Hierarchical Host Identity Tag Architecture
draft-jiang-hiprg-hhit-arch-00.txt

Abstract

This document analyzes the problems and limitation of the current flat-structured Host Identity Tag (HIT, [RFC4423]) architecture. The document specifies a hierarchical HIT architecture which is compatible with the flat-structured HIT architecture. This architecture and the process of HITs generation ensure the global uniqueness of HITs. It also enables the multiple HIP management domains, solves the deployment problem of current flat-structured HIT architecture. It also enhances the scalability and resolution efficiency of the mapping system.
1. Introduction

This document analyzes the problems and limitation of the current flat-structured Host Identity Tag (HIT, [RFC4423]) architecture. The document specifies a hierarchical HIT architecture, which splits a HIT into two parts: a HIP management domain tag and a host tag. The proposed hierarchical HIT architecture is also compatible with the flat-structured HIT architecture. The format of HIT and the detail process of HITs generation are defined. This architecture and the process of HITs generation ensure the global uniqueness of HITs. This architecture also enables the multiple HIP management domains, solves the deployment problem of current flat-structured HIT architecture. It also enhances the scalability and resolution efficiency of the mapping system.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [1].

3. Analysis of the Current Flat-structured HIT Architecture

The original HIT concept was defined in [RFC4423]. ‘’A Host Identity Tag (HIT) is used in other protocols to represent the Host Identity.’’ It is a quite restricted definition. However, [HIP-base] has updated the HIT concept and enhanced the functionality of HIT. ‘’... the Host
Identity Tag (HIT), becomes the operational representation. It is 128 bits long and is used in the HIP payloads and to index the corresponding state in the end hosts.

In order to be able to represent hosts, the uniqueness of HITs is required in global scope. "In the HIP packets, the HITs identify the sender and recipient of a packet. Consequently, a HIT should be unique in the whole IP universe as long as it is being used."

Although mathematically "the probability of HIT collision between two hosts is very low" [HIP-base], there is no mechanism to ensure that a HIT is global unique. The current defined HIT is generated according to the ORCHID generation method described in [RFC4843]. [RFC4843] suggests "several possible methods ... to preserve a low enough probability of collisions". However, it cannot guarantee the global uniqueness of HITs. Furthermore, while the number of end devices continuously grows in the future, the possibility of HIT collision will increase rapidly. A technical mechanism is needed to ensure the global uniqueness of HITs, particularly with the consideration that collisions may happen.

[RFC 4423] states "In the extremely rare case of a single HIT mapping to more than one Host Identity, the Host Identifiers (public keys) will make the final difference." It means the mapping system between HIP and IP must store or at least be aware of the Host Identifiers of all hosts. Given the facts that the Host Identifiers is quite large and may be in various formats, the storage and management burden of the mapping system could be quite high. If there was a mechanism to ensure the global uniqueness of HITs, then, the mapping system would not have to be aware the Host Identifiers.

Furthermore, within the flat-structured HIT architecture, the robustness of resolution efficiency in the supporting mapping system is in a big question mark: a mapping server has to hold or at least to be able to access a large database that contains all HITs information in the global scope. The number of HITs is at least in billion-level giving the fact there are billions hosts now. In the future, it may rapidly grow up to trillion-level, or even higher. The storage burden, maintenance consumption and synchronization updating are problems that are very difficult to solve. For each single looking up operation, one may search through most of the database, on average, O(number of total global HITs). It is unfeasible for both computing power and time reasons.

One more disadvantage that the flat-structured HIT architecture is the difficulties for management. There is no common between HITs that
their HIs assigned by the same authority or that their represented hosts have the same properties. Hence, it is difficult to categorize HITS. The ACL operators have to have explicit list of HIs in the ACL. Contrarily, the hierarchical HIs are aggregatable. It makes HIs manageable. Each network manager just needs to manage and maintain HIs and their mapping information in a relatively small range.

According to the above analysis, it is nature to break up the flat HIT architecture into hierarchy. It can effectively break up global uniqueness into smaller scope uniqueness. It can improve the resolution processing and enhance the scalability and resolution efficiency. Furthermore, it can optimize the management of both the host identity and the mapping database. Each management domain is responsible only for a part of the global HIT architecture. However, it is useful that the new hierarchical HIT architecture keep compatible with the flat HIT architecture for privacy purpose and other usage scenarios.

4. Hierarchical HIT Architecture

In this document, we introduce a two-level hierarchically structured HIT architecture. HIT is “128 bits long value and is used in the HIP payloads and to index the corresponding state in the end hosts.” [HIP-base] “In the HIP packets, the HIs identify the sender and recipient of a packet.” [RFC 4423] HIs refer to nodes or virtual nodes. All nodes are required to have at least one HIT. A single node may also have multiple HIs. Applications on a same node may bind to different HIs. This is sometimes convenient for point-to-point communications.

We break the current 128-bit flat-structured HIT into two parts: 32-bit HIP management domain tag and 96-bit host tag. It can represent maximum $2^{32}$ management domains and $2^{96}$ hosts within each management domain.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>HIP management domain tag</td>
<td>host tag</td>
</tr>
</tbody>
</table>

For the secure consideration, we assign more bits to the host tag, which is hash output, leaving less but enough bits for HIP management domain tag. The more the number of bits the host tag is, the more secure it is against brute-force attacks. In the worst case, if the hash algorithm cannot be inverted, the expected number of iterations required for a brute force attack is $O(2^{96})$ in order to find a host identity that matches with a given host tag.
The HIP management domain, as its literal, is a logic region in which the HIs of all nodes are assigned by the same authority. Within a same HIP management domain, all the nodes should have the same HIP management domain tag or the same leftmost certain bits. Furthermore, the authority may be organized internally hierarchically.

The HIP management domain tag should be assigned by a global management organization with the principle that every HIP management domain tag must be globally unique.

Consequentially, the HIP management domain tags may be organized hierarchically. For example, a big organization may obtain a block of HIP management tags with a given leftmost 24-bit. It then can assign 32-bit HIP management domain tags to its sub-organizations. All these sub-organizations have the same leftmost 24-bit.

The host tags remain the original meaning of HIT — ‘a hashed encoding of the Host Identity’. For each HIP management domain, it is mandatory to maintain the uniqueness of all host tags. It is guaranteed by the process of generating a HIT, see Section 5.

For index and resolution purposes, HITs are aggregatable with management domain tags of arbitrary bit-length, similar to IPv4 addresses under Classless Inter-Domain Routing [RFC4632].

4.1. Compatible flat-structured HITs

Obviously, not all hosts are willing to use hierarchical HITs in all scenarios for various reasons, such as privacy, etc. Therefore, it is useful that the hierarchical HIT architecture keep compatible with the flat HIT architecture.

The flat HITs can be defined as a specific sub-set of the hierarchical HITs architecture. With a the same reserved Flat HIT Tag (2 or 3 bits) at the beginning and the number of bits that can be chosen arbitrarily reduce 2 or 3 bits out of 128, flat HITs can be used as defined in [RFC 4423].

<table>
<thead>
<tr>
<th>128 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>[FHIT Tag] flat host identity tag</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
</tr>
</tbody>
</table>

4.2. HITs on HIP-enabled nodes

HIP-enabled nodes may have considerable or little knowledge of the internal structure of the hierarchical HIT, depending on the role the
node plays (for instance, host versus mapping server). At a minimum, a node may consider pre-generated HITs have no internal structure:

```
|                             128 bits |
+-------------------------------------+
|                                   host identity tag |
+-------------------------------------+
```

Only sophisticated hosts may additionally be aware of the type of their HITs and use the hierarchical structure of HITs to simplify the resolution procedure.

5. Generating a hierarchical HIT

The process of generating a new hierarchical HIT takes three input values: a 32-bit HIP management domain tag, a 2-bit collusion count, the host identity (the public key of an asymmetric key pair). A hierarchical HIT should be generated as follows:

1. Set the 2-bit collision count to zero.
2. Concatenate from left to right the HIP management domain tag, the collusion count, and the host identity. Execute the SHA-1 algorithm on the concatenation. Take the 94 leftmost bits of the SHA-1 hash value.
3. Concatenate from left to right the 32-bit HIP management domain tag, the 2-bit collusion count and 94-bit hash output to form a 128-bit HIT.
4. Perform duplicate detection within the HIP management domain scope. If a HIT collision is detected, increment the collision count by one and go back to step 2. However, after four collisions, stop and report the error.

The design that includes the HIP management domain tag in the hash input is mainly against the re-computation attack: create a database of HITs and matching public keys. With the design, an attacker must create a separate database for each HIP management domain.

The design reduces the number of bit of hash output from 96 to 94. It does reduce the safety. However, \( O(2^{94}) \) iterations is large enough to prevent brute-force attacks.

For security reason, the abovementioned SHA-1 hash algorithm may be replaced any safer algorithm.
6. Security Considerations

The most important security property of HIT is that it is self-certifying (i.e., given a HIT, it is computationally hard to find a Host Identity key that matches the HIT). Although this document limits the hash output to be 94-bit long, it does not affect the self-certifying security property.

7. IANA Considerations

This document defines a new namespace: HIP management domain tag. It is a 32-bit long value, which represents a globally unique HIP management domain. IANA may found an authority institute to manage the global assignment of HIP management domain tag.

8. References

8.1. Normative References


8.2. Informative References

[RFC4632] V. Fuller, ''Classless Inter-Domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan'', RFC4632, August 2006.

Author’s Addresses

Sheng Jiang
Huawei Technologies
QuikKe Bld., No.6 Rd, Xinxi St.,
Shang-Di Information Industry Base,
Hai-Dian District, Beijing, P.R. China
100085
Phone: 86-10-82836774
Email: shengjiang@huawei.com

Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any
Intellectual Property Rights or other rights that might be claimed to
pertain to the implementation or use of the technology described in
this document or the extent to which any license under such rights
might or might not be available; nor does it represent that it has
made any independent effort to identify any such rights. Information
on the procedures with respect to rights in RFC documents can be
found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any
assurances of licenses to be made available, or the result of an
attempt made to obtain a general license or permission for the use of
such proprietary rights by implementers or users of this
specification can be obtained from the IETF on-line IPR repository at

The IETF invites any interested party to bring to its attention any
copyrights, patents or patent applications, or other proprietary
rights that may cover technology that may be required to implement
this standard. Please address the information to the IETF at
ietf-ipr@ietf.org.

Disclaimer of Validity

This document and the information contained herein are provided on an
"AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS
OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND
THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS
OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF
THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED
WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.
Copyright Statement

Copyright (C) The IETF Trust (2008).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.