The Shim6 Architecture for IPv6 Multihoming

Alberto García-Martínez and Marcelo Bagnulo, Universidad Carlos III de Madrid
Iljitsch van Beijnum, IMDEA Networks

Abstract

The Shim6 architecture enables IPv6 multihoming without compromising the scalability of the global routing system by using provider aggregatable addresses. To do so, hosts use different addresses as locators for data packet transmission, but present the same source and destination identifier pair to transport and upper layers. The components of this architecture are the Shim6 entity, which maps and translates upper-layer identifiers and locators for remote hosts; the Shim6 protocol, which exchanges mapping information between two hosts that communicate; and the REAP protocol, which monitors the existing unidirectional paths and finds new valid locator combinations in case of failure. To protect against new vulnerabilities this architecture may introduce compared to IPv6, Shim6 hosts use either cryptographically generated addresses or hash-based addresses.

Introduction

Multihoming (i.e., the connection to the Internet through multiple providers), has been widely adopted by Internet sites, mainly to provide fault tolerance. In the multihoming solution currently deployed in the IPv4 Internet, the multihomed site announces a single address block through all its providers using Border Gateway Protocol (BGP). As a result, for each multihomed site, multiple routes toward the multihomed site are available in the inter-domain routing system. The currently deployed IPv4 multihoming solution is then one of the major contributors to the superlinear growth of the routing tables. Even though some claim it will be possible to build equipment that can handle the explosive growth of the routing tables, the equipment lifetime and therefore the economic viability of the Internet would be affected negatively. This approach is even less suitable for the expected number of multihomed sites in the future IPv6 Internet. To illustrate this, consider the demand for multihoming that can be triggered by the wide adoption of low-budget broadband access technologies such as asymmetric digital subscriber line (ADSL) or cable TV (CATV) in small office/home office (SOHO) environments. To address these concerns, an alternative multihoming solution for IPv6 has been developed in the Shim6 Working Group of the Internet Engineering Task Force (IETF). In this solution, a multihomed network obtains a prefix from the address block of each of its providers, so end hosts are assigned with multiple global IPv6 unicast addresses. In this way the injection of routes into the global routing system associated with individual multihomed end sites or networks is no longer needed. However, to preserve established communications through outages, the endpoints have to change the addresses in use during the lifetime of the communication according to the available providers. Moreover, this address replacement has to be performed in a transparent fashion with respect to transport and application layers, in order to actually preserve the established communication, since current applications and transport layers, such as TCP and UDP, identify the endpoints of a communication through the IP addresses of the nodes involved.

The solution proposed relies on a new sublayer inside the IP layer, the Shim6 sublayer, along with two new protocols, Shim6 [1] and ReAchability Protocol (REAP) [2], which exchange information between the Shim6 sublayers of two communicating hosts. The Shim6 sublayer translates the address used for exchanging packets on the wire (the *locator*) to the constant address that is presented to upper layers (the *upper-layer identifier* [ULID]), and from the ULID to the locators used as source addresses (Fig. 1). In the Shim6 architecture ULIDs are topologically valid addresses, so they are also used as locators. The new sublayer is placed between the IP routing sublayer, performing forwarding functionalities like determining the next hop for outgoing packets, and the IP endpoint sub-layer containing end-to-end mechanisms such as IPsec. The Shim6 protocol exchanges the locators associated with a pair of ULIDs (i.e., establishes a Shim6 context in two communicating nodes). At any given time, a pair of source and destination locators is used for one direction, and a possibly different pair of locators is used for the other one, since the Shim6 framework supports a different path in each direction. Different upper-layer communications (i.e., different TCP connections or UDP exchanges) can use the same Shim6 context. The final component of the architecture is the REAP, a lightweight protocol used to detect failures in the current communicating

0163-6804/10/$25.00 © 2010 IEEE

paths and determine the new paths to use for each unidirectional path.

The rest of the article is organized as follows. In the next two sections we present the Shim6 and REAP protocols, respectively. We then discuss the security vulnerabilities that may arise from the use of both protocols, and the protection measures developed to provide similar security to IPv6, mainly by the use of cryptographic addresses such as cryptographically generated addresses (CGAs) and hash based addresses (HBAs). Then we show an example in which two nodes' Shim6 hosts communicate, and we draw some conclusions.

SHIM6 PROTOCOL

As mentioned earlier, the Shim6 protocol is in charge of making the Shim6 sublayer of the remote host aware of the different locators available for a given communication. To do so, a Shim6 context is established for each pair of communicating hosts.

Consider the case in which one of the parties involved in a current or future communication decides to create a Shim6 context in order to benefit from the enhanced fault tolerance capabilities of multihoming. We refer to the party that decides to initiate the context exchange as the *initiator* and the other party involved in the communication as the *responder*. Note that the initiator may differ from the actual initiator of the communication itself and that a Shim6 context can be created well after a communication has been started (e.g., some heuristic can decide that the communication may live long enough to benefit from the protection provided).

To create a Shim6 session, a four-way handshake as depicted in Fig. 2 is defined, in a similar way to the Host Identity Protocol (HIP) base exchange [3]. In the next paragraphs we summarily describe the process, delaying the description of the exchange of related security information to a later section.

Message `I1` (Initiator 1) is sent by the initiator to request the creation of a context associated with a ULID pair. The ULID pair of the context to be created can be either the locators being used for the `I1` message or a different address pair, in which case the address pair must be included explicitly in the `I1` message. By the use of this first message, Shim6 provides denial-of-service (DoS) attack protection to the responder. This is required because the creation of the context associated with the session implies the storage of information in the responder, and an attacker could try to create a large number of sessions to exhaust these resources. To prevent such attacks, the responder does not create any context related state until the initiator has proven its location by responding in a later packet with a nonce issued by the responder. In general, nonces are used to prevent third parties out of the communicating path interfering with the handshake. Although this security measure does not fully preclude the possibility of DoS attacks, at least it imposes an additional effort for the attacker and provides some tracing capabilities.

The `I1` message also includes an *initiator context tag*, which is a session identifier used to allow the

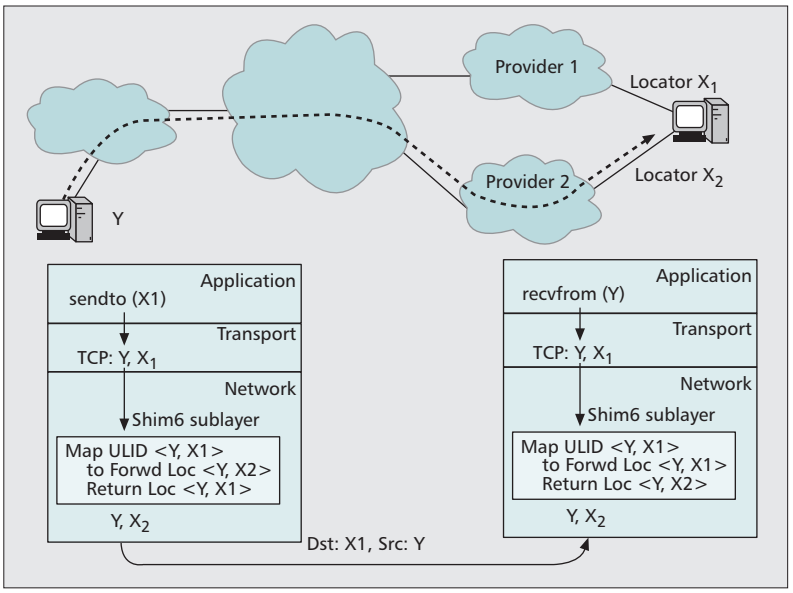

Figure 1. *Overview of Shim6 operation.*

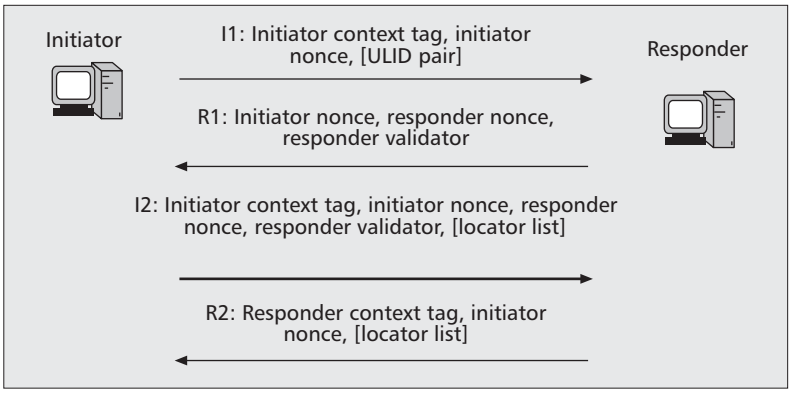

Figure 2. *Shim6 four-way handshake.*

Shim6 sublayer at the initiator to identify the appropriate context for a received data packet in case the locators have changed. Note that many Shim6 contexts may be established for two communicating hosts, using different ULIDs of the hosts but the same locator pair for a given direction. Therefore, some means to identify the context to use when translating a received data packet has to be provided. To do so, data packets sent from the responder to the initiator using locators different from the ULID of the context carry a Shim6 payload extension header containing the initiator context tag. Since all the context tags received by the initiator, even those belonging to different communications, are assigned by the initiator itself, it is easy to ensure its uniqueness. Note that data packets only use the Shim6 payload extension header when a locator different from the ULID is used, so no overhead is introduced to data exchange otherwise.

Upon the reception of the `I1` message, the responder can discard it if there is no multihoming support or interest in enabling multihoming for communications with this host, or reply with an `R1` message. This message contains the *responder validator*, a hash of the context information of `I1`, plus a secret token of the responder, which will allow the responder to check

Context Recovery can be useful for heavy-loaded servers, which can establish a large number of Shim6 contexts with clients, and then discard them aggressively, leaving the initiative in failure detection and recovery to the client.

later if the parameters used to create the state were the same received in I1.

After the reception of R1, the initiator sends the I2 message in which the locator set available at the initiator can be included.

A responder receiving the I2 message creates the SHIM6 context, and replies with an R2 message, in which it includes its own context tag and its locator set. When the initiator receives this message, both communicating nodes know the locators of each peer, and the Shim6 context is established in both ends.

During the communication, it may be required to change the available locator set for a host. For that reason, Update Request and Update Acknowledgment messages are defined. The Update Request message contains the complete set of available locators for a host, and the Update Acknowledgment confirms reception of the update. The semantics of the Update Request operation is to replace the available locator set for the session with the one included in the message. In this way existing locators can be removed, for instance, if a failure occurs and one of the prefixes available in the multihomed site is deprecated. It should be noted that an address can be removed as a locator but still be used as a ULID if that was its role when establishing the communication. It is also interesting to highlight that these messages also provide support for host mobility and site renumbering, since a host that acquires a new locator (either through attachment to another link or renumbering) can inform its peer about it by means of the Shim6 protocol.

There are no protocol messages to close Shim6 sessions, so context discarding is performed independently on each host if no data has been exchanged for a given period of time.

The Shim6 protocol includes some interesting features such as context forking and context recovery. Context forking allows a host to fork an existing Shim6 context in two, to enable the association of different locator sets with each context. In order to support Context Forking, a forked instance identifier (FII) is used to distinguish the original context (FII equal of 0) from the new context (any other value of FII). The node forking the context initiates a Shim6 four-way handshake in which the FII for the new context is included. The other host may also fork a new context as a result of this request.

The context recovery functionality allows recovering a context that has been lost in one of the hosts. If a failure in the communication is detected by the node keeping the context, this node can select new locators and send data packets using the context tag initially assigned by the remote host. The reception of one of these packets in the node that lost the context triggers a simplified message exchange composed of an R1bis message, an I2bis message, and the R2 message. Note that data packets cannot be delivered to the appropriate upper-layer instance until the context has been recovered. Context recovery can be useful for heavy-loaded servers, which can establish a large number of Shim6 contexts with clients and then discard them aggressively, leaving the initiative in failure detection and recovery to the client.

REACHABILITY PROTOCOL

REAP [2] detects failures across the communicating path, that is, verifies reachability for the locator pair in use for each direction, and finds a new pair of locators when a failure occurs in any unidirectional path.

The approach followed for failure detection in REAP is to require Shim6 entities to care about the data path status only when they are sending data. In this case the path is considered to be valid if incoming traffic is received. Most protocols send data in both directions, and if there is only traffic in one direction, REAP sends keepalives in the opposite direction. So with REAP running, if there is outgoing traffic but no incoming traffic, there must be a failure. In particular, a Shim6 entity sending data considers that communication is proceeding successfully if it receives at least one packet from the remote host before its local Send Timer expires. When a packet is received from the remote host, the Send Timer is stopped and restarted the next time the local host sends a packet. Considering the situation in which two hosts, exchanging data at rates faster than the Send Timer expiration rate, and a failure in one of the communication paths, we can see that the host not receiving incoming traffic for a Send Timer period will infer that a communication problem has occurred.

When data traffic cannot be used to provide positive feedback about the validity of the paths to a sending entity (e.g., because communication is unidirectional), a REAP-specific Keepalive message is generated by the host receiving the data packets. Keepalive messages are issued by a Shim6 entity if a given time has elapsed since the reception of a packet without any data packet being sent. Consider now that a unidirectional communication is established between sender A and receiver B. When both unidirectional paths are valid, the data of A makes B send Keepalive messages that are received by A before the expiration of its Send Timer. On one hand, if the path from A to B fails, B will stop sending Keepalive messages, and the Send Timer will expire in A. On the other hand, if the path from B to A fails, B will continue sending Keepalive messages, but these messages will not arrive to A, resulting in the expiration of the Send Timer in A. Of course, if both directions fail, A also detects the communication problem.

Note that some time after both hosts stop sending data, REAP packet exchanges are also stopped.

When the Send Timer expires in a host, the host starts sending REAP Probe messages to test the unidirectional paths currently in use. If it does not receive a response to this validation, Probe messages with different combinations of source/destination locators are sent sequentially, with an exponential backoff to increase the time between successive probes. Although parallel probing would reduce the time to find a new working path, it would raise concern about signaling storms in case a failure affects to a large number of Shim6 hosts. When one of these Probe messages is received at the other end, the remote host includes this information in its own probes to inform the other end about the validity of that unidirectional

path. The process ends when two unidirectional paths have been discovered, and both hosts know the locators to use in each direction.

Security Model for the Shim6 Protocol

It is worth noting that the separation of identifiers and locators, and the use of a protocol to exchange this information can introduce new vulnerabilities to IPv6. In particular, the Shim6 protocol could be used by an attacker to launch *redirection attacks* [4] (i.e., attacks that create false identifier-to-locator mappings). The two possible types of redirection attacks are *hijacking attacks* and *flooding attacks*. In the next subsections we describe these types of attack, discuss the security level that should be provided for each, and present the corresponding protection measures provided by Shim6.

Hijacking Attacks

In a hijacking attack the Shim6 protocol can be used to induce a victim to associate one of the attacker's locators with a target ULID. Then, when the victim sends packets to the target ULID, it will be actually sending packets to the attacker. Through this attack, the attacker manages to steal the target's identity and hijack the communications between the target and the victim.

To determine the security level required against hijacking attacks, we can use the rule of thumb that Shim6 must not add new vulnerabilities to the ones currently present in the Internet. Hijacking attacks are feasible in today's Internet when the attacker is along the path between the communicating hosts. In other words, current communications are susceptible to man-in-the-middle attacks, and an attacker capable of intercepting packets can hijack a communication if no additional security measures such as IPsec or transport layer security (TLS) are adopted. Therefore, the prevention of man-in-the-middle attacks does not need to be a goal for the Shim6 protocol. A special type of hijacking attack is the so-called *time-shifted attack* [5], in which the attacker launches the attack from an on-path position and then leaves, but the effect of the attack remains after it has stopped. Current TCP/IP communications are not susceptible to time-shifted attacks. In the case of the Shim6 protocol, the attacker would remain on-path the time required to create a false ULID-to-locator mapping in the victim's host and then it could leave. However, the victim will preserve the false mapping long after the attacker has left. Because this is a new vulnerability introduced by the multihoming protocol, the security mechanisms of the protocol must prevent such time-shifted hijacking attacks.

To provide the protection discussed in the previous paragraph, the use of CGAs or HBAs for Shim6 is proposed.

CGAs [6] are regular IPv6 unicast addresses whose interface identifier (the 64 least significant bits of the IPv6 address) is built as a one-way hash of the CGA parameters structure, which contains the public key, the prefix of the address, and a 128-bit modifier, along with other parameters not relevant to our discussion.

The information used to build the CGA is conveyed in the `R1`, `I2`, and `Update Request` messages when the locators are exchanged. Each host uses the private key of its CGA to sign the locator set exchanged in messages `R1`, `I2`, or the `Update Request`. Then the responder to these messages validates the CGA of the remote host by regenerating it from its components, and validates the signature with the public key associated with the CGA. By means of this security chain check, the CGA provides proof of address ownership, and the locators can be added securely to the Shim6 context.

To evaluate the security provided by CGA, we estimate the difficulty for an attacker to hijack a communication. To do so, an attacker must be able to sign a locator in its link with a public key for which the CGA generation procedure results in the same CGA as the host to be impersonated. In this way it could start a communication with a third host using the CGA as ULID, and diverting the communication to a locator in which the attacker resides. The attacker may try to obtain the private key from the public key or a signature, but this is computationally hard, especially for long keys. Another approach would be to generate a key pair, and start an exhaustive search for a modifier until the resulting CGA hash is equal to the interface identifier of the CGA of the host to impersonate. Since there are 59 bits available for placing the result of the address hash after taking out the u and g bits that are preserved as defined in the IPv6 specification, and the three bits for a parameter used to provide additional security, the expected number of iterations is at least $O(2^{59})$. We believe that the resulting security is enough for protecting regular traffic that flows unprotected through the network from potential redirection attacks introduced by multihoming mechanisms [7].

HBAs [8] follow a different approach to provide security. An HBA set provides a mechanism to securely chain multiple addresses of a host by means of a hash, as depicted in Fig. 3. The result of the HBA generation process is as many different addresses as prefixes, with different interface identifiers. Each interface identifier is generated as the hash of the parameter data structure containing the sequence of prefixes assigned to the host. The ordering of the prefixes for the different addresses varies in order to provide privacy by obfuscating the binding among the addresses of a set. A host communicating with an address belonging to an HBA can evaluate with just one hash if another address belongs to the HBA set. This provides an inexpensive way to validate if a communication session or association can safely be diverted to the new address. As for CGA, the prefix list is exchanged in the `R1`, `I2`, and `Update Request` messages.

An attacker trying to impersonate a host identified with one address of the HBA must create a new HBA set with the following characteristics:

- The HBA address of the attacked host must be included, that is, its prefix set must appear in the multi-prefix extension, and the interface identifier computed for this prefix must be equal to the address of the attacked host.
- The prefix set of the HBA must include the

It is worth noting that the separation of identifiers and locators and the use of a protocol to exchange this information can introduce new vulnerabilities to IPv6. In particular, the Shim6 protocol could be used by an attacker to launch redirection attacks.

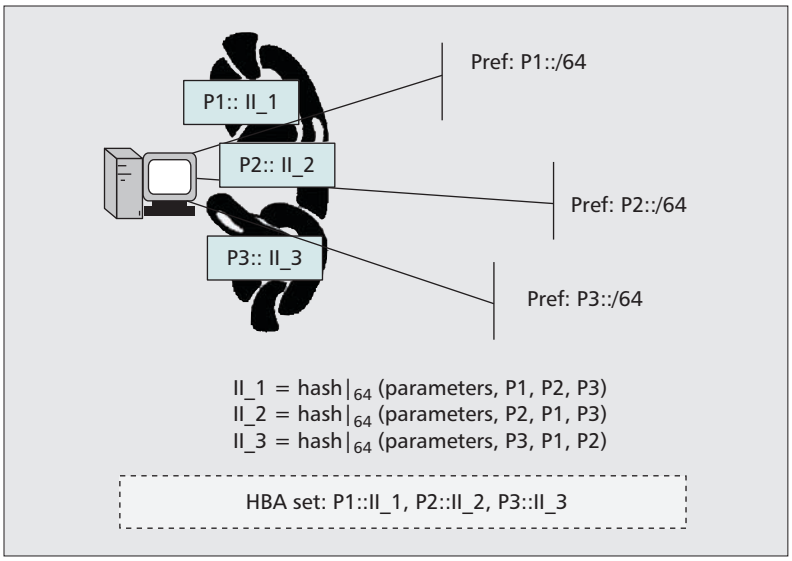

Figure 3. *HBA example.*

prefix of the attacker to be able to redirect the traffic to this address.

The attacker can vary the modifier of the CGA parameters structure until these conditions are fulfilled, but this requires a brute-force scan similar to the CGA case, with $O(2^{59})$ expected iterations.

For compatibility reasons, HBAs are defined as an extension of CGA, despite the evident semantic difference, by appending the list of prefixes of the node to the CGA parameters structure. The Shim6 protocol allows any combination of CGA and HBA addresses for a host. Hosts with HBA-only addresses are limited to use the addresses associated with the prefixes initially included in the prefix list, but CPU requirements for validating the address in remote hosts are lower than in the CGA case. CGA addresses, on the other hand, allow the dynamic addition of new locators by signing them with the private address associated with the CGA. Finally, it is possible to have a CGA/HBA address in which both a public key and a set of interface identifiers for different prefixes are bound.

Flooding Attacks

In a flooding attack the attacker starts a communication with a server, for instance, to download heavy streaming content. Then the attacker uses the Shim6 protocol to re-home the communication to a victim's locator, causing the server's flow to flood the victim.

Flooding attacks are possible in the current Internet only from an on-path location. However, an attacker could use Shim6 to establish a communication to a server, and then require a change in the server's mapping to the locator of a victim.

Shim6 provides protection against this attack by means of the inclusion of context tags in the REAP `Probe` message checks alternative paths to a destination: a node can detect that the host to which the communication is being diverted is not the same one that established the Shim6 communication because it does not return the same context tag.

Protocol Walkthrough

In Fig. 4 we show an example of the behavior of two Shim6 hosts communicating in a common scenario. Host X has two prefixes assigned, PX_1 and PX_2, and has configured one key pair to build a CGA with PX_1 (CGA_1) and another key pair to build a CGA with PX_2 (CGA_2). Host Y is in a link with three prefixes, PY_1, PY_2, and PY_3, and has generated one address per prefix according to the HBA specification, resulting in address set HBA_1, HBA_2, and HBA_3.

X initiates a communication with Y. Typically, an application on X issues a DNS request for a name associated with Y, obtaining in the request some subset of the addresses assigned to Y. The regular address selection process for IPv6 is used by X to select one of the addresses of Y (e.g., HBA_1) as the destination address for the outgoing packets, and one of its own addresses (CGA_1) as the source address. In both cases appropriate configuration of each host and/or the local network must ensure that packets with a given source address are forwarded to the provider from which the address was delegated so that the packets comply with potential ingress filters. These addresses selected at the beginning of the communication will be used as endpoint identifiers for transport and application layers when required (i.e., as ULIDs). Note that the procedure of establishing the communication happens before any Shim6 protocol exchange.

Consider that Y is a content server that sends traffic at high rates, while X sends packets sparsely (we assume that the sending period is larger than the REAP timers). We suppose that some time later the network layer of X decides through a heuristic to request higher quality in terms of reliability, so it initiates the Shim6 exchange (Fig. 4). X has signed the CGA_2 address with the private key associated with CGA_1, so Y validates CGA_1 and the signature of the locator, and associates CGA_2 as an alternative locator for the context. On the other hand, Y has transmitted to X all the parameters required for generating the HBA, and transmitted the alternative locators. X checks the validity of transmitted locators using the CGA multi-prefix extension, and associates the locators with the Shim6 session.

Host X, as an infrequent sender, generates periodic `Keepalive` messages to provide feedback to Y about the validity of the paths.

Some time later a failure in the network occurs, preventing communication through the provider delegating the PX_1 prefix to host X. Then neither can possible pending `Keepalive` messages arrive at Y, nor is data traffic received by X. As a result, the `Send Timer` at Y expires. A `Probe` message is sent to X using the current locators to check the validity of the path in use. Since Y receives no answer, it initiates exploration for an alternative locator pair, sending a `Probe` with different source and destination addresses. As a result, responses could be received for any pair not containing the PX_1 prefix. Y checks, for example, $<CGA_2, HBA_1>$, and the `Probe` is received by X. Then X checks for another locator pair for the unidirectional path from X to Y, being also $<CGA_2, HBA_1>$ in this case, and includes in the `Probe` information about the working condition

of locators <CGA_2, HBA_1> in the *Y* to *X* direction. When *Y* receives this information, it updates the current locator path for sending data in the Shim6 context, and confirms the locators for the *X* to *Y* unidirectional path. From now on, *X* will then receive data packets with the new locators, and the communication is preserved.

CONCLUSIONS

In this article we have presented the Shim6 architecture for IPv6 multihoming, which protects communications from failures in a scalable way. The standardization of the Shim6 mechanism has been completed in the IETF, resulting in the Shim6 protocol [1], the HBA address format [8], and REAP [2]. In addition, several implementations have been reported.[1]

As for any new technology, the challenge that Shim6 faces at this stage is wide adoption. The incentives and hazards for its adoption have been analyzed in depth in [9]. From our perspective, we believe that the resulting architecture is easy to adopt. Shim6 does not require changes in the applications since it can be implemented as an operating system service, although Shim6-aware applications could benefit from improved control through a Shim6 application programming interface [10]. Additionally, end site management is simple, since none of the presented mechanisms require manual configuration, allowing poorly managed sites to easily deploy the proposed solution. Shim6 hosts can easily coexist with non-Shim6-aware nodes, since for these nodes CGA or HBA addresses, which are valid locators, appear as regular IPv6 addresses. This coexistence is not possible for other solutions such as HIP [3], in which a strict separation between locators and identifiers is required. Fault tolerance support is directly implemented in the end hosts and works without requiring user configuration. All these features enable easy adoption of the presented solution in SOHO environments that lack network administration.

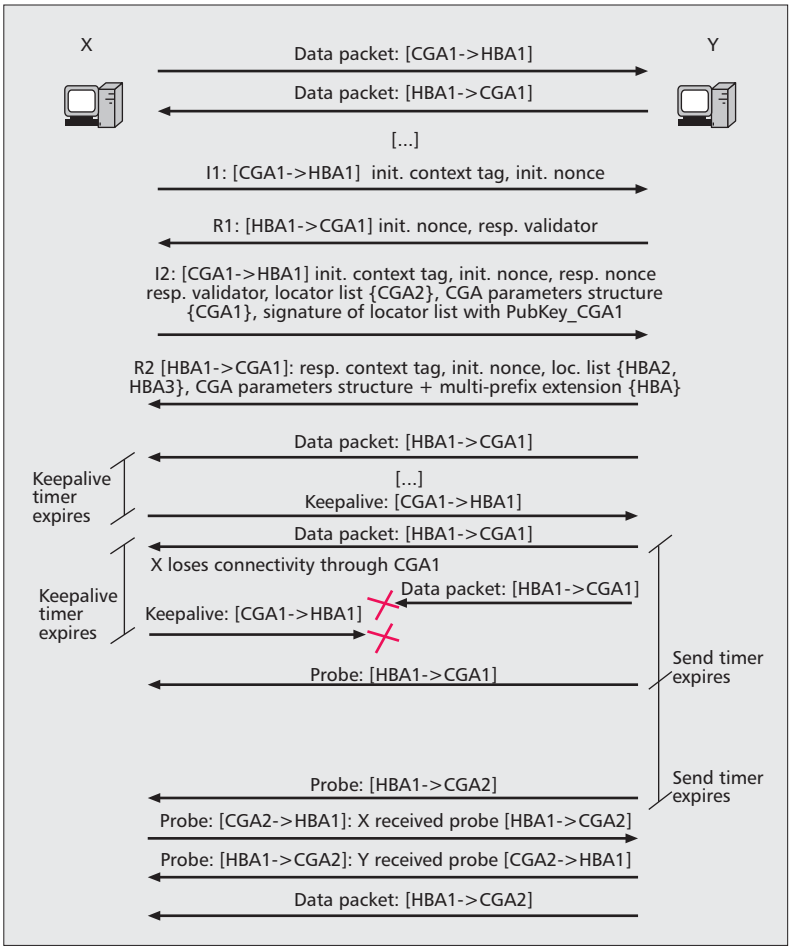

Figure 4. *Example of data exchange between two Shim6 hosts.*

ACKNOWLEDGMENTS

The work of Alberto García-Martínez is supported by the T2C2 project (TIN2008-06739-C04-01), funded by the Spanish Ministerio de Ciencia e Innovació. The work of Marcelo Bagnulo and Iljitsch van Beijnum is supported by Trilogy (ICT-216372, http://www.trilogy-project.org), a research project partially funded by the European Community.

REFERENCES

[1] E. Nordmark and M. Bagnulo, "Shim6: Level 3 Multihoming Shim Protocol for IPv6," IETF RFC 5533, June 2009.
[2] J. Arkko and I. van Beijnum, "Failure Detection and Locator Pair Exploration Protocol for IPv6 Multihoming," IETF RFC 5534, June 2009.
[3] R. Moskowitz *et al.*, "Host Identity Protocol," IETF RFC 5201, Apr. 2008.
[4] E. Nordmark and T. Li, "Threats Relating to IPv6 Multihoming Solutions," IETF RFC 4218, Oct. 2005.
[5] P. Nikander *et al.*, "Mobile IP Version 6 Route Optimization Security Design Background," IETF RFC 4225, Dec. 2005.
[6] T. Aura, "Cryptographically Generated Addresses (CGA)," IETF RFC 3972, Mar. 2005.
[7] M. Bagnulo, A. García-Martínez, and A. Azcorra, "Efficient Security for IPv6 Multihoming," *ACM Comp. Commun. Rev.*, vol. 35, no. 2, Apr. 2005, pp. 61–68.
[8] M. Bagnulo, "Hash-Based Addresses (HBA)," IETF RFC 5535, June 2009.
[9] R. Clayton, "Internet Multi-Homing Problems: Explanations from Economics," *8th Annual Wksp. Economics Info. Security*, London, June 2009.
[10] S. Sugimoto, Ed., "Socket Application Program Interface (API) for Multihoming Shim," draft-ietf-shim6-multihome-shim-api-11, work in progress, Dec. 2009.

BIOGRAPHIES

ALBERTO GARCIA-MARTINEZ (alberto@it.uc3m.es) received a telecommunication engineering degree in 1995 and a Ph.D. in telecommunications in 1999, both from the Polytechnic University of Madrid, Spain. In 1998 he joined the University Carlos III of Madrid (UC3M), where he has been an associate professor since 2001. He has published several papers in technical journals, magazines, and conferences. His main interest areas are IPv6, layer 2 routing, and interdomain routing.

MARCELO BAGNULO (marcelo@it.uc3m.es) received an electrical engineering degree in 1999 from the University of Uruguay, and a Ph.D. in telecommunications in 2005 from the UC3M. In 2000 he joined UC3M, where he has been an associate professor since 2006. He has published several papers in technical journals, magazines, and conferences. His main interest areas are IPv6 and interdomain routing.

ILJITSCH VAN BEIJNUM (iljitsch.vanbeijnum@imdea.org) received his Bachelor of Information and Communication Technology degree from the Haagse Hogeschool at The Hague, The Netherlands in 2005, and his Master of Telematics Engineering degree from UC3M in 2008. He has worked in the Internet service provider business since 1995, and has written books about BGP and IPv6 and many articles in the trade press. He currently works for IMDEA Networks as a research assistant, pursuing a Ph.D. degree at UC3M.

[1] *Information about implementations of Shim6 can be found at http://www.shim6.org.*