Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 12, 2015
P. Pfister
B. Paterson
Cisco Systems
J. Arkko
Ericsson
February 8, 2015

Distributed Prefix Assignment Algorithm
draft-ietf-homenet-prefix-assignment-03

Abstract

This document specifies a distributed algorithm for automatic prefix assignment. Given a set of delegated prefixes, it ensures that at most one prefix is assigned from each delegated prefix to each link. Nodes may assign available prefixes to the links they are directly connected to, or for other private purposes. The algorithm eventually converges and ensures that all assigned prefixes do not overlap.

Table of Contents

1. Introduction

This document specifies a distributed algorithm for automatic prefix assignment. Given a set of delegated prefixes, nodes may assign available prefixes to links they are directly connected to, or for their private use. The algorithm ensures that the following assertions are eventually true:

1. At most one prefix from each delegated prefix is assigned to each link.

2. Assigned prefixes are not included in and do not include other assigned prefixes.

3. Assigned prefixes do not change in the absence of topology or configuration changes.

In the rest of this document the two first conditions are referred to as the correctness conditions of the algorithm while the third condition is referred to as its convergence condition.

Each assignment has a priority specified by the node making the assignment, allowing for more advanced assignment policies. When multiple nodes assign different prefixes from the same delegated prefix to the same link, or when multiple nodes assign overlapping prefixes, the assignment with the highest priority is kept and other assignments are removed.

The prefix assignment algorithm requires that participating nodes share information through a flooding mechanism. If the flooding mechanism ensures that all messages are propagated to all nodes faster than a given timing upper bound, the algorithm also ensures that all assigned prefixes used for networking operations (e.g., host configuration) remain unchanged, unless another node assigns an overlapping prefix with a higher assignment priority, or the topology changes and renumbering cannot be avoided.

2. Terminology

In this document, the key words "MAY", "MUST, "MUST NOT", "OPTIONAL", and "SHOULD", are to be interpreted as described in [RFC2119].

This document makes use of the following terminology:

Link: An object the distributed algorithm will assign prefixes to. A Node may only assign prefixes to Links it is directly connected to. A Link is either Shared or Private.

Private Link: A Private Link is an abstract concept defined for the sake of this document. It allows nodes to make assignments for their private use or delegation. For instance, every DHCPv6-PD [RFC3633] client MAY be considered as a different Private Link.

Shared Link: A Link multiple nodes may be connected to. Most of the time, a Shared Link would consist in a multi-access link or point-to-point link, virtual or physical, requiring prefixes to be assigned to.

Delegated Prefix: A prefix provided to the algorithm and used as a prefix pool for Assigned Prefixes.

Node ID: A value identifying a given participating node. The set of identifiers MUST be strictly and totally ordered (e.g., using the alphanumeric order).

Flooding Mechanism: A mechanism implementing reliable broadcast and used to advertise published Assigned Prefixes.

Flooding Delay: Value which SHOULD be provided by the Flooding Mechanism indicating a deterministic or likely upper bound of the information propagation delay. When the Flooding Mechanism does not provide a value, it is set to DEFAULT_FLOODING_DELAY (Section 7).

Advertised Prefix: A prefix advertised by another node and delivered to the local node by the Flooding Mechanism. It has an Advertised Prefix Priority and, when assigned to a directly connected Shared Link, is associated with a Shared Link.

Advertised Prefix Priority: A value that defines the priority of an Advertised Prefix received from the Flooding Mechanism or a published Assigned Prefix. Whenever multiple Advertised Prefixes are conflicting, all Advertised Prefixes but the one with the greatest priority will eventually be removed. In case of tie, the assignment advertised by the node with the greatest Node ID is kept and others are removed. In order to ensure convergence, the range of priority values MUST have an upper bound.

Assigned Prefix: A prefix included in a Delegated Prefix and assigned to a Shared or Private Link. It represents a local decision to assign a given prefix from a given Delegated Prefix to a given Link. The algorithm ensures that there never is more than one Assigned Prefix per Delegated Prefix and Link pair. When destroyed, an Assigned Prefix is set as not applied, ceases to be advertised, and is removed from the set of Assigned Prefixes.

Applied (Assigned Prefix): When an Assigned Prefix is applied, it MAY be used (e.g., for host configuration, routing protocol configuration, prefix delegation). When not applied, it MUST NOT be used for any other purposes than the prefix assignment algorithm. Each Assigned Prefix is associated with a timer (Apply Timer) used to apply the Assigned Prefix. An Assigned Prefix is unapplied when destroyed.

Published (Assigned Prefix): The Assigned Prefix is advertised through the Flooding Mechanism as assigned to its associated Link. A published Assigned Prefix MUST have an Advertised Prefix Priority. It will appear as an Advertised Prefix to other nodes, once received through the Flooding Mechanism.

Prefix Adoption: When an Advertised Prefix which does not conflict with any other Advertised Prefix or published Assigned Prefix stops being advertised, any other node connected to the same Link MAY, after some random delay, start advertising the same prefix. This procedure is called adoption and provides seamless assignment transfer from a node to another, e.g., in case of node failure.

Backoff Timer: Every Delegated Prefix and Link pair is associated with a timer counting down to zero. It is used to avoid multiple nodes from making colliding assignments by delaying the creation of new Assigned Prefixes or the advertisement of adopted Assigned Prefixes by a random amount of time.

Renumbering: Event occurring when an Assigned Prefix which was applied is destroyed. It is undesirable as it usually implies reconfiguring routers or hosts.

3. Applicability statement

Each node MUST have a set of disjoint Delegated Prefixes. This set MAY change over time and be different from one node to another at some point, but nodes MUST eventually have the same set of disjoint Delegated Prefixes.

Given this set of disjoint Delegated Prefixes, nodes may assign available prefixes from each Delegated Prefix to the Links they are directly connected to. The algorithm ensures that at most one prefix from a given Delegated Prefix is assigned to a given Link.

The algorithm can be applied to any address space and can be used to manage multiple address spaces simultaneously. For instance, an implementation can make use of IPv4-mapped IPv6 addresses [RFC4291] in order to manage both IPv4 and IPv6 prefix assignment using a single prefix space.

The algorithm supports dynamically changing topologies:

- Nodes may join or leave the set of participating nodes.
- Nodes may join or leave Links.
- Links may be joined or split.

All nodes MUST run a common Flooding Mechanism in order to share published Assigned Prefixes. The set of participating nodes is defined as the set of nodes participating in the Flooding Mechanism.

The Flooding Mechanism MUST:

- Provide a way to flood Assigned Prefixes assigned to a directly connected Link along with their respective Advertised Prefix Priority and the Node ID of the node which advertises it.
- Specify whether an Advertised Prefix was assigned to a directly connected Shared Link, and if so, on which one.

In addition, a Flooding Delay SHOULD be specified and respected in order to avoid renumbering. If not specified, or whenever the Flooding Mechanism is unable to respect the provided delay, renumbering may happen. As such delay often depends on the size of the network, it MAY change over time and MAY be different from one node to another.

The algorithm ensures that whenever the Flooding Delay is provided and respected, and in the absence of topology change or delegated prefix removal, renumbering never happens.

Each node MUST have a Node ID. Node IDs MAY change over time and be the same on multiple nodes at some point, but each node MUST eventually have a Node ID which is unique among the set of participating nodes.

4. Algorithm Specification

This section specifies the behavior of nodes implementing the prefix assignment algorithm.

4.1. Algorithm Terminology

The algorithm makes use of the following terms:

Current Assignment: For a given Delegated Prefix and Link, the Current Assignment is the Assigned Prefix (if any) included in the Delegated Prefix and assigned to the given Link.

Precedence: An Advertised Prefix takes precedence over an Assigned Prefix if and only if:

* The Assigned Prefix is not published.

* The Assigned Prefix is published and the Advertised Prefix Priority from the Advertised Prefix is strictly greater than the Advertised Prefix Priority from the Assigned Prefix.

* The Assigned Prefix is published, the priorities are identical, and the Node ID from the node advertising the Advertised Prefix is strictly greater than the local Node ID.

Best Assignment: For a given Delegated Prefix and Link, the Best Assignment is (if any) the Advertised Prefix:

* Including or included in the Delegated Prefix.

* Assigned on the given Link.

  * Having the greatest Advertised Prefix Priority among Advertised Prefixes assigned on the given Link (and, in case of tie, the prefix advertised by the node with the greatest Node ID among all prefixes with greatest priority).

  * Taking precedence over the Current Assignment associated with the same Link and Delegated Prefix (if any).

Valid (Assigned Prefix) An Assigned Prefix is valid if and only if the two following conditions are met:

  * No Advertised Prefix including or included in the Assigned Prefix takes precedence over the Assigned Prefix.

  * No Advertised Prefix including or included in the same Delegated Prefix as the Assigned Prefix and assigned to the same Link takes precedence over the Assigned Prefix.

4.2. Prefix Assignment Algorithm Routine

This section specifies the prefix assignment algorithm routine. It is defined for a given Delegated Prefix/Link pair and may be run either as triggered by the Backoff Timer, or not.

For a given Delegated Prefix and Link pair, the routine MUST be run as not triggered by the Backoff Timer whenever:

o An Advertised Prefix including or included in the considered Delegated Prefix is added or removed.

o An Assigned Prefix included in the considered Delegated Prefix and associated with a different Link than the considered Link was destroyed, while there is no Current Assignment associated with the given pair. This case MAY be ignored if the creation of a new Assigned Prefix associated with the considered pair is not desired.

o The considered Delegated Prefix is added.

o The considered Link is added.

o The Node ID is modified.

Additionally, for a given Delegated Prefix and Link pair, the routine MUST be run as triggered by the Backoff Timer whenever:

o The Backoff Timer associated with the considered Delegated Prefix/ Link pair fires while there is no Current Assignment associated with the given pair.

When such an event occurs, a node MAY delay the execution of the routine instead of executing it immediately, e.g. while receiving an update from the Flooding Mechanism, or for security reasons (see Section 8). Even though other events occur in the meantime, the routine MUST be run only once. It is also assumed that, whenever one of these events is the Backoff Timer firing, the routine is executed as triggered by the Backoff Timer.

In order to execute the routine for a given Delegated Prefix/Link pair, first look for the Best Assignment and Current Assignment associated with the Delegated Prefix/Link pair, then execute the corresponding case:

1. If there is no Best Assignment and no Current Assignment: Decide whether the creation of a new assignment for the given Delegated Prefix/Link pair is desired (As any result would be valid, the way the decision is taken is out of the scope of this document) and do the following:

   * If it is not desired, stop the execution of the routine.

   * Else if the Backoff Timer is running, stop the execution of the routine.

   * Else if the routine was not executed as triggered by the Backoff Timer, set the Backoff Timer to some random delay between ADOPT_MAX_DELAY and BACKOFF_MAX_DELAY (see Section 7) and stop the execution of the routine.

   * Else, continue the execution of the routine.

   Select a prefix for the new assignment (see Section 5 for guidance regarding prefix selection). This prefix MUST be included in or be equal to the considered Delegated Prefix and MUST NOT include or be included in any Advertised Prefix. If a suitable prefix is found, use it to create a new Assigned Prefix:

   * Assigned to the considered Link.

   * Not applied.

   * The Apply Timer set to '2 * Flooding Delay'.

   * Published with some selected Advertised Prefix Priority.

2. If there is a Best Assignment but no Current Assignment: Cancel the Backoff Timer and use the prefix from the Best Assignment to create a new Assigned Prefix:

   * Assigned to the considered Link.

   * Not applied.

   * The Apply Timer set to '2 * Flooding Delay'.

   * Not published.

3. If there is a Current Assignment but no Best Assignment:

   * If the Current Assignment is not valid, destroy it, and execute the routine again, as not triggered by the Backoff Timer.

   * If the Current Assignment is valid and published, stop the execution of the routine.

   * If the Current Assignment is valid and not published, the node MAY either:

     + Adopt the prefix by cancelling the Apply Timer and set the Backoff Timer to some random delay between 0 and ADOPT_MAX_DELAY (see Section 7). This procedure is used to avoid renumbering when the node advertising the prefix left the Shared Link.

     + Destroy it and execute case 1 in order to create a different assignment.

4. If there is a Current Assignment and a Best Assignment:

   * Cancel the Backoff Timer.

   * If the two prefixes are identical, set the Current Assignment as not published. If the Current Assignment is not applied and the Apply Timer is not set, set the Apply Timer to '2 * Flooding Delay'.

   * If the two prefixes are not identical, destroy the Current Assignment and go to case 2.

When the prefix assignment algorithm routine requires an assignment to be created or adopted, any Advertised Prefix Priority value can be used. Other documents MAY provide restrictions over this value

depending on the context the algorithm is operating in, or leave it as implementation-specific.

When the prefix assignment algorithm routine requires an assignment to be created or adopted, the chosen Advertised Prefix Priority is unspecified (any value would be valid). The values to be used in such situations MAY be specified by other documents making use of the prefix assignment algorithm or be left as an implementation specific choice.

4.3. Overriding and Destroying Existing Assignments

In addition to the behaviors specified in Section 4.2, the following procedures MAY be used in order to provide more advanced behavior (Section 6):

Overriding Existing Assignments: For any given Link and Delegated Prefix, a node MAY create a new Assigned Prefix using a chosen prefix and Advertised Prefix Priority such that:

* The chosen prefix is included in or is equal to the considered Delegated Prefix.

* The Current Assignment, if any, as well as all existing Assigned Prefixes which include or are included inside the chosen prefix, are destroyed.

* It is not applied.

* The Apply Timer set to '2 * Flooding Delay'.

* It is published.

* The Advertised Prefix Priority is greater than the Advertised Prefix Priority from all Advertised Prefixes which include or are included in the chosen prefix.

In order to ensure algorithm convergence:

* Such overriding assignments MUST NOT be created unless there was a change in the node configuration, a Link was added, or an Advertised Prefix was added or removed.

* The chosen Advertised Prefix Priority for the new Assigned Prefix SHOULD be greater than all priorities from the destroyed Assigned Prefixes. If not, simple topologies with only two nodes may not converge. Nodes which do not respect this rule MUST implement a mechanism which detects whether the

distributed algorithm do not converge and, whenever this would happen, stop creating overriding Assigned Prefixes which do not hold this rule. The specifications for such safety procedures are out of the scope of this document.

Removing an Assigned Prefix: A node MAY destroy any Assigned Prefix which is published. Such an event reflects the desire from a node to not assign a prefix from a given Delegated Prefix to a given Link anymore. In order to ensure algorithm convergence, such procedure MUST NOT be executed unless there was a change in the node configuration. Additionally, whenever an Assigned Prefix is destroyed this way, the prefix assignment algorithm routine MUST be run for the Delegated Prefix/Link pair associated with the deleted Assigned Prefix.

These procedures are OPTIONAL. They could be used for diverse purposes, e.g., for providing custom prefix assignment configuration or reacting to prefix space exhaustion (by overriding short Assigned Prefixes and assigning longer ones).

4.4. Other Events

When the Apply Timer fires, the associated Assigned Prefix MUST be applied.

When the Backoff Timer associated with a given Delegated Prefix/Link pair fires while there is a Current Assignment associated with the same pair, the Current Assignment MUST be published with some associated Advertised Prefix Priority and, if the prefix is not applied, the Apply Timer MUST be set to '2 * Flooding Delay'.

When a Delegated Prefix is removed from the set of Delegated Prefixes, all Assigned Prefixes included in the removed Delegated Prefix MUST be destroyed.

When one Delegated Prefix is replaced by another one that includes or is included in the deleted Delegated Prefix, all Assigned Prefixes which were included in the deleted Delegated Prefix but are not included in the added Delegated Prefix MUST be destroyed. Others MAY be kept.

When a Link is removed, all Assigned Prefixes assigned to that Link MUST be destroyed.

5. Prefix Selection Considerations

When the prefix assignment algorithm routine specified in Section 4.2 requires a new prefix to be selected, the prefix MUST be selected either:

- Among prefixes which were previously assigned and applied on the considered Link. For that purpose, Applied Prefixes may be stored in stable storage along with their associated Link.

- Randomly, picked in a set of at least RANDOM_SET_SIZE (see Section 7) candidate prefixes. If less than RANDOM_SET_SIZE candidates can be found, the prefix MUST be picked among all candidates.

- Based on some custom selection process specified in the configuration.

A simple implementation MAY randomly pick the prefix among all available prefixes, but this strategy is inefficient in terms of address space use as a few long prefixes may exhaust the pool of available short prefixes.

The rest of this section describes a more efficient approach which MAY be applied any time a node needs to pick a prefix for a new assignment. The two following definitions are used:

Available prefix: The prefix A/N is available if and only if A/N does not include and is not included in any Assigned or Advertised Prefix but A/(N-1) does include an Assigned or Advertised Prefix (or N equals 0 and there is no Assigned or Advertised Prefixes at all).

Candidate prefix: A prefix which is included in or is equal to an available prefix.

The procedure described in this section takes the three following criteria into account:

Stability: In some cases, it is desirable that the selected prefix remains the same across executions and reboots. For this purpose, prefixes previously applied on the Link or pseudo-random prefixes generated based on node and Link specific values may be considered.

Randomness: When no stored or pseudo-random prefix is chosen, a prefix may be randomly picked among RANDOM_SET_SIZE candidates of

desired length. If less than RANDOM_SET_SIZE candidates can be found, the prefix is picked among all candidates.

Addressing-space usage efficiency: In the process of assigning prefixes, a small set of badly chosen long prefixes may prevent any shorter prefix from being assigned. For this reason, the set of RANDOM_SET_SIZE candidates is created from the set of available prefixes with longest prefix lengths and, in case of tie, preferring small prefix values.

When executing the procedure, do as follows:

1. For each prefix stored in stable-storage, check if the prefix is included in or equal to an available prefix. If so, pick that prefix and stop.

2. For each prefix length, count the number of available prefixes of the given length.

3. If the desired prefix length was not specified, select one. The available prefixes count computed previously may be used to help picking a prefix length such that:

   * There is at least one candidate prefix.

   * The prefix length is chosen great enough to not exhaust the address space.

   Let N be the chosen prefix length.

4. Iterate over available prefixes starting with prefixes of length N down to length 0 and create a set of RANDOM_SET_SIZE candidate prefixes of length exactly N included in or equal to available prefixes. The end goal here is to create a set of RANDOM_SET_SIZE candidate prefixes of length N included in a set of available prefixes of maximized prefix length. In case of a tie, smaller prefix values (as defined by the bit-wise lexicographical order) are preferred.

5. For each pseudo-random prefix, check if the prefix is equal to a candidate prefix. If so, pick that prefix and stop.

6. Choose a random prefix from the set of selected candidates.

The complexity of this procedure is equivalent to the complexity of iterating over available prefixes. Such operation may be accomplished in linear time, e.g., by storing Advertised and Assigned Prefixes in a binary trie.

6. Implementation Capabilities and Node Behavior

Implementations of the prefix assignment algorithm may vary from very basic to highly customizable, enabling different types of fully interoperable behaviors. The three following behaviors are given as examples:

Listener: The node only acts upon assignments made by other nodes, i.e, it never creates new assignments nor adopt existing ones. Such behavior does not require the implementation of the considerations specified in Section 5 or Section 4.3. The node never checks existing assignments validity, which makes this behavior particularly suited to lightweight devices which can rely on more capable neighbors to make assignments on directly connected Shared Links.

Basic: The node is capable of assigning new prefixes or adopting prefixes which do not conflict with any other existing assignment. Such behavior does not require the implementation of the considerations specified in Section 4.3. It is suited to situations where there is no preference over which prefix should be assigned to which Link, and there is no priority between different Links.

Advanced: The node is capable of assigning new prefixes, adopting existing ones, making overriding assignments and destroying existing ones. Such behavior requires the implementation of the considerations specified in Section 5 and Section 4.3. It is suited when the administrator desires some particular prefix to be assigned on a given Link, or some Links to be assigned prefixes with a greater priority.

7. Algorithm Parameters

This document does not provide values for ADOPT_MAX_DELAY, BACKOFF_MAX_DELAY and RANDOM_SET_SIZE. The algorithm ensures convergence and correctness for any chosen values, even when these are different from node to node. They MAY be adjusted depending on the context, providing a tradeoff between convergence time, efficient addressing, low verbosity (less traffic is generated by the Flooding Mechanism), and low collision probability.

ADOPT_MAX_DELAY (respectively BACKOFF_MAX_DELAY) represents the maximum backoff time a node may wait before adopting an assignment (respectively making a new assignment). BACKOFF_MAX_DELAY MUST be greater than or equal to ADOPT_MAX_DELAY. The greater ADOPT_MAX_DELAY and (BACKOFF_MAX_DELAY - ADOPT_MAX_DELAY), the lower

the collision probability and the verbosity, but the greater the convergence time.

RANDOM_SET_SIZE represents the desired size of the set a random prefix will be picked from. The greater RANDOM_SET_SIZE, the better the convergence time and the lower the collision probability, but the worse the addressing-space usage efficiency.

When the Flooding Mechanism does not provide a Flooding Delay, it is set to DEFAULT_FLOODING_DELAY. As participating nodes do not need to agree on a common Flooding Delay value, this default value MAY be different from one node to another. If the context in which the algorithm is used does not suffer from renumbering, the value 0 MAY be used. Otherwise it depends on the Flooding Mechanism properties and the desired renumbering probability, and is therefore out of scope of this document.

8. Security Considerations

The prefix assignment algorithm functions on top of two distinct mechanisms, the Flooding Mechanism and the Node ID assignment mechanism.

An attacker able to publish Advertised Prefixes through the flooding mechanism may perform the following attacks:

* Publish a single overriding assignment for a whole Delegated Prefix or for the whole address space, thus preventing any node from assigning prefixes to Links.

* Quickly publish and remove Advertised Prefixes, generating traffic at the Flooding Mechanism layer and causing multiple executions of the prefix assignment algorithm in all participating nodes.

* Publish and remove Advertised Prefixes in order to prevent the convergence of the execution.

An attacker able to prevent other nodes from accessing a portion or the whole set of Advertised Prefixes may compromise the correctness of the execution.

An attacker able to cause repetitive Node ID changes may induce traffic generation from the Flooding Mechanism and multiple executions of the prefix assignment algorithm in all participating nodes.

An attacker able to publish Advertised Prefixes using a Node ID used by another node may prevent the correctness and convergence of the execution.

Whenever the security of the Flooding Mechanism and Node ID assignment mechanism could not be ensured, the convergence of the execution may be prevented. In environments where such attacks may be performed, the execution of the prefix assignment algorithm routine SHOULD be rate limited, as specified in Section 4.2.

9. IANA Considerations

This document has no actions for IANA.

10. Acknowledgments

The authors would like to thank those who participated in the previous document's version development as well as the present one. In particular, the authors would like to thank Tim Chown, Fred Baker, Mark Townsley, Lorenzo Colitti, Ole Troan, Ray Bellis, Markus Stenberg, Wassim Haddad, Joel Halpern, Samita Chakrabarti, Michael Richardson, Anders Brandt, Erik Nordmark, Laurent Toutain, Ralph Droms, Acee Lindem and Steven Barth for interesting discussions and document review.

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

11.2. Informative References

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6", RFC 3633, December 2003.

Appendix A. Static Configuration Example

This section describes an example of how custom configuration of the prefix assignment algorithm may be implemented.

The node configuration is specified as a finite set of rules. A rule is defined as:

- A prefix to be used.

- A Link on which the prefix may be assigned.

- An Assigned Prefix Priority (smallest possible Assigned Prefix Priority if the rule may not override other Assigned Prefixes).

- A rule priority (0 if the rule may not override existing Advertised Prefixes).

In order to ensure the convergence of the execution, the Assigned Prefix Priority MUST be an increasing function (not necessarily strictly) of the configuration rule priority (i.e. the greater is the configuration rule priority, the greater the Assigned Prefix Priority must be).

Each Assigned Prefix is associated with a rule priority. Assigned Prefixes which are created as specified in Section 4.2 are given a rule priority of 0.

Whenever the configuration is changed or the prefix assignment algorithm routine is run: For each Link/Delegated Prefix pair, look for the configuration rule with the highest configuration rule priority such that:

- The prefix specified in the configuration rule is included in the considered Delegated Prefix.

- The Link specified in the configuration rule is the considered Link.

- All the Assigned Prefixes which would need to be destroyed in case a new Assigned Prefix is created from that configuration rule (as specified in Section 4.3) have an associated rule priority which is strictly lower than the one of the considered configuration rule.

- The assignment would be valid when published with an Advertised Prefix Priority equal to the one specified in the configuration rule.

If a rule is found, a new Assigned Prefix is created based on that rule in conformance with Section 4.3. The new Assigned Prefix is associated with the Advertised Prefix Priority and the rule priority specified in the considered configuration rule.

Note that the use of rule priorities ensures the convergence of the execution.

Authors' Addresses

Pierre Pfister
Cisco Systems
Paris
France

Email: pierre.pfister@darou.fr

Benjamin Paterson
Cisco Systems
Paris
France

Email: benjamin@paterson.fr

Jari Arkko
Ericsson
Jorvas 02420
Finland

Email: jari.arkko@piuha.net