A Naming Scheme for Identifiers in HiiMap

Christoph Spleiß and Oliver Hanka
Technische Universität München, Germany
christoph.spleiss@tum.de, oliver.hanka@tum.de

I. Introduction

Many researchers agreed that the split of locator and identifier seems to be a very promising approach. Although this solution addresses the most critical issues in todays Internet architecture, new challenges arise through the mapping system which is necessary to resolve identifiers into the corresponding locators. One interesting question is how the naming of identifiers is done. In this work we give an overview of a naming scheme for identifiers based on the HiiMap locator/identifier split Internet architecture. The naming scheme supports user friendly identifiers for hosts, content and persons and does not rely on DNS. We furthermore give a possible solution for a search algorithm that can deal with spelling mistakes.

II. HiiMap Architecture

The HiiMap architecture [1] uses never changing identifier, so called UIDs (unique ID) and two-tier locators. One part of the locator is the LTA (local temporary address) which is assigned by a provider and routable inside the provider's own network. The other part is the gUID (gateway UID). This is a global routable address of the provider's border gateway router and specifies an entrance point into the network.

HiiMap splits the mapping system into different regions, whereby each region is its own independent mapping system that is responsible for the UID/locator mappings of nodes/content/persons registered in this region. In order to query for UIDs which region is not known, a region prefix (RP) to any UID is introduced. This RP can be queried at the so called Global Authority (GA), which resolves UIDs to RPs.

If a user wants to contact a specific node/content/person in the HiiMap architecture he needs to know its UID. However, as the UID is a bit representation of fixed length (i.e. 128 bit) it is very hard for humans to remember such values. One possible solution would be to use the existing Domain Name System to resolve user friendly names with variable length for hosts/content/persons into UIDs that can be resolved by the mapping system in a second step. However, we do not want to to have two database systems for resolving user friendly names to locators.

III. New naming scheme for Identifiers

In order to allow user friendly names for nodes, content or persons, a hashing algorithm H can be used to generate the corresponding UID which is stored in the mapping system. The registration process for a new UID is quite the same like with DNS names. The corresponding region checks, like the NIC, if a specific UID is already registered. If not, the new UID is stored in the system. If somebody wants to access a specific node, content or person, he can calculate the UID by simply hashing the corresponding name. However, just using e.g. a persons name and applying a hashing function is not sufficient for generating a personal identifier, as more people can have the same name. Similar problems occur with identifiers for content. In the following we contribute a generalized scheme how to generate global unique identifiers for hosts, content and persons. It allows to store all these identifiers in the same mapping database and is yet flexible enough to support different databases for different types of identifiers.

Fig. 1. UID with regional prefix RP

A. Generalized Identifier

Figure 1 shows the generalized structure of a UID with the prepended region prefix (RP), a type field (T), the hash value of a human friendly name for hosts, content or persons, and two extension fields (Ext 1 and Ext2) that are explained in the following subsections. The grey fields are part of the UID and stored in the mapping system of a specific region, denoted by RP.

The type field T denotes if the identifier is for hosts, content or persons. Although we only use three different types here, it can be easily extended to more. As the type-field contains the most significant bits (MSB) in the UID, it is also possible to map different identifier types to different databases in the mapping system. We suggest 128 bit for the UID, whereby 4 bits are for the type, 92 bits for the hash value, and 16 bit for each extension field.

B. Identifiers for Hosts

Identifiers for hosts is the most common use case these days. Today we use DNS to resolve hostnames to IP addresses in order to access a specific machine [2]. For generating the UID for a specific machine we use a human readable hostname like google.com and apply a 108 bit hash function. Note that we also use Ext 1 for the hashed hostname. The Ext 2 field is used to address a specific service on the host. However, the value in Ext 2 is not necessary when requesting the locator for a specific host from the mapping system and is therefore to zero. It would make no sense to store identifiers for each node's service that all point to the same locator.

C. Identifiers for Content

As the focus of the internet is shifting from accessing specific nodes to accessing information and content, different approaches towards a content-centric network have been made [3]. Hereby, each content, which can be e.g. a webpage or an audio or video file, gets its own UID. For generating a content UID we use a meaningful name for the corresponding content or information and apply a 92 bit hash function. The Ext 1 field is used for sub-informations, which can be e.g. different articles of a newspaper or different songs of a music album. Ext 2 is used to access different versions of the content. If Ext 2 is zero, the current version is returned.

D. Identifiers for Persons

With the emergence of social networks, internet capable devices, VoIP, etc. the need for personal IDs arose, as the *person* is in the focus of interest and not the host or content. The main part of a persons UID consists of a 96 bit hash function of the persons full name. However, as more people will have the same name, we use a random value for Ext 1 when generating the UID. Ext 2 is used to specify the desired communication channel to the corresponding person. Thereby, according to the value of Ext 2, different locators will be returned which can refer to a mailbox, a cell phone or an account in a social network. The value zero for Ext 2 can be used to get the locator of the machine the person is currently working on and the desired communication channel can be signaled in a higher layer.

IV. Search Mechanism

As the main part of any UID consists of a hash function, the desired identifier can only be found if the plain text search string is exactly known and no spelling mistake occurred. We therefore additionally introduce the n-gram based search to the pure UID search.

A. n-gram generation

The plaintext input string for the hash function is split up in substrings of length *n*. With $n = 3$, i.e. the hostname `google` e.g. is split up into four trigrams: $\texttt{goo}, \texttt{oog}, \texttt{ogl}, \texttt{gle}$. Additionally to the hash value of the hostname, also each n-gram h_i of the hostname is stored in the mapping system. The n-gram entry thereby only consists of the tuple $\langle H(h_i); hostname \rangle$ [4] [5]. Whenever the host changes its location, no updates of the n-grams are necessary, as they only contain the plain hostname, which is just another representation of the UID.

B. Querying UIDs

In the first step the mapping system is always queried with the complete UID. If no mapping entry is found because of a spelling mistake e.g. it is up to the querying user to generate the corresponding n-grams and re-query the mapping system. The mapping system will return all n-gram entries and the user can sort the results according to their occurrence in the returned n-grams. Thus, a feature like Googles "Did you mean..." can be realized in the mapping system.

References

[1] O. Hanka, C. Spleiß, J. Eberspächer, *HiiMap: Hierarchical Internet Mapping Architecture*, ICFIN 09, Beijing, China

[2] V.P. Kafle, H. Otsuki, M. Inoune, *An ID/locator split architecture for future networks*, IEEE Communications Magazine 2010, Vol. 48/2

[3] C. Dannewitz, *NetInf: An Information-Centric Design for the Future Internet*, 3rd GI/ITG KuVS Workshop on The Future Internet, May 2009

[4] G. Kunzmann, *Performance Analysis and Optimal Operation of Structured P2P overlay Networks*, Doctoral Thesis, Technische Universität München, 2009

[5] C. Manning, P. Raghavan, H. Schütze, *Introduction to information retrieval*, Cambridge University Press, 2008