI would still offer the services of the HIT or LSI for NAT traversal. It would be possible to carry HITs in HIP packets that had neither privacy nor authentication. Such schemes may be employed for resource-constrained devices, such as small sensors operating on battery power, but are not further analyzed here.

If it is desirable to use HIP in a low-security situation where public key computations are considered expensive, HIP can be used with very short Diffie-Hellman and Host Identity keys. Such use makes the participating hosts vulnerable to MitM and connection hijacking attacks. However, it does not cause flooding dangers, since the address check mechanism relies on the routing system and not on cryptographic strength.

11.5. Trust on First Use

[RFC7435] highlights four design principles for Leap of Faith, or Trust On First Use (TOFU), protocols that apply also to opportunistic HIP:

1. Coexist with explicit policy
2. Prioritize communication
3. Maximize security peer by peer
4. No misrepresentation of security

According to the first TOFU design principle, "Opportunistic security never displaces or preempts explicit policy". Some application data may be too sensitive, so the related policy could require authentication (i.e., the public key or certificate) in such a case instead of the unauthenticated opportunistic mode. In practice, this has been realized in HIP implementations as follows [RFC6538].

The OpenHIP implementation allowed an Initiator to use opportunistic mode only with an explicitly configured Responder IP address, when the Responder's HIT is unknown. At the Responder, OpenHIP had an option to allow opportunistic mode with any Initiator -- trust any Initiator.

HIP for Linux (HIPL) developers experimented with more fine-grained policies operating at the application level. The HIPL implementation utilized so-called "LD_PRELOAD" hooking at the application layer that allowed a dynamically linked library to intercept socket-related calls without rebuilding the related application binaries. The library acted as a shim layer between the application and transport layers. The shim layer translated the non-HIP-based socket calls from the application into HIP-based socket calls. While the shim library involved some level of complexity as described in more detail in [komu-leap], it achieved the goal of applying opportunistic mode at the granularity of individual applications.

The second TOFU principle essentially states that communication should prioritized over security. So opportunistic mode should be, in general, allowed even if no authentication is present, and even possibly a fallback to unencrypted communications could be allowed (if policy permits) instead of blocking communications. In practice, this can be realized in three steps. In the first step, a HIP Initiator can look up the HI of a Responder from a directory such as DNS. When the Initiator discovers a HI, it can use the HI for authentication and skip the rest of the following steps. In the second step, the Initiator can, upon failing to find a HI, try opportunistic mode with the Responder. In the third step, the Initiator can fall back to non-HIP-based communications upon failing with opportunistic mode if the policy allows it. This three-step model has been implemented successfully and described in more detail in [komu-leap].

The third TOFU principle suggests that security should be maximized, so that at least opportunistic security would be employed. The three-step model described earlier prefers authentication when it is available, e.g., via DNS records (and possibly even via DNSSEC when available) and falls back to opportunistic mode when no out-of-band credentials are available. As the last resort, fallback to non-HIP-based communications can be used if the policy allows it. Also, since perfect forward secrecy (PFS) is explicitly mentioned in the third design principle, it is worth mentioning that HIP supports it.

The fourth TOFU principle states that users and noninteractive applications should be properly informed about the level of security being applied. In practice, non-HIP-aware applications would assume that no extra security is being applied, so misleading at least a noninteractive application should not be possible. In the case of interactive desktop applications, system-level prompts have been utilized in earlier HIP experiments [karvonen-usable] [RFC6538] to guide the user about the underlying HIP-based security. In general, users in those experiments perceived when HIP-based security was being used versus not used. However, the users failed to notice the difference between opportunistic, non-authenticated HIP and non-opportunistic, authenticated HIP. The reason for this was that the opportunistic HIP (i.e., lowered level of security) was not clearly indicated in the prompt. This provided a valuable lesson to further improve the user interface.

In the case of HIP-aware applications, native sockets APIs for HIP as specified in [RFC6317] can be used to develop application-specific logic instead of using generic system-level prompting. In such a case, the application itself can directly prompt the user or otherwise manage the situation in other ways. In this case, noninteractive applications also can properly log the level of security being employed because the developer can now explicitly program the use of authenticated HIP, opportunistic HIP, and plain-text communication.

It is worth mentioning a few additional items discussed in [RFC7435]. Related to active attacks, HIP has built-in protection against ciphersuite downgrade attacks as described in detail in [RFC7401]. In addition, pre-deployed certificates could be used to mitigate against active attacks in the case of opportunistic mode as mentioned in [RFC6538].

Detection of peer capabilities is also mentioned in the TOFU context. As discussed in this section, the three-step model can be used to detect peer capabilities. A host can achieve the first step of authentication, i.e., discovery of a public key, via DNS, for instance. If the host finds no keys, the host can then try opportunistic mode as the second step. Upon a timeout, the host can then

proceed to the third step by falling back to non-HIP-based communications if the policy permits. This last step is based on an implicit timeout rather an explicit (negative) acknowledgment like in the case of DNS, so the user may conclude prematurely that the connectivity has failed. To speed up the detection phase by explicitly detecting if the peer supports opportunistic HIP, researchers have proposed TCP-specific extensions [RFC6538] [komu-leap]. In a nutshell, an Initiator sends simultaneously both an opportunistic I1 packet and the related TCP SYN datagram equipped with a special TCP option to a peer. If the peer supports HIP, it drops the SYN packet and responds with an R1. If the peer is HIP incapable, it drops the HIP packet (and the unknown TCP option) and responds with a TCP SYN-ACK. The benefit of the proposed scheme is a faster, one round-trip fallback to non-HIP-based communications. The drawback is that the approach is tied to TCP (IP options were also considered, but do not work well with firewalls and NATs). Naturally, the approach does not work against an active attacker, but opportunistic mode is not supposed to protect against such an adversary anyway.

It is worth noting that while the use of opportunistic mode has some benefits related to incremental deployment, it does not achieve all the benefits of authenticated HIP [komu-diss]. Namely, authenticated HIP supports persistent identifiers in the sense that hosts are identified with the same HI independent of their movement. Opportunistic HIP meets this goal only partially: after the first contact between two hosts, HIP can successfully sustain connectivity with its mobility management extensions, but problems emerge when the hosts close the HIP association and try to reestablish connectivity. As hosts can change their location, it is no longer guaranteed that the same IP address belongs to the same host. The same address can be temporally assigned to different hosts, e.g., due to the reuse of IP addresses (e.g., by a DHCP service), the overlapping of private address realms (see also the discussion on Internet transparency in Appendix A.1), or due to an attempted attack.

12. IANA Considerations

This document has no IANA actions.

13. Changes from RFC 4423

In a nutshell, the changes from RFC 4423 [RFC4423] are mostly editorial, including clarifications on topics described in a difficult way and omitting some of the non-architectural (implementation) details that are already described in other documents. A number of missing references to the literature were also added. New topics include the drawbacks of HIP, a discussion on 802.15.4 and MAC security, HIP for IoT scenarios, deployment considerations, and a description of the base exchange.

14. References

14.1. Normative References

[RFC5482] Eggert, L. and F. Gont, "TCP User Timeout Option", RFC 5482, DOI 10.17487/RFC5482, March 2009, <https://www.rfc-editor.org/info/rfc5482>.

[RFC6079] Camarillo, G., Nikander, P., Hautakorpi, J., Keranen, A., and A. Johnston, "HIP BONE: Host Identity Protocol (HIP) Based Overlay Networking Environment (BONE)", RFC 6079, DOI 10.17487/RFC6079, January 2011, <https://www.rfc-editor.org/info/rfc6079>.

[RFC7086] Keranen, A., Camarillo, G., and J. Maenpaa, "Host Identity Protocol-Based Overlay Networking Environment (HIP BONE) Instance Specification for REsource LOcation And Discovery (RELOAD)", RFC 7086, DOI 10.17487/RFC7086, January 2014, <https://www.rfc-editor.org/info/rfc7086>.

[RFC7343] Laganier, J. and F. Dupont, "An IPv6 Prefix for Overlay Routable Cryptographic Hash Identifiers Version 2 (ORCHIDv2)", RFC 7343, DOI 10.17487/RFC7343, September 2014, <https://www.rfc-editor.org/info/rfc7343>.

[RFC7401] Moskowitz, R., Ed., Heer, T., Jokela, P., and T. Henderson, "Host Identity Protocol Version 2 (HIPv2)", RFC 7401, DOI 10.17487/RFC7401, April 2015, <https://www.rfc-editor.org/info/rfc7401>.

[RFC7402] Jokela, P., Moskowitz, R., and J. Melen, "Using the Encapsulating Security Payload (ESP) Transport Format with the Host Identity Protocol (HIP)", RFC 7402, DOI 10.17487/RFC7402, April 2015, <https://www.rfc-editor.org/info/rfc7402>.

[RFC8002] Heer, T. and S. Varjonen, "Host Identity Protocol Certificates", RFC 8002, DOI 10.17487/RFC8002, October 2016, <https://www.rfc-editor.org/info/rfc8002>.

[RFC8003] Laganier, J. and L. Eggert, "Host Identity Protocol (HIP) Registration Extension", RFC 8003, DOI 10.17487/RFC8003, October 2016, <https://www.rfc-editor.org/info/rfc8003>.

[RFC8004] Laganier, J. and L. Eggert, "Host Identity Protocol (HIP) Rendezvous Extension", RFC 8004, DOI 10.17487/RFC8004, October 2016, <https://www.rfc-editor.org/info/rfc8004>.

[RFC8005] Laganier, J., "Host Identity Protocol (HIP) Domain Name System (DNS) Extension", RFC 8005, DOI 10.17487/RFC8005, October 2016, <https://www.rfc-editor.org/info/rfc8005>.

[RFC8046] Henderson, T., Ed., Vogt, C., and J. Arkko, "Host Mobility with the Host Identity Protocol", RFC 8046, DOI 10.17487/RFC8046, February 2017, <https://www.rfc-editor.org/info/rfc8046>.

[RFC8047] Henderson, T., Ed., Vogt, C., and J. Arkko, "Host Multihoming with the Host Identity Protocol", RFC 8047, DOI 10.17487/RFC8047, February 2017, <https://www.rfc-editor.org/info/rfc8047>.

[RFC9028] Keränen, A., Melén, J., and M. Komu, Ed., "Native NAT Traversal Mode for the Host Identity Protocol", RFC 9028, DOI 10.17487/RFC9028, July 2021, <https://www.rfc-editor.org/info/rfc9028>.

14.2. Informative References

[amir-hip] Amir, K., Forsgren, H., Grahn, K., Karvi, T., and G. Pulkkis, "Security and Trust of Public Key Cryptography for HIP and HIP Multicast", International Journal of Dependable and Trustworthy Information Systems (IJDTIS), Vol. 2, Issue 3, pp. 17-35, DOI 10.4018/jdtis.2011070102, 2013, <https://doi.org/10.4018/jdtis.2011070102>.

[aura-dos] Aura, T., Nikander, P., and J. Leiwo, "DOS-Resistant Authentication with Client Puzzles", 8th International Workshop on Security Protocols, Security Protocols 2000, Lecture Notes in Computer Science, Vol. 2133, pp. 170-177, Springer, DOI 10.1007/3-540-44810-1_22, September 2001, <https://doi.org/10.1007/3-540-44810-1_22>.

[beal-dos] Beal, J. and T. Shepard, "Deamplification of DoS Attacks via Puzzles", October 2004.

[camarillo-p2psip] Camarillo, G., Mäenpää, J., Keränen, A., and V. Anderson, "Reducing delays related to NAT traversal in P2PSIP session establishments", IEEE Consumer Communications and Networking Conference (CCNC), pp. 549-553, DOI 10.1109/CCNC.2011.5766540, 2011, <https://doi.org/10.1109/CCNC.2011.5766540>.

[chiappa-endpoints] Chiappa, J., "Endpoints and Endpoint Names: A Proposed Enhancement to the Internet Architecture", 1999, <http://mercury.lcs.mit.edu/~jnc/tech/endpoints.txt>.

[heer-end-host] Heer, T., Hummen, R., Komu, M., Gotz, S., and K. Wehrle, "End-Host Authentication and Authorization for Middleboxes Based on a Cryptographic Namespace", 2009 IEEE International Conference on Communications, DOI 10.1109/ICC.2009.5198984, 2009, <https://doi.org/10.1109/ICC.2009.5198984>.

[heer-midauth] Heer, T., Ed., Hummen, R., Wehrle, K., and M. Komu, "End-Host Authentication for HIP Middleboxes", Work in Progress, Internet-Draft, draft-heer-hip-middle-auth-04, 31 October 2011, <https://datatracker.ietf.org/doc/html/draft-heer-hip-middle-auth-04>.

[henderson-vpls] Henderson, T. R., Venema, S. C., and D. Mattes, "HIP-based Virtual Private LAN Service (HIPLS)", Work in Progress, Internet-Draft, draft-henderson-hip-vpls-11, 3 August 2016, <https://datatracker.ietf.org/doc/html/draft-henderson-hip-vpls-11>.

[hip-dex] Moskowitz, R., Ed., Hummen, R., and M. Komu, "HIP Diet EXchange (DEX)", Work in Progress, Internet-Draft, draft-ietf-hip-dex-24, 19 January 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-hip-dex-24>.

[hip-lte] Liyanage, M., Kumar, P., Ylianttila, M., and A. Gurtov, "Novel secure VPN architectures for LTE backhaul networks", Security and Communication Networks, Vol. 9, pp. 1198-1215, DOI 10.1002/sec.1411, January 2016, <https://doi.org/10.1002/sec.1411>.

[hip-srtp] Tschofenig, H., Shanmugam, M., and F. Muenz, "Using SRTP transport format with HIP", Work in Progress, Internet-Draft, draft-tschofenig-hiprg-hip-srtp-02, 25 October 2006, <https://datatracker.ietf.org/doc/html/draft-tschofenig-hiprg-hip-srtp-02>.

[hummen] Hummen, R., Hiller, J., Henze, M., and K. Wehrle, "Slimfit - A HIP DEX compression layer for the IP-based Internet of Things", 2013 IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), pp. 259-266, DOI 10.1109/WiMOB.2013.6673370, October 2013, <https://doi.org/10.1109/WiMOB.2013.6673370>.

[IEEE.802.15.4] IEEE, "IEEE Standard for Low-Rate Wireless Networks", IEEE Standard 802.15.4, DOI 10.1109/IEEESTD.2020.9144691, July 2020, <https://ieeexplore.ieee.org/document/9144691>.

[IEEE.802.15.9] IEEE, "IEEE Draft Recommended Practice for Transport of Key Management Protocol (KMP) Datagrams", IEEE P802.15.9/D04, May 2015.

[karvonen-usable] Karvonen, K., Komu, M., and A. Gurtov, "Usable security management with host identity protocol", 2009 IEEE/ACS International Conference on Computer Systems and Applications, pp. 279-286, DOI 10.1109/AICCSA.2009.5069337, 2009, <https://doi.org/10.1109/AICCSA.2009.5069337>.

[komu-cloud] Komu, M., Sethi, M., Mallavarapu, R., Oirola, H., Khan, R., and S. Tarkoma, "Secure Networking for Virtual Machines in the Cloud", 2012 IEEE International Conference on Cluster Computing Workshops, pp. 88-96, DOI 10.1109/ClusterW.2012.29, 2012, <https://doi.org/10.1109/ClusterW.2012.29>.

[komu-diss] Komu, M., "A Consolidated Namespace for Network Applications, Developers, Administrators and Users", Dissertation, Aalto University, Espoo, Finland, ISBN 978-952-60-4904-5 (printed), ISBN 978-952-60-4905-2 (electronic), December 2012.

[komu-leap] Komu, M. and J. Lindqvist, "Leap-of-Faith Security is Enough for IP Mobility", 2009 6th IEEE Consumer Communications and Networking Conference, Las Vegas, NV, USA, pp. 1-5, DOI 10.1109/CCNC.2009.4784729, January 2009, <https://doi.org/10.1109/CCNC.2009.4784729>.

[komu-mitigation] Komu, M., Tarkoma, S., and A. Lukyanenko, "Mitigation of Unsolicited Traffic Across Domains with Host Identities and Puzzles", 15th Nordic Conference on Secure IT Systems, NordSec 2010, Lecture Notes in Computer Science, Vol. 7127, pp. 33-48, Springer, ISBN 978-3-642-27936-2, DOI 10.1007/978-3-642-27937-9_3, October 2010, <https://doi.org/10.1007/978-3-642-27937-9_3>.

[kovacshazi-host] Kovacshazi, Z. and R. Vida, "Host Identity Specific Multicast", International Conference on Networking and Services (ICNS '07), Athens, Greece, pp. 1-1, DOI 10.1109/ICNS.2007.66, 2007, <https://doi.org/10.1109/ICNS.2007.66>.

[levae-barriers] Levä, T., Komu, M., and S. Luukkainen, "Adoption barriers of network layer protocols: the case of host identity protocol", Computer Networks, Vol. 57, Issue 10, pp. 2218-2232, ISSN 1389-1286, DOI 10.1016/j.comnet.2012.11.024, March 2013, <https://doi.org/10.1016/j.comnet.2012.11.024>.

[lindqvist-enterprise] Lindqvist, J., Vehmersalo, E., Komu, M., and J. Manner, "Enterprise Network Packet Filtering for Mobile Cryptographic Identities", International Journal of Handheld Computing Research (IJHCR), Vol. 1, Issue 1, pp. 79-94, DOI 10.4018/jhcr.2010090905, 2010, <https://doi.org/10.4018/jhcr.2010090905>.

[Nik2001] Nikander, P., "Denial-of-Service, Address Ownership, and Early Authentication in the IPv6 World", 9th International Workshop on Security Protocols, Security Protocols 2001, Lecture Notes in Computer Science, Vol. 2467, pp. 12-21, Springer, DOI 10.1007/3-540-45807-7_3, 2002, <https://doi.org/10.1007/3-540-45807-7_3>.

[nsrg-report] Lear, E. and R. Droms, "What's In A Name: Thoughts from the NSRG", Work in Progress, Internet-Draft, draft-irtf-nsrg-report-10, 22 September 2003, <https://datatracker.ietf.org/doc/html/draft-irtf-nsrg-report-10>.

[paine-hip] Paine, R. H., "Beyond HIP: The End to Hacking As We Know It", BookSurge Publishing, ISBN-10 1439256047, ISBN-13 978-1439256046, 2009.

[pham-leap] Pham, V. and T. Aura, "Security Analysis of Leap-of-Faith Protocols", 7th International ICST Conference, Security and Privacy for Communication Networks, SecureComm 2011, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Vol. 96, DOI 10.1007/978-3-642-31909-9_19, 2012, <https://doi.org/10.1007/978-3-642-31909-9_19>.

[ranjbar-synaptic] Ranjbar, A., Komu, M., Salmela, P., and T. Aura, "SynAPTIC: Secure and Persistent Connectivity for Containers", 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), Madrid, 2017, pp. 262-267, DOI 10.1109/CCGRID.2017.62, 2017, <https://doi.org/10.1109/CCGRID.2017.62>.

[RFC2136] Vixie, P., Ed., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, DOI 10.17487/RFC2136, April 1997, <https://www.rfc-editor.org/info/rfc2136>.

[RFC2766] Tsirtsis, G. and P. Srisuresh, "Network Address Translation - Protocol Translation (NAT-PT)", RFC 2766, DOI 10.17487/RFC2766, February 2000, <https://www.rfc-editor.org/info/rfc2766>.

[RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, DOI 10.17487/RFC3022, January 2001, <https://www.rfc-editor.org/info/rfc3022>.

[RFC3102] Borella, M., Lo, J., Grabelsky, D., and G. Montenegro, "Realm Specific IP: Framework", RFC 3102, DOI 10.17487/RFC3102, October 2001, <https://www.rfc-editor.org/info/rfc3102>.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, Ed., "Extensible Authentication Protocol (EAP)", RFC 3748, DOI 10.17487/RFC3748, June 2004, <https://www.rfc-editor.org/info/rfc3748>.

[RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)", RFC 3972, DOI 10.17487/RFC3972, March 2005, <https://www.rfc-editor.org/info/rfc3972>.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <https://www.rfc-editor.org/info/rfc4033>.

[RFC4225] Nikander, P., Arkko, J., Aura, T., Montenegro, G., and E. Nordmark, "Mobile IP Version 6 Route Optimization Security Design Background", RFC 4225, DOI 10.17487/RFC4225, December 2005, <https://www.rfc-editor.org/info/rfc4225>.

[RFC4380] Huitema, C., "Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs)", RFC 4380, DOI 10.17487/RFC4380, February 2006, <https://www.rfc-editor.org/info/rfc4380>.

[RFC4423] Moskowitz, R. and P. Nikander, "Host Identity Protocol (HIP) Architecture", RFC 4423, DOI 10.17487/RFC4423, May 2006, <https://www.rfc-editor.org/info/rfc4423>.

[RFC5218] Thaler, D. and B. Aboba, "What Makes for a Successful Protocol?", RFC 5218, DOI 10.17487/RFC5218, July 2008, <https://www.rfc-editor.org/info/rfc5218>.

[RFC5338] Henderson, T., Nikander, P., and M. Komu, "Using the Host Identity Protocol with Legacy Applications", RFC 5338, DOI 10.17487/RFC5338, September 2008, <https://www.rfc-editor.org/info/rfc5338>.

[RFC5887] Carpenter, B., Atkinson, R., and H. Flinck, "Renumbering Still Needs Work", RFC 5887, DOI 10.17487/RFC5887, May 2010, <https://www.rfc-editor.org/info/rfc5887>.

[RFC6078] Camarillo, G. and J. Melen, "Host Identity Protocol (HIP) Immediate Carriage and Conveyance of Upper-Layer Protocol Signaling (HICCUPS)", RFC 6078, DOI 10.17487/RFC6078, January 2011, <https://www.rfc-editor.org/info/rfc6078>.

[RFC6250] Thaler, D., "Evolution of the IP Model", RFC 6250, DOI 10.17487/RFC6250, May 2011, <https://www.rfc-editor.org/info/rfc6250>.

[RFC6281] Cheshire, S., Zhu, Z., Wakikawa, R., and L. Zhang, "Understanding Apple's Back to My Mac (BTMM) Service", RFC 6281, DOI 10.17487/RFC6281, June 2011, <https://www.rfc-editor.org/info/rfc6281>.

[RFC6317] Komu, M. and T. Henderson, "Basic Socket Interface Extensions for the Host Identity Protocol (HIP)", RFC 6317, DOI 10.17487/RFC6317, July 2011, <https://www.rfc-editor.org/info/rfc6317>.

[RFC6537] Ahrenholz, J., "Host Identity Protocol Distributed Hash Table Interface", RFC 6537, DOI 10.17487/RFC6537, February 2012, <https://www.rfc-editor.org/info/rfc6537>.

[RFC6538] Henderson, T. and A. Gurtov, "The Host Identity Protocol (HIP) Experiment Report", RFC 6538, DOI 10.17487/RFC6538, March 2012, <https://www.rfc-editor.org/info/rfc6538>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <https://www.rfc-editor.org/info/rfc7296>.

[RFC7435] Dukhovni, V., "Opportunistic Security: Some Protection Most of the Time", RFC 7435, DOI 10.17487/RFC7435, December 2014, <https://www.rfc-editor.org/info/rfc7435>.

[sarela-bloom] Särelä, M., Esteve Rothenberg, C., Zahemszky, A., Nikander, P., and J. Ott, "BloomCasting: Security in Bloom Filter Based Multicast", Information Security Technology for Applications, NordSec 2010, Lecture Notes in Computer Science, Vol. 7127, pages 1-16, Springer, DOI 10.1007/978-3-642-27937-9_1, 2012, <https://doi.org/10.1007/978-3-642-27937-9_1>.

[schuetz-intermittent] Schütz, S., Eggert, L., Schmid, S., and M. Brunner, "Protocol enhancements for intermittently connected hosts", ACM SIGCOMM Computer Communication Review, Vol. 35, Issue 3, pp. 5-18, DOI 10.1145/1070873.1070875, July 2005, <https://doi.org/10.1145/1070873.1070875>.

[shields-hip] Shields, C. and J. J. Garcia-Luna-Aceves, "The HIP protocol for hierarchical multicast routing", Proceedings of the seventeenth annual ACM symposium on Principles of distributed computing, pp. 257-266, ISBN 0-89791-977-7, DOI 10.1145/277697.277744, 1998, <https://doi.org/10.1145/277697.277744>.

[tempered-networks] Tempered Networks, "Identity-Defined Network (IDN) Architecture: Unified, Secure Networking Made Simple", White Paper, 2016.

[tritilanunt-dos] Tritilanunt, S., Boyd, C., Foo, E., and J.M.G. Nieto, "Examining the DoS Resistance of HIP", On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops, Lecture Notes in Computer Science, Vol. 4277, pp. 616-625, Springer, DOI 10.1007/11915034_85, 2006, <https://doi.org/10.1007/11915034_85>.

[urien-rfid] Urien, P., Chabanne, H., Pepin, C., Orga, S., Bouet, M., de Cunha, D.O., Guyot, V., Pujolle, G., Paradinas, P., Gressier, E., and J.-F. Susini, "HIP-based RFID Networking Architecture", 2007 IFIP International Conference on Wireless and Optical Communications Networks, pp. 1-5, DOI 10.1109/WOCN.2007.4284140, 2007, <https://doi.org/10.1109/WOCN.2007.4284140>.

[urien-rfid-draft] Urien, P., Lee, G. M., and G. Pujolle, "HIP support for RFIDs", Work in Progress, Internet-Draft, draft-irtf-hiprg-rfid-07, 23 April 2013, <https://datatracker.ietf.org/doc/html/draft-irtf-hiprg-rfid-07>.

[varjonen-split] Varjonen, S., Komu, M., and A. Gurtov, "Secure and Efficient IPv4/IPv6 Handovers Using Host-Based Identifier-Location Split", Journal of Communications Software and Systems, Vol. 6, Issue 1, ISSN 18456421, DOI 10.24138/jcomss.v6i1.193, 2010, <https://doi.org/10.24138/jcomss.v6i1.193>.

[xin-hip-lib] Xin, G., "Host Identity Protocol Version 2.5", Master's Thesis, Aalto University, Espoo, Finland, June 2012.

[ylitalo-diss] Ylitalo, J., "Secure Mobility at Multiple Granularity Levels over Heterogeneous Datacom Networks", Dissertation, Helsinki University of Technology, Espoo, Finland, ISBN 978-951-22-9531-9, 2008.

[ylitalo-spinat] Ylitalo, J., Salmela, P., and H. Tschofenig, "SPINAT: Integrating IPsec into Overlay Routing", First International Conference on Security and Privacy for Emerging Areas in Communication Networks, SECURECOMM'05, Athens, Greece, pp. 315-326, ISBN 0-7695-2369-2, DOI 10.1109/SECURECOMM.2005.53, 2005, <https://doi.org/10.1109/SECURECOMM.2005.53>.

[zhang-revocation] Zhang, D., Kuptsov, D., and S. Shen, "Host Identifier Revocation in HIP", Work in Progress, Internet-Draft, draft-irtf-hiprg-revocation-05, 9 March 2012, <https://datatracker.ietf.org/doc/html/draft-irtf-hiprg-revocation-05>.

[zhu-hip] Zhu, X., Ding, Z., and X. Wang, "A Multicast Routing Algorithm Applied to HIP-Multicast Model", 2011 International Conference on Network Computing and Information Security, Guilin, China, pp. 169-174, DOI 10.1109/NCIS.2011.42, 2011, <https://doi.org/10.1109/NCIS.2011.42>.

[zhu-secure] Zhu, X. and J. W. Atwood, "A Secure Multicast Model for Peer-to-Peer and Access Networks Using the Host Identity Protocol", 2007 4th IEEE Consumer Communications and Networking Conference, Las Vegas, NV, USA, pages 1098-1102, DOI 10.1109/CCNC.2007.221, 2007, <https://doi.org/10.1109/CCNC.2007.221>.

Appendix A. Design Considerations

A.1. Benefits of HIP

In the beginning, the network layer protocol (i.e., IP) had the following four "classic" invariants:

1. Non-mutable: The address sent is the address received.
2. Non-mobile: The address doesn't change during the course of an "association".
3. Reversible: A return header can always be formed by reversing the source and destination addresses.
4. Omniscient: Each host knows what address a partner host can use to send packets to it.

Actually, the fourth can be inferred from 1 and 3, but it is worth mentioning explicitly for reasons that will be obvious soon if not already.

In the current "post-classic" world, we are intentionally trying to get rid of the second invariant (both for mobility and for multihoming), and we have been forced to give up the first and the fourth. Realm Specific IP [RFC3102] is an attempt to reinstate the fourth invariant without the first invariant. IPv6 attempts to reinstate the first invariant.

Few client-side systems on the Internet have DNS names that are meaningful. That is, if they have a Fully Qualified Domain Name (FQDN), that name typically belongs to a NAT device or a dial-up server, and does not really identify the system itself but its current connectivity. FQDNs (and their extensions as email names) are application-layer names; more frequently naming services than particular systems. This is why many systems on the Internet are not registered in the DNS; they do not have services of interest to other Internet hosts.

DNS names are references to IP addresses. This only demonstrates the interrelationship of the networking and application layers. DNS, as the Internet's only deployed and distributed database, is also the repository of other namespaces, due in part to DNSSEC and application-specific key records. Although each namespace can be stretched (IP with v6, DNS with KEY records), neither can adequately provide for host authentication or act as a separation between internetworking and transport layers.

The Host Identity (HI) namespace fills an important gap between the IP and DNS namespaces. An interesting thing about the HI is that it actually allows a host to give up all but the 3rd network-layer invariant. That is to say, as long as the source and destination addresses in the network-layer protocol are reversible, HIP takes care of host identification, and reversibility allows a local host to receive a packet back from a remote host. The address changes occurring during NAT transit (non-mutable) or host movement (non-omniscient or non-mobile) can be managed by the HIP layer.

With the exception of high-performance computing applications, the sockets API is the most common way to develop network applications. Applications use the sockets API either directly or indirectly through some libraries or frameworks. However, the sockets API is based on the assumption of static IP addresses, and DNS with its lifetime values was invented at later stages during the evolution of the Internet. Hence, the sockets API does not deal with the lifetime of addresses [RFC6250]. As the majority of the end-user equipment is mobile today, their addresses are effectively ephemeral, but the sockets API still gives a fallacious illusion of persistent IP addresses to the unwary developer. HIP can be used to solidify this illusion because HIP provides persistent, surrogate addresses to the application layer in the form of LSIs and HITs.

The persistent identifiers as provided by HIP are useful in multiple scenarios (see, e.g., [ylitalo-diss] or [komu-diss] for a more elaborate discussion):

- When a mobile host moves physically between two different WLAN networks and obtains a new address, an application using the identifiers remains isolated regardless of the topology changes while the underlying HIP layer reestablishes connectivity (i.e., a horizontal handoff).

- Similarly, the application utilizing the identifiers remains again unaware of the topological changes when the underlying host equipped with WLAN and cellular network interfaces switches between the two different access technologies (i.e., a vertical handoff).
- Even when hosts are located in private address realms, applications can uniquely distinguish different hosts from each other based on their identifiers. In other words, it can be stated that HIP improves Internet transparency for the application layer [komu-diss].
- Site renumbering events for services can occur due to corporate mergers or acquisitions, or by changes in Internet service provider. They can involve changing the entire network prefix of an organization, which is problematic due to hard-coded addresses in service configuration files or cached IP addresses at the client side [RFC5887]. Considering such human errors, a site employing location-independent identifiers as promoted by HIP may experience fewer problems while renumbering their network.
- More agile IPv6 interoperability can be achieved, as discussed in Section 4.4. IPv6-based applications can communicate using HITs with IPv4-based applications that are using LSIs. Additionally, the underlying network type (IPv4 or IPv6) becomes independent of the addressing family of the application.
- HITs (or LSIs) can be used in IP-based access control lists as a more secure replacement for IPv6 addresses. Besides security, HIT-based access control has two other benefits. First, the use of HITs can potentially halve the size of access control lists because separate rules for IPv4 are not needed [komu-diss]. Second, HIT-based configuration rules in HIP-aware middleboxes remain static and independent of topology changes, thus simplifying administrative efforts particularly for mobile environments. For instance, the benefits of HIT-based access control have been harnessed in the case of HIP-aware firewalls, but can be utilized directly at the end-hosts as well [RFC6538].

While some of these benefits could be and have been redundantly implemented by individual applications, providing such generic functionality at the lower layers is useful because it reduces software development effort and networking software bugs (as the layer is tested with multiple applications). It also allows the developer to focus on building the application itself rather than delving into the intricacies of mobile networking, thus facilitating separation of concerns.

HIP could also be realized by combining a number of different protocols, but the complexity of the resulting software may become substantially larger, and the interaction between multiple, possibly layered protocols may have adverse effects on latency and throughput. It is also worth noting that virtually nothing prevents realizing the HIP architecture, for instance, as an application-layer library, which has been actually implemented in the past [xin-hip-lib]. However, the trade-off in moving the HIP layer to the application layer is that legacy applications may not be supported.

A.2. Drawbacks of HIP

In computer science, many problems can be solved with an extra layer of indirection. However, the indirection always involves some costs as there is no such a thing as a "free lunch". In the case of HIP, the main costs could be stated as follows:

- In general, an additional layer and a namespace always involve some initial effort in terms of implementation, deployment, and maintenance. Some education of developers and administrators may also be needed. However, the HIP community at the IETF has spent years in experimenting, exploring, testing, documenting, and implementing HIP to ease the adoption costs.
- HIP introduces a need to manage HIs and requires a centralized approach to manage HIP-aware endpoints at scale. What were formerly IP address-based ACLs are now trusted HITs, and the HIT-to-IP address mappings as well as access policies must be managed. HIP-aware endpoints must also be able to operate autonomously to ensure mobility and availability (an endpoint must be able to run without having to have a persistent management connection). The users who want this better security and mobility of HIs instead of IP address-based ACLs have to then manage this additional 'identity layer' in a nonpersistent fashion. As exemplified in Appendix A.3.5, these challenges have been already solved in an infrastructure setting to distribute policy and manage the mappings and trust relationships between HIP-aware endpoints.
- HIP decouples identifier and locator roles of IP addresses. Consequently, a mapping mechanism is needed to associate them together. A failure to map a HIT to its corresponding locator may result in failed connectivity because a HIT is "flat" by its nature and cannot be looked up from the hierarchically organized DNS. HITs are flat by design due to a security trade-off. The more bits that are allocated for the hash in the HIT, the less likely there will be (malicious) collisions.
- From performance viewpoint, HIP control and data plane processing introduces some overhead in terms of throughput and latency as elaborated below.

Related to deployment drawbacks, firewalls are commonly used to control access to various services and devices in the current Internet. Since HIP introduces an additional namespace, it is expected that the HIP namespace would be filtered for unwanted connectivity also. While this can be achieved with existing tools directly in the end-hosts, filtering at the middleboxes requires modifications to existing firewall software or additional middleboxes [RFC6538].

The key exchange introduces some extra latency (two round trips) in the initial transport-layer connection establishment between two hosts. With TCP, additional delay occurs if the underlying network stack implementation drops the triggering SYN packet during the key exchange. The same cost may also occur during HIP handoff procedures. However, subsequent TCP sessions using the same HIP association will not bear this cost (within the key lifetime). Both the key exchange and handoff penalties can be minimized by caching TCP packets. The latter case can further be optimized with TCP user timeout extensions [RFC5482] as described in further detail by Schütz et al. [schuetz-intermittent].

The most CPU-intensive operations involve the use of the asymmetric keys and Diffie-Hellman key derivation at the control plane, but this occurs only during the key exchange, its maintenance (handoffs and refreshing of key material), and teardown procedures of HIP associations. The data plane is typically implemented with ESP because it has a smaller overhead due to symmetric key encryption. Naturally, even ESP involves some overhead in terms of latency (processing costs) and throughput (tunneling) (see, e.g., [ylitalo-diss] for a performance evaluation).

A.3. Deployment and Adoption Considerations

This section describes some deployment and adoption considerations related to HIP from a technical perspective.

A.3.1. Deployment Analysis

HIP has been adapted and deployed in an industrial control network in a production factory, in which HIP's strong network-layer identity supports the secure coexistence of the control network with many untrusted network devices operated by third-party vendors [paine-hip]. Similarly, HIP has also been included in a security product to support Layer 2 VPNs [henderson-vpls] to enable security zones in a supervisory control and data acquisition (SCADA) network. However, HIP has not been a "wild success" [RFC5218] in the Internet as argued by Levä et al. [levae-barriers]. Here, we briefly highlight some of their findings based on interviews with 19 experts from the industry and academia.

From a marketing perspective, the demand for HIP has been low and substitute technologies have been favored. Another identified reason has been that some technical misconceptions related to the early stages of HIP specifications still persist. Two identified misconceptions are that HIP does not support NAT traversal and that HIP must be implemented in the OS kernel. Both of these claims are untrue; HIP does have NAT traversal extensions [RFC9028], and kernel modifications can be avoided with modern operating systems by diverting packets for userspace processing.

The analysis by Levä et al. clarifies infrastructural requirements for HIP. In a minimal setup, a client and server machine have to run HIP software. However, to avoid manual configurations, usually DNS records for HIP are set up. For instance, the popular DNS server software Bind9 does not require any changes to accommodate DNS records for HIP because they can be supported in binary format in its configuration files [RFC6538]. HIP rendezvous servers and firewalls are optional. No changes are required to network address points, NATs, edge routers, or core networks. HIP may require holes in legacy firewalls.

The analysis also clarifies the requirements for the host components that consist of three parts. First, a HIP control plane component is required, typically implemented as a userspace daemon. Second, a data plane component is needed. Most HIP implementations utilize the so-called Bound End-to-End Tunnel (BEET) mode of ESP that has been available since Linux kernel 2.6.27, but the BEET mode is also included as a userspace component in a few of the implementations. Third, HIP systems usually provide a DNS proxy for the local host that translates HIP DNS records to LSIs and HITs, and communicates the corresponding locators to the HIP userspace daemon.

While the third component is not mandatory, it is very useful for avoiding manual configurations. The three components are further described in the HIP experiment report [RFC6538].

Based on the interviews, Levä et al. suggest further directions to facilitate HIP deployment. Transitioning a number of HIP specifications to the Standards Track in the IETF has already taken place, but the authors suggest other additional measures based on the interviews. As a more radical measure, the authors suggest to implement HIP as a purely application-layer library [xin-hip-lib] or other kind of middleware. On the other hand, more conservative measures include focusing on private deployments controlled by a single stakeholder. As a more concrete example of such a scenario, HIP could be used by a single service provider to facilitate secure connectivity between its servers [komu-cloud].

A.3.2. HIP in 802.15.4 Networks

The IEEE 802 standards have been defining MAC-layer security. Many of these standards use Extensible Authentication Protocol (EAP) [RFC3748] as a Key Management System (KMS) transport, but some like IEEE 802.15.4 [IEEE.802.15.4] leave the KMS and its transport as "out of scope".

HIP is well suited as a KMS in these environments:

- HIP is independent of IP addressing and can be directly transported over any network protocol.
- Master keys in 802 protocols are commonly pair-based with group keys transported from the group controller using pairwise keys.
- Ad hoc 802 networks can be better served by a peer-to-peer KMS than the EAP client/server model.
- Some devices are very memory constrained, and a common KMS for both MAC and IP security represents a considerable code savings.

A.3.3. HIP and Internet of Things

HIP requires certain amount computational resources from a device due to cryptographic processing. HIP scales down to phones and small system-on-chip devices (such as Raspberry Pis, Intel Edison), but small sensors operating with small batteries have remained problematic. Different extensions to the HIP have been developed to scale HIP down to smaller devices, typically with different security trade-offs. For example, the non-cryptographic identifiers have been proposed in RFID scenarios. The Slimfit approach [hummen] proposes a compression layer for HIP to make it more suitable for constrained networks. The approach is applied to a lightweight version of HIP (i.e., "Diet HIP") in order to scale down to small sensors.

The HIP Diet EXchange (DEX) [hip-dex] design aims to reduce the overhead of the employed cryptographic primitives by omitting public-key signatures and hash functions. In doing so, the main goal is to still deliver security properties similar to the Base Exchange (BEX).

DEX is primarily designed for computation- or memory-constrained sensor/actuator devices. Like BEX, it is expected to be used together with a suitable security protocol such as the ESP for the protection of upper-layer protocol data. In addition, DEX can also be used as a keying mechanism for security primitives at the MAC layer, e.g., for IEEE 802.15.9 networks [IEEE.802.15.9].

The main differences between HIP BEX and DEX are:

1. Minimum collection of cryptographic primitives to reduce the protocol overhead.
   - Static Elliptic Curve Diffie-Hellman (ECDH) key pairs for peer authentication and encryption of the session key.
   - AES-CTR for symmetric encryption and AES-CMAC for MACing function.
   - A simple fold function for HIT generation.
2. Forfeit of perfect forward secrecy with the dropping of an ephemeral Diffie-Hellman key agreement.
3. Forfeit of digital signatures with the removal of a hash function. Reliance on the ECDH-derived key used in HIP_MAC to prove ownership of the private key.
4. Diffie-Hellman derived key ONLY used to protect the HIP packets. A separate secret exchange within the HIP packets creates the session key(s).
5. Optional retransmission strategy tailored to handle the potentially extensive processing time of the employed cryptographic operations on computationally constrained devices.

A.3.4. Infrastructure Applications

The HIP experimentation report [RFC6538] enumerates a number of client and server applications that have been trialed with HIP. Based on the report, this section highlights and complements some potential ways how HIP could be exploited in existing infrastructure such as routers, gateways, and proxies.

HIP has been successfully used with forward web proxies (i.e., client-side proxies). HIP was used between a client host (web browser) and a forward proxy (Apache server) that terminated the HIP/ESP tunnel. The forward web proxy translated HIP-based traffic originating from the client into non-HIP traffic towards any web server in the Internet. Consequently, the HIP-capable client could communicate with HIP-incapable web servers. This way, the client could utilize mobility support as provided by HIP while using the fixed IP address of the web proxy, for instance, to access services that were allowed only from the IP address range of the proxy.

HIP with reverse web proxies (i.e., server-side proxies) has also been investigated, as described in more detail in [komu-cloud]. In this scenario, a HIP-incapable client accessed a HIP-capable web service via an intermediary load balancer (a web-based load balancer implementation called HAProxy). The load balancer translated non-HIP traffic originating from the client into HIP-based traffic for the web service (consisting of front-end and back-end servers). Both the load balancer and the web service were located in a data center. One of the key benefits for encrypting the web traffic with HIP in this scenario was supporting a private-public cloud

scenario (i.e., hybrid cloud) where the load balancer, front-end servers, and back-end servers were located in different data centers, and thus the traffic needed to be protected when it passed through potentially insecure networks between the borders of the private and public clouds.

While HIP could be used to secure access to intermediary devices (e.g., access to switches with legacy telnet), it has also been used to secure intermittent connectivity between middlebox infrastructure. For instance, earlier research [komu-mitigation] utilized HIP between Simple Mail Transport Protocol (SMTP) servers in order to exploit the computational puzzles of HIP as a spam mitigation mechanism. A rather obvious practical challenge in this approach was the lack of HIP adoption on existing SMTP servers.

To avoid deployment hurdles with existing infrastructure, HIP could be applied in the context of new protocols with little deployment. Namely, HIP has been studied in the context of a new protocol, peer-to-peer SIP [camarillo-p2psip]. The work has resulted in a number of related RFCs [RFC6078], [RFC6079], and [RFC7086]. The key idea in the research work was to avoid redundant, time-consuming ICE procedures by grouping different connections (i.e., SIP and media streams) together using the low-layer HIP, which executes NAT traversal procedures only once per host. An interesting aspect in the approach was the use of P2P-SIP infrastructure as rendezvous servers for the HIP control plane instead of utilizing the traditional HIP rendezvous services [RFC8004].

Researchers have proposed using HIP in cellular networks as a mobility, multihoming, and security solution. [hip-lte] provides a security analysis and simulation measurements of using HIP in Long Term Evolution (LTE) backhaul networks.

HIP has been studied for securing cloud internal connectivity. First with virtual machines [komu-cloud] and then between Linux containers [ranjbar-synaptic]. In both cases, HIP was suggested as a solution to NAT traversal that could be utilized both internally by a cloud network and between multi-cloud deployments. Specifically in the former case, HIP was beneficial sustaining connectivity with a virtual machine while it migrated to a new location. In the latter case, a Software-Defined Networking (SDN) controller acted as a rendezvous server for HIP-capable containers. The controller enforced strong replay protection by adding middlebox nonces [heer-end-host] to the passing HIP base exchange and UPDATE messages.

A.3.5. Management of Identities in a Commercial Product

Tempered Networks provides HIP-based products. They refer to their platform as Identity-Defined Networking (IDN) [tempered-networks] because of HIP's identity-first networking architecture. Their objective has been to make it simple and nondisruptive to deploy HIP-enabled services widely in production environments with the purpose of enabling transparent device authentication and authorization, cloaking, segmentation, and end-to-end networking. The goal is to eliminate much of the circular dependencies, exploits, and layered complexity of traditional "address-defined networking" that prevents mobility and verifiable device access control. The products in the portfolio of Tempered Networks utilize HIP are as follows:

HIP Switches / Gateways
: These are physical or virtual appliances that serve as the HIP gateway and policy enforcement point for non-HIP-aware applications and devices located behind it. No IP or infrastructure changes are required in order to connect, cloak, and protect the non-HIP-aware devices. Currently known supported platforms for HIP gateways are x86 and ARM chipsets, ESXi, Hyper-V, KVM, AWS, Azure, and Google clouds.

HIP Relays / Rendezvous
: These are physical or virtual appliances that serve as identity-based routers authorizing and bridging HIP endpoints without decrypting the HIP session. A HIP relay can be deployed as a standalone appliance or in a cluster for horizontal scaling. All HIP-aware endpoints and the devices they're connecting and protecting can remain privately addressed. The appliances eliminate IP conflicts, tunnel through NAT and carrier-grade NAT, and require no changes to the underlying infrastructure. The only requirement is that a HIP endpoint should have outbound access to the Internet and that a HIP Relay should have a public address.

HIP-Aware Clients and Servers
: This is software that is installed in the host's network stack and enforces policy for that host. HIP clients support split tunneling. Both the HIP client and HIP server can interface with the local host firewall, and the HIP server can be locked down to listen only on the port used for HIP, making the server invisible from unauthorized devices. Currently known supported platforms are Windows, OS X, iOS, Android, Ubuntu, CentOS, and other Linux derivatives.

Policy Orchestration Managers
: These physical or virtual appliances serve as the engine to define and distribute network and security policy (HI and IP mappings, overlay networks, and whitelist policies, etc.) to HIP-aware endpoints. Orchestration does not need to persist to the HIP endpoints and vice versa, allowing for autonomous host networking and security.

A.4. Answers to NSRG Questions

The IRTF Name Space Research Group has posed a number of evaluating questions in their report [nsrg-report]. In this section, we provide answers to these questions.

1. How would a stack name improve the overall functionality of the Internet?

   HIP decouples the internetworking layer from the transport layer, allowing each to evolve separately. The decoupling makes end-host mobility and multihoming easier, also across IPv4 and IPv6 networks. HIs make network renumbering easier, and they also make process migration and clustered servers easier to implement. Furthermore, being cryptographic in nature, they provide the basis for solving the security problems related to end-host mobility and multihoming.

2. What does a stack name look like?

   A HI is a cryptographic public key. However, instead of using the keys directly, most protocols use a fixed-size hash of the public key.

3. What is its lifetime?

HIP provides both stable and temporary Host Identifiers. Stable HIs are typically long-lived, with a lifetime of years or more. The lifetime of temporary HIs depends on how long the upper-layer connections and applications need them, and can range from a few seconds to years.

4. Where does it live in the stack?

   The HIs live between the transport and internetworking layers.

5. How is it used on the endpoints?

   The Host Identifiers may be used directly or indirectly (in the form of HITs or LSIs) by applications when they access network services. Additionally, the Host Identifiers, as public keys, are used in the built-in key agreement protocol, called the HIP base exchange, to authenticate the hosts to each other.

6. What administrative infrastructure is needed to support it?

   In some environments, it is possible to use HIP opportunistically, without any infrastructure. However, to gain full benefit from HIP, the HIs must be stored in the DNS or a PKI, and the rendezvous mechanism is needed [RFC8005].

7. If we add an additional layer, would it make the address list in SCTP unnecessary?

   Yes

8. What additional security benefits would a new naming scheme offer?

   HIP reduces dependency on IP addresses, making the so-called address ownership [Nik2001] problems easier to solve. In practice, HIP provides security for end-host mobility and multihoming. Furthermore, since HIP Host Identifiers are public keys, standard public key certificate infrastructures can be applied on the top of HIP.

9. What would the resolution mechanisms be, or what characteristics of a resolution mechanisms would be required?

   For most purposes, an approach where DNS names are resolved simultaneously to HIs and IP addresses is sufficient. However, if it becomes necessary to resolve HIs into IP addresses or back to DNS names, a flat resolution infrastructure is needed. Such an infrastructure could be based on the ideas of Distributed Hash Tables, but would require significant new development and deployment.

Acknowledgments

For the people historically involved in the early stages of HIP, see the Acknowledgments section in the Host Identity Protocol specification.

During the later stages of this document, when the editing baton was transferred to Pekka Nikander, the comments from the early implementers and others, including Jari Arkko, Jeff Ahrenholz, Tom Henderson, Petri Jokela, Miika Komu, Mika Kousa, Andrew McGregor, Jan Melen, Tim Shepard, Jukka Ylitalo, Sasu Tarkoma, and Jorma Wall, were invaluable. Also, the comments from Lars Eggert, Spencer Dawkins, Dave Crocker, and Erik Giesa were also useful.

The authors want to express their special thanks to Tom Henderson, who took the burden of editing the document in response to IESG comments at the time when both of the authors were busy doing other things. Without his perseverance, the original document might have never made it as RFC 4423.

This main effort to update and move HIP forward within the IETF process owes its impetus to a number of HIP development teams. The authors are grateful for Boeing, Helsinki Institute for Information Technology (HIIT), NomadicLab of Ericsson, and the three universities: RWTH Aachen, Aalto, and University of Helsinki for their efforts. Without their collective efforts, HIP would have withered as on the IETF vine as a nice concept.

Thanks also to Suvi Koskinen for her help with proofreading and with the reference jungle.

Authors' Addresses

Robert Moskowitz (EDITOR)
HTT Consulting
Oak Park, Michigan
United States of America
Email: rgm@labs.htt-consult.com

Miika Komu
Ericsson
Hirsalantie 11
FI-02420 Jorvas
Finland
Email: miika.komu@ericsson.com