Network Working Group S. Barre
Internet-Draft O. Bonaventure
Expires: March 28, 2010 UCLouvain, Belgium
September 24, 2009

Shim6 Implementation Report : LinShim6
draft-barre-shim6-impl-03

Status of this Memo

Copyright Notice

Abstract

LinShim6 is an implementation of the Shim6 and REAP protocols, on the Linux platform. This draft provides a description of the architecture and describes the current state of our implementation. The level of support of each protocol feature is detailed. Protocol conformance is evaluated against the main drafts.

Table of Contents

1. Introduction

The Shim6 protocol [RFC5533] has been designed to add multihoming capabilities to IPv6, while avoiding the drawbacks of current IPv4 multihoming practice (prefix announcements in BGP), and giving more control to the end host (through locator selection).

Together with the Shim6 protocol, the working group has designed a failure detection mechanism, called REAP [RFC5534], that allows hosts to detect and recover from failures, thanks to a combination of traffic monitoring and active probing.

Implementing such new protocols is crucial to allow tracking errors or weaknesses in the overall design, as well as evaluating protocol behaviour in the real world. We developped an implementation of Shim6 and REAP, available from <http://inl.info.ucl.ac.be/LinShim6>. LinShim6 has been used to evaluate the performance of REAP path exploration [BARRE07].

This draft is aimed at describing the challenges of a proper integration of Shim6 in a protocol stack while preserving its efficiency. LinShim6 supports the base Shim6 protocol (negotiation and address rewriting) as well as failure detection and recovery (REAP). To our knowledge LinShim6 is also the first publicly available implementation that supports both the HBA and CGA mechanisms for securing the locator set exchange (the CGA/HBA code is derived mostly from the DoCoMo SEND project [1]).

In this draft, we present a detailed report of the supported parts of the protocol, in terms of the terminology defined in section 2 of [RFC5533]. Some non critical features for the current application of LinShim6 have not been implemented yet. They will be added as soon as a need for them arises. For instance, the Forked Instance Identifier is only useful if a socket API is implemented (such as the API defined in [I-D.ietf-shim6-multihome-shim-api]). The Locator Preference Option may only be used if the corresponding tuning capability is provided, either by the user or by an automated technique.

Other features will be supported in a future version of the implementation. These are detailed in Section 4.

This draft describes version 0.9 of LinShim6.

2. General architecture

The LinShim6 implementation is composed of two parts. First, a kernel patch adds support for shim6 negotiation trigger, address rewriting and failure detection. Second, a daemon is responsible for the management of the Shim6 control plane (negotiation, path exploration). The kernel communicates with the user space daemon through the Netlink interface [RFC3549].

Hereafter we briefly describe the kernel and user level part of LinShim6. A more extensive description can be found in [BARRE07b] or [BARRE08].

2.1. Kernel patch

The default negotiation trigger makes use of the NF_IP6_LOCAL_IN and NF_IP6_LOCAL_OUT netfilter hooks to listen to the packets travelling through the networking stack. A Shim6 negotiation is triggered by default when either 2 KB of data have been seen for a given address pair or the flow exists for one minute. Those values have been chosen through observation of netflow traces, showing that more than 80% of the observed traffic last less than 1 minute, and also 80% is less than 2 KB in size. This default heuristic thus appeared as a reasonable discriminator to avoid starting a Shim6 negotiation when it is not needed. In some cases (e.g. heavily loaded servers), one may want to never trigger a Shim6 context establishment, except if it is explicitly requested by the peer. This is possible by simply not loading the heuristic module. One can also define its own heuristic, by designing a customized module.

Address rewriting is implemented as an extension to the XFRM framework, originally designed for IPsec [KANDA04]. The XFRM framework allows for dynamically adding a new sublayer in the Networking stack for some flows, according to a policy. Examples of already defined sublayers are the AH sublayer (Authentication Header) or the ESP sublayer (Encapsulating Security Payload). Similarly, we define a new sublayer for Shim6. The policies responsible for directing packets to this new module are communicated from the daemon to the kernel through Netlink, when a change in the locators is needed or a new Shim6 context is created. For outgoing packets, the policy takes the form of a matching rule with the ULIDs (Upper Layer IDentifiers, defined in [RFC5533]). For incoming packets that do not have the Shim6 extension header, the same kind of matching rule is used. We also defined a matching rule based on the context tag, in order to be able to demultiplex tagged incoming packets.

Failure detection is performed inside the kernel for efficiency reasons: a timer must be started or stopped for each incoming or outgoing packet. We maintain REAP failure detection timers inside the XFRM states, so that the daemon is notified (through Netlink) when a keepalive must be sent or an exploration is to be started.

2.2. LinShim6 daemon

The daemon continuously listens to three types of events. First, Shim6 and REAP control messages are received through a raw socket. Second, Netlink messages provide information from the kernel, for example whether a context must be created, a keepalive must be sent or an exploration must be started. Finally, messages can be received through a pipe where the other threads may write commands. Four threads are currently defined:

- o Main thread: Maintains all the critical states.
- o XFRM: Listens to the XFRM events from the kernel. Currently only the state expiry event is used. It is generated when a kernel context has seen no traffic during more than 10 minutes. The result is that the daemon deletes the corresponding association.
- o Timer: It maintains a timer queue and wakes up when any timer expires. On expiration of any timer, it requests the main thread to run the corresponding handling function.
- o Information server: A simple telnet server that provides a convenient interface to the daemon. The server can be accessed with the shim6c tool.

2.2.1. Random number generation

We generate random numbers based on the Linux random() function, with a seed taken from /dev/random when the daemon starts, and every 1000 generation.

2.2.2. HBA/CGA support

The user is able to set HBA and CGA parameters thanks to a configuration file. A tool (cgatool), derived from the DoCoMo SEND project, allows for manual generation of CGA keys, CGA addresses and HBA addresses. Four types of addresses can coexist in an end-system: unsecured, HBA, CGA and CGA-compatible HBA. It is up to the applications to decide which address is used as ULID for a given communication. If the application chooses the unspecified source address, then the kernel applies RFC3484[RFC3484] rules to pick a suitable source address from the available set. When performing the locator set exchange, LinShim6 decides what locators to use in the local locator set based on the ULID type:

Internet-Draft Shim6 Implementation Report : LinShim6 September 2009

- o Unsecured address: the local ULID is neither a CGA nor an HBA. LinShim6 decides that the locator set is made of only the ULID, because it would be impossible for the peer to check the validity of the other locators.
- o HBA address: the local ULID is an HBA (not CGA-compatible): LinShim6 sends all the addresses that are in the same HBA set and are currently available in the system. For example if an HBA set is configured to gather four prefixes, but the host only receives Router Advertisements for two of them, only the corresponding two addresses are announced to the peer. If later other addresses become reachable, they are announced through an Update Request.
- o CGA address: since a signature is used to authenticate a locator set, any locator can be put in the set. LinShim6 behaviour is then to advertise all available locators in the system.
- o CGA-compatible HBA address: LinShim6 also sends all available locators to the peer. The only difference with pure CGA addresses is that the subset of addresses belonging to the same HBA set as the ULID are verified with HBA rather than included in the signature, thus leading to a faster verification process.

2.3. Locator updates

During the lifetime of a Shim6 context, locators may appear or disappear. If a new locator becomes available in the system, all peers are updated (except if the new address cannot be added to some of the contexts, according to the rules described in Section 2.2.2). As required by [RFC5533], the new locator starts being actually part of a Shim6 context only when the new locator set has been aknowledged by the peer.

On the other hand, when a locator disappears, it is immediately removed from all contexts and a locator update is sent to the peer. It does not make sense to wait for the acknowledgement in that case, since the locator is not reachable anymore. Moreover, if the removed locator is current for any context (that is, actually used for sending packets), a REAP path exploration is triggered.

2.4. Context removal

As mentioned in the previous section, a context is removed upon reception of an XFRM event from the kernel, indicating that no traffic has been seen for that context during at least 10 minutes. The daemon then cleans up all data related to the expired context, both in the daemon and in the kernel. Shim6 kernel state is also cleaned everytime the daemon is started to avoid inconsistency.

In the future, we will also check if no opened socked is using the context before removing it. This will avoid the current possibility that a context gets stalled, if it remains idle during more than 10 minutes and then tries to send data again.

3. RFC 2119 evaluation

In this section we detail the conformance of the LinShim6 implementation in terms of the RFC2119 [RFC2119] terminology. Additionally, we define hereafter the keywords that are used to describe the level of support for the different features.

- o YES: The feature is fully supported.

- o FEATURE NOT SUPPORTED: if a MUST is followed by "FEATURE NOT SUPPORTED", this means that the MUST makes sense only if the feature exists. That is, the implementation is still compliant but does not implement the particular feature. Currently unsupported features are:

  - * R1bis: this message is defined to allow the recovery of a context, when one endpoint has dropped the context while the other endpoint is still using it. When [RFC5533] specifies to send a R1bis message, we currently ignore the message supposed to trigger the sending of the R1bis.

  - * Error messages: used to inform the peer about what went wrong.

  - * IPsec: the design of LinShim6 is based on the XFRM architecture in the kernel. The same architecture is used by IPsec, thus a small adaptation (if any) of LinShim6 should allow it to work well together with IPsec. However, we have not yet tested such an interaction.

  - * FII (Forked Instance Identifier): the FII is defined in [RFC5533] as a way to fork Shim6 contexts, so that several contexts may share the same ULID pair, and are distinguished thanks to an integer called the FII. This has interest only if a socket API is implemented, so that applications may choose a context rather than another to send packets (which allows selecting a different set of locators). There is no short term plan to support this.

  - * ULID pair option: it is defined to allow performing context negotiation with a locator pair that differs from the ULID pair. This may be useful for example if non routable ULIDs are used. There is no short term plan to support this, because non-routable ULIDs are not (yet ?) deployed in the current Internet.

- o NO: Unsupported optional features are simply followed by NO.

- o CONFIGURABLE: The feature is supported, but requires manual configuration from the user for correct behaviour.

- o PARTIAL SUPPORT: The feature is partially supported, that is, the requirement in verified in some cases, but not all. In that case we point to a section that gives more details on the behaviour.

3.1. Checks common to all control messages

A host MUST silently discard any received control message that does not statisfy all of the following validity checks:

- o The Shim header length field is verified against the length of the IPv6 packet to make sure that the shim message doesn't claim to end past the end of the IPv6 packet: YES (Checked in the kernel)

- o the checksum is correct: YES (Checked in the kernel)

- o Neither the IPv6 destination field nor the IPv6 source field is a multicast address nor the unspecified address: YES (Checked in the kernel)

3.2. I1 Message

- o The Reserved1 field MUST be ignored on receipt: YES

- o The R field MUST be ignored on receipt: YES

- o When another instance of an existent context with the same ULID pair is being created, a Forked Instance Identifier option MUST be included to distinguish this new instance from the existent one: FEATURE NOT SUPPORTED (FII)

- o The I1 message MUST include the ULID pair: YES (always in the IPv6 header)

- o A host MUST silently discard any received I1 message that does not statisfy all of the following validity checks:

  * Hdr Ext Len field at least 1: YES

  * If the ULID pair option is present, the host verifies that the locator of the Initiator is included in Ls(peer): FEATURE NO SUPPORTED (ULID pair option)

3.3. R1 Message

- o The Reserved1 field MUST be ignored on receipt: YES

- o The Reserved2 field MUST be ignored on receipt: YES

- o The Responder Validator Option MUST be included: YES

- o A host MUST silently discard any received R1 message that does not statisfy all of the following validity checks:

  * Hdr Ext Len field at least 1: YES

  * the host looks for an existing context which matches the Initiator Nonce and where the locators are contained in Ls(peer) and Ls(local), respectively. If no such context is found, then the R1 message is silently discarded: YES

  * If the context found using the above rules is not in I1-SENT state, the R1 message is silently discarded: YES

3.4. I2 Message

- o The Reserved1 field MUST be ignored on receipt: YES

- o The R field MUST be ignored on receipt: YES

- o The Reserved2 field MUST be ignored on receipt: YES

- o The Responder Validator Option MUST be included: YES

- o The Responder Validator Option MUST be generated copying the Responder Validator option received in the R1 message: YES

- o When the IPv6 source and destination addresses in the IPv6 header do not match the ULID pair, the ULID-pair option MUST be included: FEATURE NOT SUPPORTED (ULID pair option)

- o When another instance of an existent context with the same ULID pair is being created, a Forked Instance Identifier option MUST be included to distinguish this new instance from the existent one: FEATURE NOT SUPPORTED (FII)

- o When the Locator List Option is sent, the necessary HBA/CGA information for verifying the locator list MUST also be included: YES o The CGA PDS option MUST be included when the locator list is included: YES.

o The CGA Signature option MUST be included when some of the locators in the list use CGA (and not HBA) for verification: YES

o If the initiator does not receive an R2 message after I2_TIMEOUT time after sending an I2 message it MAY retransmit the I2 message, using binary exponential backoff and randomized timers: YES

o In the case that the initiator decides not to retransmit I2 messages or in the case that the initiator still does not recieve an R2 message after retransmitting I2 messages I2_RETRIES_MAX times, the initiator SHOULD fall back to retransmitting the I1 message: YES

o A host MUST silently discard any received I2 message that does not statisfy all of the following validity checks:

* Hdr Ext Len field at least 2: YES

* The responder nonce is a recent one. Nonces that are no older than VALIDATOR_MIN_LIFETIME SHOULD be considered recent: YES

* the Responder Validator option matches the validator the host would have computed for the ULID, locators, responder nonce, initiator nonce and FII: YES

* If a CGA Parameter Data Structure (PDS) is included in the message, then the host MUST verify if the actual PDS contained in the message corresponds to the ULID(peer): YES

* If the state is I1-SENT, then the host verifies if the source locator is included in Ls(peer) or, it is included in the Locator List contained in the I2 message and the HBA/CGA verification for this specific locator is successful: YES

o If a host is in I1-SENT state, receives an I2 message and all the above checks are successful, then it MUST send a R2 message back: YES

3.5. R2 Message

o The Reserved1 field MUST be ignored on receipt: YES

o The R field MUST be ignored on receipt: YES

o When the Locator List Option is sent, the necessary HBA/CGA information for verifying the locator list MUST also be included: YES

o Before an R2 message is sent, the host MUST look for a possible context confusion: YES (this is verified at I2/R2 reception)

o A host MUST silently discard any received R2 message that does not statisfy all of the following validity checks:

* Hdr Ext Len field at least 1: YES

* the host looks for an existing context which matches the Initiator Nonce and where the locators are contained in Ls(peer) and Ls(local), respectively. If no such context is found, then the R2 message is silently dropped: YES

* If state is I1-SENT, I2-SENT or I2BIS-SENT and a CGA Parameter Data Structure (PDS) is included in the message, then the host MUST verify if the actual PDS contained in the message corresponds to the ULID(peer): YES

o Before the host completes the R2 processing it MUST look for a possible context confusion: YES

3.6. R1bis, I2bis

Those messages are not supported yet. They are ignored on receipt.

3.7. Update Request(UR)/Acknowledgement(UA) messages

o The Reserved1 field MUST be ignored on receipt: YES

o The R field MUST be ignored on receipt: YES

o A host MUST silently discard any received UR/UA message that does not statisfy all of the following validity checks:

* Hdr Ext Len field at least 1: YES

* the host looks for an existing context whose CT(local) matches the context tag. If no such context is found, it sends a R1bis message: FEATURE NOT SUPPORTED (R1bis)

* Since context tags can be reused, the host MUST verify that the IPv6 source address field is part of Ls(peer) and that the IPv6 destination address field is part of Ls(local). In this case the host MUST send a R1bis message, and otherwise ignore the

UR/UA message: FEATURE NOT SUPPORTED (R1bis)

* UR only: If a CGA Parameter Data Structure (PDS) is included in the message, then the host MUST verify if the actual PDS contained in the message corresponds to the ULID(peer): YES

3.8. Keepalive and Probe Messages

o The Type field must be 66 for a keepalive, 67 for a probe: YES

o The Reserved1 and Reserved2 fields MUST be ignored on receipt: YES

o The R bit MUST be ignored on receipt: YES

o A keepalive MAY contain options: NO (no option is currently defined)

o The first set of sent probe fields of a probe message pertains to the currently sent probe message and MUST be present: YES

o This value SHOULD be generated using a random number generator that is known to have good randomness properties as outlined in RFC 4086: YES

o If the host is using a non-default Send Timeout value, it SHOULD communicate this value as a Keepalive Timeout value to the peer: YES

o When sending a Probe message, the State field MUST be set to a value that matches the conceptual state of the sender after sending the Probe: YES

o The Reserved field MUST be ignored on receipt: YES

3.9. Keepalive Timeout Option

o This option MAY be sent in the I2, I2bis, R2, or UPDATE messages: YES (The option is sent in I2,R2 and UPDATE messages. I2bis is not yet supported)

o The option SHOULD only need to be sent once in a given shim6 association.: YES (However, if Tsend is set manually through the shim6c tool, all contexts are updated to reflect the new Tsend value. This implies the sending of an UPDATE message, with the keepalive timeout option).

o If a host receives this option it SHOULD update its Keepalive Timeout value for the correspondent: YES. (Note that LinShim6

handles Tsend as a global value for the system, and Tka as a per-context value).

o The Type field identifies the option and MUST be set to 10 (Keepalive Timeout): YES.

3.10. Error messages

FEATURE NOT SUPPORTED

3.11. Message Options

o The length field MUST NOT include the padding: YES

o Any added padding bytes MUST be zeroed by the sender: YES

o The values of the padding bytes SHOULD NOT be checked by the receiver: YES

o If C=1 and the option is not recognized by the receiver, then the host SHOULD send back a Shim6 error message with Error Code=1, with the Pointer referencing the first octet in the Option Type field: FEATURE NOT SUPPORTED (error messages)

o If C=1 and the option is not recognized by the receiver, then the rest of the message MUST NOT be processed: YES

o Locator Preferences: Any element definition of length greater than 3 MUST be defined so that the first three bytes agree the definition given in the draft: YES (we do not define longer element fields)

o The Reserved2 field of the ULID pair option MUST be ignored on receipt: FEATURE NOT SUPPORTED (ULID pair option)

o If the verification method in the Locator List option is not supported by the host, or if the verification method is not consistent with the CGA Parameter Data Structure, then the host MUST ignore the Locator List and the message in which it is contained: YES

o If the verification method in the Locator List option is not supported by the host, or if the verification method is not consistent with the CGA Parameter Data Structure, then the host SHOULD generate a Shim6 Error message with Error Code=2, with the Pointer referencing the octet in the Verification method that was found inconsistent: FEATURE NOT SUPPORTED (Error messages)

3.12. Payload data

- The insertion of the Shim6 extension header in payload packets MUST NOT cause any recalculation of the ULP checksums: YES
- When receiving a packet with a context tag that does not match any context, the receiver SHOULD generate a R1bis message: FEATURE NOT SUPPORTED (R1bis)
- If payload data is received with a context tag that matches with a context in state I2-SENT or I2BIS-SENT, the host resp. sends back a I2 or I2bis and proceeds to process the message: NO (the message is processed only for an ESTABLISHED state)

3.13. General requirements of the Shim6 draft

- The I1, I2 and I2bis messages MUST contain the ULID pair, either in the IPv6 header or in a ULID pair option: YES (During negotiation the locators are always the identifiers, thus the ULID pair option is not needed.)
- The context tag MUST be unique for each context: YES
- At least 30 bits of the context tag MUST be populated by random or pseudo-random bits: YES (all 47 bits are pseudo-random)
- The host SHOULD randomly cycle through the unstructured tag name space: YES
- The HBA/CGA verification SHOULD be performed by the host before the host acknowledges the new locator, by sending an Update Acknowledgement message, or an R2 message: YES
- Before a host can use a locator (different from the ULID) as the destination locator it MUST perform the HBA/CGA verification if this was not performed before upon the reception of the locator set: YES (Checked by the daemon upon reception)
- Before a host can use a locator (different from the ULID) as the destination locator, it MUST verify that the ULID is indeed present at that locator. This verification is performed by doing a return- routability test as part of the Probe sub-protocol: YES
- I2, I2bis and R2 messages MUST include a sufficiently large set of locators in a Locator List option that the peer can determine whether or not two contexts have the same host as the peer by comparing if there is any common locators in Ls(peer): CONFIGURABLE (see Section 2.2.2)

- In case of context confusion detection ([RFC5533]), the old context which used the context tag MUST be removed: YES
- An implementation MAY re-create a context to replace the one that was removed because of confusion detection: NO (it is not automatically re-created, but it can be negotiated again if the heuristic triggers a context establishement or the peer sends a new I1).
- It is RECOMMENDED that hosts do not tear down the context when they know that there is some upper layer protocol that might use the context: PARTIAL SUPPORT (see Section 2.4)
- The minimum acceptable key length for public keys used in the generation of CGAs SHOULD be 1024 bits: YES
- in case that IPsec is implemented as Bump-In-The-Wire (BITW), either the shim MUST be disabled, or the shim MUST also be implemented as Bump-In-The-Wire, in order to satisfy the requirement that IPsec is layered above the shim: CONFIGURABLE (disable LinShim6 to use a BITW IPsec device)
- If a shim6 node has some protected and some unprotected interfaces for the purposes of IPsec, then it MUST treat the locator sets for the protected and unprotected interfaces as separate locator sets and not intermix them: FEATURE NOT SUPPORTED (IPsec).

3.14. General requirements of the REAP draft

- Available addresses are discovered and monitored through mechanisms outside the scope of SHIM6.SHIM6 implementations MUST be able to employ information provided by IPv6 Neighbor Discovery, Address Autoconfiguration, and DHCP (when DHCP is implemented). This information includes the availability of a new address and status changes of existing addresses (such as when an address becomes invalid): PARTIAL SUPPORT (Address discovery is performed using all mechanisms available in the kernel, but not monitored later)
- Locally operational addresses are discovered and monitored through mechanisms outside the SHIM6 protocol.SHIM6 implementations MUST be able to employ information provided from Neighbor Unreachability Detection: NO
- Locally operational addresses are discovered and monitored through mechanisms outside the SHIM6 protocol. Implementations MAY also employ additional, link layer specific mechanisms: NO

o SHIM6 implementations MUST support the discovery of operational address pairs through the use of explicit rechability tests and Forced Bidirectional Communication (FBD), described later in this specification: YES

o In addition, implementations MAY employ the following additional mechanisms:

  * Positive feedback from upper layer protocols: NO

  * Negative feedback from upper layer protocols: NO

  * ICMP error messages: NO

o After the reception of a data packet from the peer, REAP keepalive packets SHOULD continue to be sent at the Keepalive Interval until either a data packet in the SHIM6 context has been sent to the peer or the Keepalive Timeout expires: YES

o Upon changing to a new address pair, the network path traversed most likely has changed, thus the ULP SHOULD be informed: YES (A NETEVENT_PATH_UPDATE notification is generated whenever a new address pair is used. Any entity in the kernel can listen to that notification and act accordingly. Currently only the TCP layer listens to the notification, and reacts by resetting its Retransmission Timeout)

o Out of the set of possible candidate address pairs, nodes SHOULD attempt to test through all of them until an operational pair is found, and retrying the process as is necessary: YES

o All nodes MUST perform the exploration process sequentially and with exponential back-off: YES

o The externally observable behaviour of an implementation MUST conform to the REAP state machine: YES

o Unprotected indications from other parts of the protocol stack SHOULD NOT be taken as a proof of connectivity problems: YES

4. Protocol conformance by feature

In the following list we make a division of the Shim6 specification into several features, in order to easily identify which of them are supported and which are not.

o Context forking: No (Only useful if an API exists)

o Context recovery: Not yet

o Locator preferences option: Not yet

o Locator list updates: YES

o Cryptographically Generated Addresses: YES

o Hash Based Addresses: YES

o Failure detection and recovery: YES

o Context confusion detection ([RFC5533] sec. 7.6): YES

o Handling of ICMP error messages: Not yet

o Keepalive Timeout Option: YES

5. Conclusion and further work

This draft describes the current state of the LinShim6 implementation, version 0.9. It uses a heuristic to decide whether to trigger a Shim6 negotiation, and it is able to monitor the state of the communication thanks to the REAP state machine. It has been shown to successfully support the switch to an alternative locator pair, and it is the first known Shim6 implementation that supports HBA and CGA. LinShim6 is still under development. We aim at finally providing the complete set of features. In the near future we will work on context recovery and error messages. Other missing features seem to have a lower priority and are left for later.

We have established an exhaustive listing of supported and unsupported elements of the protocols, which appears as making much easier to verify the level of support and security of the protocol.

6. Acknowledgments

Sebastien Barre is funded by Trilogy, a research project (ICT-216372) supported by the European Community under its Seventh Framework Programme. The views expressed here are those of the author(s) only. The European Commission is not liable for any use that may be made of the information in this document.

Part of this work has been supported by a grant from FRIA (Fonds pour la Formation a la Recherche dans l'Industrie et dans l'Agriculture, rue d'Egmont 5 - 1000 Bruxelles, Belgium).

John Ronan reviewed this document and provided comments. He also spent many hours testing the code in many different scenarios, giving valuable feedback and helping in several tricky bug fixes. His help has been very invaluable to improve LinShim6 overall stability.

Matthijs Mekking has written a wireshark patch for Shim6, that has been very helpful in testing the implementation, and also used LinShim6 himself and provided feedback.

The CGA/HBA support, cgad and cgatool benefitted from much code from the DoCoMo SEND implementaion, the clarity of the code made extension and adaptation for LinShim6 very effective.

Francis Dupont has written the very first known implementation of HBA. Although we have written a second one based on DoCoMo SEND project, Francis Dupont's work, and especially his test suite, has been used to validate our HBA module.

Other people has helped getting things better by comments, bug reports, or discussions: Lu Junxiu, Martin Kulas, Aaron Kunde, Sazzadur Rahman, Iljitsch van Beijnum, Marcelo Bagnulo, James Swan, Shinta Sugimoto, Masahide Nakamura, Sebastian Meier, Amine Dhraief, the INL team.

7. References

[I-D.ietf-shim6-multihome-shim-api]
Komu, M., Bagnulo, M., Slavov, K., and S. Sugimoto, "Socket Application Program Interface (API) for Multihoming Shim", draft-ietf-shim6-multihome-shim-api-09 (work in progress), Jul 2009.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, Mar 1997.

[RFC3484] Draves, R., "Default Address Selection for Internet Protocol version 6 (IPv6)", RFC 3484, Feb 2003.

[RFC3549] Salim, J., Khosravi, H., Kleen, A., and A. Kuznetsov, "Linux Netlink as an IP Services Protocol", RFC 3549, Jul 2003.

[RFC5533] Nordmark, E. and M. Bagnulo, "Shim6: Level 3 Multihoming Shim Protocol for IPv6", RFC 5533, Jun 2009.

[RFC5534] Arkko, J. and I. van Beijnum, "Failure Detection and Locator Pair Exploration Protocol for IPv6 Multihoming", RFC 5534, Jun 2009.

[KANDA04] Kanda, M., Miyazawa, K., and H. Esaki, "USAGI IPv6 IPsec development for Linux", in International Symposium on Applications and the Internet, pp. 159-163, Jan 2004.

[BARRE07] Barre, S. and O. Bonaventure, "Improved Path Exploration in shim6-based Multihoming", in SIGCOMM 2007 Workshop "IPv6 and the Future of the Internet", Kyoto, Japan, Aug 2007.

[BARRE07b]
Barre, S. and O. Bonaventure, "Implementing SHIM6 using the Linux XFRM framework", in Routing In Next Generation workshop, Madrid, Spain, Dec 2007.

[BARRE08] Barre, S., "LinShim6 - Implementation of the Shim6 protocol", Feb 2008, <http://inl.info.ucl.ac.be/publications/linshim6-implementation-shim6-protocol>.

[1] <http://www.docomolabs-usa.com/lab_opensource.html>

Authors' Addresses

Sebastien Barre
Universite catholique de Louvain
Place Ste Barbe, 2
Louvain-la-Neuve 1348
BE

Email: sebastien.barre@uclouvain.be
URI: http://inl.info.ucl.ac.be/sbarre

Olivier Bonaventure
Universite catholique de Louvain
Place Ste Barbe, 2
Louvain-la-Neuve 1348
BE

URI: http://inl.info.ucl.ac.be/obo