互联网动态地址命名与解析服务

朱 亮[1],徐 恪[1],冯 梅[2]
(1. 清华大学计算机科学与技术系 100084,北京;2. 中国石油勘探开发研究院,北京 100083)

摘 要: 命名解析机制作为互联网的核心组件,其固化的静态设计以及可扩展性的缺失是导致体系结构陷入僵化的主要原因. 针对网络核心层面的创新困境,提出了一种动态的命名与解析服务,允许按需设计、构建和部署多样的地址策略,灵活支撑顶层业务需求. 基于 xml 的通用地址描述规范通过精心抽象的元素属性以及灵活的组合方式,提供了对异构命名模型的统一表达和解释能力;动态解析服务基于预先配置的对象绑定拓扑,支持新型命名空间、协议实体以及解析机制的透明引入. 试验表明,通用命名与解析服务具备良好的可扩展性和时间性能,相比传统互联网地址系统,能够赋予网络核心层面的演进能力,从而为体系结构维度的创新提供良好支撑.

关键词: 命名; 解析; 通用地址描述; 动态解析服务; 互联网地址
中图分类号: TP393 **文献标识码:** A **文章编号:** 0372-2112 (2018)05-1089-06
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2018.05.010

A Dynamic Service of Internet Naming and Resolving

ZHU Liang[1], XU Ke[1], FENG Mei[2]
(1. *Department of Computer Science and Technology, Tsinghua University, Beijing* 100084, *China*;
2. *PetroChina Research Institute of Petroleum Exploration Development, Beijing* 100083, *China*)

Abstract: Naming and resolving as the core components of the Internet, the static design and the lack of extensibility is the main cause of the rigid architecture. To tackle the ossification of the core network layer, a dynamic service of naming and resolving is proposed. The service allows to design, build and deploy multiple address strategies on-demand to support the application requirements. Through the elaborate elements, the general address description based on the xml provides a unified expression and resolution of the heterogeneous naming models. Based on pre-configured binding topology, the dynamic resolving service can introduce the new namespaces, entities and resolution mechanisms with transparency. Test results show that the service is provided with good performance and scalability. Compared with the traditional Internet address system, the service can promote the evolution and innovation of the Internet architecture.

Key words: naming; resolving; general address description; dynamic resolving; Internet address

1 引言

随着业务模式日新月异的发展,传统互联网架构的稳定性已逐渐演变成封闭性,网络核心层过于单一的功能难以适配多样化的应用需求. 而基于 IP 地址的命名与解析策略作为路由和网络传输的核心承载机制,自设计之初就存在固有缺陷,如缺少安全机制[1]、空间枯竭、难以适应泛在的移动性和异构性等等. 研究表明[2],传统命名解析服务扩展能力的缺失是导致整个体系结构陷入创新僵化的根本原因,主要体现在以下几个方面:

(1)网络核心机制难以扩展:由于传统互联网静态的命名寻址设计,创新仅局限于应用和网络技术,核心地址组件则难以实现创新. 比如异构地址类型以及新型映射机制难以被静态体系结构所兼容,从而导致网络"窄腰"的演进陷入停滞.

(2)应用与网络紧密耦合:传统协议栈直接将 IP 地址这一网络层的概念传递到应用层,导致了应用与地址协议形成"早绑定"的耦合关系,难以支持 ILSA 等新型地址交互范式.

收稿日期: 2017-04-07;修回日期: 2017-07-28;责任编辑: 郭游
基金项目:国家自然科学基金(No. 61170292, No. 61472212);国家科技重大专项(No. 2015ZX03003004);国家 863 高技术研究发展计划(No. 2013AA013302, No. 2015AA015601);国家 973 重点基础研究计划 (No. 2012CB315803);欧盟 CROWN 基金(No. FP7-PEOPLE-2013-IRSES-610524);清华信息科学与技术国家实验室(筹)学科交叉基金

(3)服务粒度过大:互联网体系结构仅仅为上层业务提供基于 IP 的粗粒度服务,地址语法语义、解析算法缺少灵活定制,地址策略无法针对特定的业务需求进行优化.

为解决上述问题,学术界提出了大量的新型体系结构方案[3,4],但大多倾向于沿用传统互联网的静态模型,且不论能否得到实际部署,仅从方法论上看,难以适应未来多变而又不可预测的应用模式. 随着“design for change”等设计理念的出现,研究工作开始从体系结构定制和共存的角度入手,通过增加互联网策略机制的多样性来消除创新僵局[5-7]. 然而其目标通常是从宏观架构层面阐述互联网的可定制性,存在设计空间庞大、功能耦合复杂等问题,导致无法聚焦核心组件及通用协议的具体设计.

事实上,命名与解析服务动态可扩展性决定了体系结构其它组件的演进能力[8],未来互联网应能够支持多样的命名解析机制并存,按需裁剪、灵活定制以适应不同的通信视图. 比如对于功耗敏感的应用,可以按需定制轻量级的解析策略;对于语义分离网络,能够灵活定义标识映射模型;而对于安全性严格的专用网络,则能够通过加密的命名空间集成附加安全选项. 换句话说,作为体系结构核心组件的命名与解析系统应当能够自我演化,动态、灵活地支撑顶层业务需求.

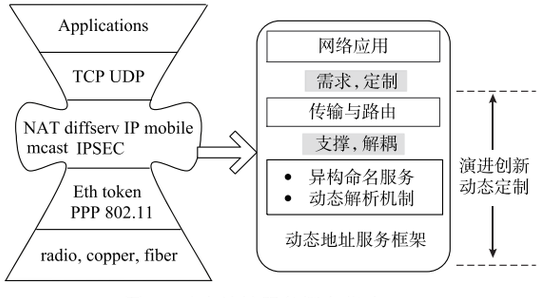

图 1 动态地址服务概念范畴

鉴于上述分析,本文旨在提出一种作为体系结构演化核心的动态命名与解析服务框架(如图 1),允许按需设计、构建和部署多样的地址策略,异构的命名解析机制能够在该框架内灵活定义,以容纳网络内在机制的创新. 同时,动态性也意味着一定程度的松耦合与开放性,有利于将命名解析与路由传输进行分离,并激励网络层面的技术竞争. 需要说明的是,本文提出的服务框架并非是一系列设计组件或者功能集合,而是一种能够容纳这些异构组件的通用平台,赋予地址系统自适应能力,从而为体系结构维度的动态性和可变性提供良好支撑.

文献[9]作为作者前期研究工作,从架构层面提出了一种通用地址框架,允许应用按需检索、创建多样的地址范式,然而并没有对具体的机制设计进行介绍. 本文作为其延续和扩展,重点从命名与解析服务方面阐述了动态定制能力,组织如下: 第 2 节提出了一种能够解释异构命名模型的统一描述规范;第 3 节基于对象绑定拓扑,阐述了动态解析服务机制;第 4 节对服务框架的性能进行了试验分析.

2 基于 xml 的通用命名服务

针对当前 IP 地址存在的缺陷,越来越多的地址命名类型被提出,其设计特征相对于 IPv4 都进行了相应的扩展或创新,比如基于加密、截取、置位等一系列操作而构造地址的 CGA[10]、T:P:L 层次化命名格式以及 HIP 的 hash(public_key)[11]公钥产生机制等等. 当前互联网中的对象标识大都通过固定的映射机制进行解析,而上述异构命名方式的语法语义存在很大区别,无法被静态体系结构所理解,从而难以引入命名机制的创新.

2.1 命名空间与命名

命名服务发生在网络对象(接口、服务、内容等)的多个层面,例如对象名字与其附着点的命名空间分别被称为名字空间(name space)和地址空间(address space). 本文将地址空间视为命名空间的子集,对二者不做严格区分. 为更加准确阐述,可进行如下定义:

命名空间可表示为 $\{<\mathrm{obj},\mathrm{symbols}>\}$ 的集合,obj 决定了命名空间语义上的概念范畴,比如 IP 和 HTTP 则分别属于不同的语义范畴;symbols 则体现了语法上的表现形式. 引入以下集合概念,可命名对象 $\mathfrak{O}$,属于特定语义范畴的对象 $\mathfrak{S}\mathfrak{O}$,任意符号集合 $\mathfrak{S}$,其中 $\mathfrak{S}\mathfrak{O}\in\mathfrak{O}$. 则命名空间可表示为:

$$\exists\,\mathrm{obj}:\mathfrak{O},n:\mathfrak{S}(\mathrm{obj}n\ \mathfrak{S}\mathfrak{O}(\mathfrak{obj}\ \mathfrak{N}\ n)\ \in\mathfrak{S}\mathfrak{O}\ x\boldsymbol{S})$$
$$\{\mathrm{obj}\mathfrak{N}n\mid\mathrm{obj}\in\mathfrak{S}\mathfrak{O}\wedge n\in\mathfrak{S}\}\qquad(1)$$

2.2 通用命名描述规范

为保证异构地址类型的设计通用性以及可理解性,本节提出了一种基于 xml 的通用地址描述规范(Generic Address Description Language,GADL),通过精心抽象的元素属性以及灵活的组合方式,提供对地址模型的充分表达和解释能力. 由于 xml schema 的可扩展性,GADL 除了能够准确描述现有模型,也能自定义或扩展命名策略并动态引入,使服务框架能够理解任意地址类型. 另外,对于复杂命名机制,如 ICN 以及基于密码学的地址产生策略,GADL 还集成了一些灵活的 metadata 集合,如 hash 算法,加密算法等,供解析时调用. 为了便于对规范进行说明,我们定义任意符号集 $\boldsymbol{\Sigma}$,其非空幂集 $\mathbb{P}_1\boldsymbol{\Sigma}==\{S:\mathbb{P}\boldsymbol{\Sigma}\mid S\notin\varnothing\}$,*nametoken* 是基于该符号集的任意字符串,即 $nametoken\in\mathbb{P}_1\boldsymbol{\Sigma}$,可以对名字的语法结构进行一般性描述:

$$name = (nametoken \ \| \ ) * nametoken \quad (2)$$

其中’‖’表示通用的一元拼接符,*表示括号内的字符串可以重复. 以(2)式为基础,图 2 说明了 GADL 的部分描述规范,主要涵盖以下核心属性:

●addr_name:地址类型,自定义名称,以 < id-name > 的形式与命名空间相关联,其中 *id* 为命名空间的唯一标识.

●addr_list:原始地址串,地址的基类型链表. 对于 IPv6 地址是 HEXBinary 类型,而对于 HIP、AIP 等,则是其对应的公钥;另外还可以使用 restriction 元素对其长度,格式加以限制,例如对于 IPv6 地址来说,可对上述规范进行扩展,使用正则表达式对地址格式进行限制:

```
< xs: simpleType >
  < xs: restriction base = "xs: hexbinary" / >
    < xs: maxExclusive value = "255" / >
    < xs:pattern value = "([A-Fa-f0-9]{1,4}:){7}" / >
</xs: simpleType >
```

●addr_struct:地址的结构,比如扁平、层次或是混合型结构;例如在 AIP 的体系结构中每个网络单元划分成一个或者多个责任单元 Ads(accountability domains),每一个 AD 都拥有全球唯一的 ID 号,为网络所在域的公钥 Hash 值;而 AD 中的每个主机唯一的终端号 EID,则是相应主机公钥的 Hash 值(其中 AD,EID 均为 160 位). 地址就被表示为 AD1:AD2:…EID 的形式.

●addr_blocks:地址块的数目,默认值为 1,即地址扁平不分块;ICN 标识研究人员可根据需求自由指定地址分块数量,诸如 NDN 中“/parc/videos/WidgetA. mpg”或者 Netinf 的“T:P:L”命名形式,以验证不同机制的解析或查询聚合性能;addr_prefix:地址的网络前缀长度;

●separator:指定地址块拼接的分隔符如‘/’或者‘.’,用以构建完整的语法形式;

●cons_method:地址的构造方法,与 metadata 集合或动态链接库 addrlib 相链接,允许安全、移动性机制的构造,使用“CDATA”标记描述. 比如对于 HIP 的 HI 地址类型则能够集成密钥机制:

```
< const_method >
  < ! [CDATA[ addrlib. MD5( pub_key) ] ] >
</cons_method >
```

3　基于配置链的动态解析服务

传统互联网不同层次实体间的关联是通过补丁式的固定解析机制(如 ARP、DNS)实现,若要在体系结构中引入一个新的协议层则必须破坏原有的解析关系链,导致新协议层及实体无法被识别. 比如在 TCP 与 IP 层之间增加了新的身份标识层,同时也必须引入新的全局映射系统来解析实体间的绑定关系. 但是由于该

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="address_name" type="Address_Type"/>
    <!—定义该地址类型的名称 —>
    <xs:element name="address_list" type="Address_List_Type"/>
    <!—该地址类型的原始字符串 —>
    <xs:element name="address_structure" type="Address_Structure_Type"/
    <!—定义该地址类型的结构 —>
    <xs:element name="address_blocks" type="xs:integer" default="1"/>
    <!—该地址类型的地址块数目 —>
    <xs:element name="address_prefix" type="xs:integer"/>
    <!—该地址类型的前缀长度 —>
    <xs:element name="separator"/>
    <!—定义地址块之间的分隔符 —>
    <xs:element name="construct_method" type="Construct_Method_Type"/>
    <!—定义该地址类型的构造方式 —>
    <xs:complexType name="Address_Type">
        <xs:sequence>
            <xs:element ref="address_list"/>
            <xs:element ref="address_structure"/>
            <xs:element ref="address_blocks"/>
            <xs:element ref="address_prefix"/>
            <xs:element ref="separator"/>
            <xs:element ref="construct_method"/>
        </xs:sequence>
    </xs:complexType>
    <xs:simpleType name="Address_List_Type">
        <xs:list>
            <xs:simpleType>
                <xs:restriction base="xs:string"/>
            </xs:simpleType>
        </xs:list>
    </xs:simpleType>
    <xs:simpleType name="Address_Structure_Type">
        <xs:restriction base="xs:token">
            <xs:enumeration value="flat"/>
            <xs:enumeration value="hierarchical"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="Construct_Method_Type">
        <xs:restriction base="xs:token">
            <xs:enumeration value="hash"/>
            <xs:enumeration value="encrypt"/>
        </xs:restriction>
    </xs:simpleType>
</xs:schema>
```

图 2　GADL部分描述规范

映射系统破坏了原有解析流,不为体系结构兼容,这也是导致新型体系结构无法增量部署的根本原因.

3.1　解析序列化

解析序列化是地址体系中关键的设计原则之一,其含义是指在解析名字 n 的过程中,通过控制解析关系 $U(n)$ 既能直接将将名字解析为可路由标识,也能够选择解析到中间多个服务代理 $U_i(n)$. 其中,中间服务代理可由体系结构按需设定,提供相应的角色或功能,从而最终将名字 n 解析为一个含多个服务代理的对象序列,即:

$$\begin{aligned} U_s(n) &= s', s' \in \{s \mid \exists < n, s > \in R_{U \times S}\} \\ &= U_{s1}(n) \cdot U_{s2}(n) \cdots U_{si}(n), m > 1, 2 \leqslant i \leqslant m \end{aligned} \quad (3)$$

从式(3)可以看出,解析发生在体系结构的多个层面的对象之间,是对象绑定关系的逆向遍历过程. 绑定事先定义了不同对象的引用关系 $\{ < obj_x, n_x > \longmapsto < obj_y, n_y > \}$,形成了特定的绑定拓扑,而解析实际上就是遍历绑定拓扑的映射迭代. 在这一设计原则指导下,解析可看作是与绑定具有相同基数的逆关系,是绑定有序对的转置矩阵. 由上述分析,可对 $< obj, n > \in \mathfrak{S}\mathfrak{O}$ 的解析过程 $\mathfrak{A}(< obj, n >)$ 定义如下:

$$\begin{aligned} \mathfrak{A}(< obj, n >) &= \beta(< obj_1, n_1 > , e_1) \\ &\quad \beta(< obj_2, n_2 > , e_2) \cdots\cdots \\ &\quad \beta(< obj_k, n_k > , e_k) = \beta^k \end{aligned} \quad (4)$$

其中,$\beta(< obj, n > , e)$ 代表一次映射,返回指向下一个语义范畴的对象引用 e,单次映射关系通过 k 次复合获

取最终用于转发的标识符. 对于任何一个待解析的 $<obj,n>$ 来说,经过上述过程,若 $e_k = null$ 则解析成功执行并终止;如果 $e_k = error$ 则表明 $\beta(<obj_k,n_k>)$ 没有被绑定关系定义,解析过程因为出错而中止. 当且仅当满足上述两个条件之一,解析过程才真正结束.

3.2 基于配置的动态解析链

在解析序列化原则指导下,设计通用解析服务的关键在于预先构造一张网络对象间的动态解析关系表(dynamic resolution table,DST),存储对象间的绑定引用关系 e,而解析过程则能够基于此配置表逆向动态实现. 另外,动态解析服务除了能够提供已有的解析功能,还允许自定义解析次序以及相应机制,以指明下一步解析行为.

图 3 说明了新的协议对象和解析机制如何透明引入已有体系结构中. 体系结构层面维护 < 对象索引、映射位置、映射机制 > 的全局绑定关系表,用以存储对象之间的引用、解析机制和解析位置. 对于传统互联网基于 lookup-by-name 的解析范式来说,分组转发之前,URL 等应用标识需要被解析成可路由地址,这种解析严格按照 <DNS→ARP> 的顺序执行,打破这种固化的解析流引入一种新的解析机制无疑是非常困难的. 而本节提出的动态解析服务是以 indirection 的方式首先查找 DST,通过对象索引确定下一步的解析机制和位置,形成运行时的动态解析服务. 以 ILSA 体系结构为例,待解析标识首先以协议对象 URL 为索引查询 DST,继而获取其对应的解析机制以及所在位置 <IDMS,DNS>,意味着对 URL 对象的解析是在标识映射服务器(ILMS)中通过 DNS 机制完成. 之后再以 ID 为索引查找 DST 获取后续的解析位置和机制,上述过程经过迭代而形成{<IDMS,DNS> → <RH,DHT> → <LOCAL,ARP>}的完整解析链,返回最终的解析结果. 对 route-by-name 的地址交互机制来说,则通过查找 DST 获取 ITR 中的 FIB 表完成名字与转发端口的映射. 这种“晚解析”的灵活性决定了可以随时在地址体系中引入任何所期望的协议实体和解析机制,而无需被体系结构所感知,在解析层面实现地址体系的通用性. 从上述过程可以看出,动态解析服务允许地址体系动态引入命名空间、协议实体以及对应解析机制,在对应用透明的同时保证原有体系结构的兼容性和可理解性,对网络核心机制的创新具有重要意义.

4 服务性能分析

命名解析服务框架的动态性不仅表现为异构命名机制通用的表述规范,同时也体现在动态解析机制的灵活定义. 其中动态解析表只需在初始阶段查找一次,即“一次解析、处处运行”,其占用的少量开销在整个地

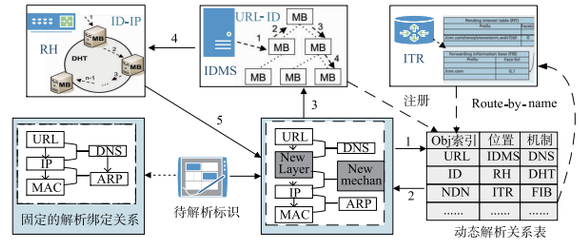

图 3 基于配置的动态解析链

址交互过程中基本可以忽略. 因此,服务性能开销主要集中在 GADL 描述规范解析的时间特性上. 图 4 说明了针对 GADL 规范的通用解析算法示例和构造过程,主要包含以下几个时间变量:*Treq* 表示节点向服务器请求获取标识的时间;*Tpar* 为 GADL 描述文件的解析时间;*Trpc* 代表链接查找 addrlib 中对应接口的时间;而 *Tcons* 则是使用相应构造方法产生最终地址的时间.

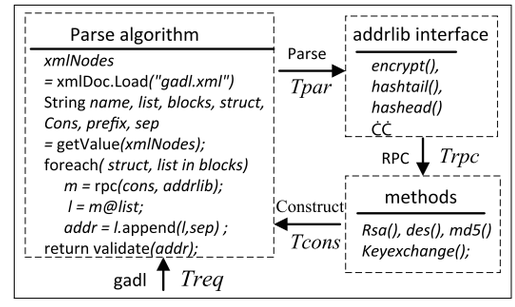

图 4 GADL解析算法及流程示例

令一般处理方式下(记为□G),即不使用 GADL 而采用传统地址分配机制(如 DHCP),从请求标识到完成分配的时间为 T,而基于通用描述规范(记为 G)的处理时间为 $T*$,则对于 n 个节点的地址分配请求来说,命名服务完成时间可由式(5)定义. 可以看出,尽管 $T*$ 比 T 多出了解析时间 *Tpar*、和调用时间 *Trpc*,但由于 GADL 的通用性以及批量处理机制,命名请求 *Treq*,规范解析 *Tpar* 和方法调用 *Trpc* 仅需在初始化时执行一次,而基于传统地址池的技术则需针对每个请求单独占用 *Treq* 和 *Tcons*,因此对于大量节点的并发请求,GADL 解析实际上具备更为显著的优化能力. 为验证上述通用命名服务机制的性能,本节通过将 GADL 命名服务封装为 *webservice*,分别模拟了 IPv4,HIP[11],CGA[10] 三类典型地址机制的标识分配过程,其中客户端不同类型节点在不同的数量规模下以服务调用方式发送请求,并分别针对对其时间性能以及变化趋势进行了分析,实验结果如图 5 所示.

$$\begin{cases} T = \sum_{i=1}^{n} (Treq_i + Tcons_i) \\ T* = Treq + Tpar + Trpc + \sum_{i=1}^{n} Tcons_i \end{cases} \tag{5}$$

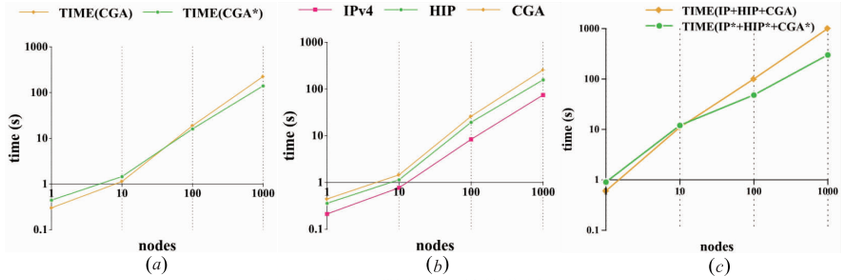

图 5　服务时间性能分析

图(*a*)以 CGA 地址类型为例,说明了分别在□*G* 和 *G* 的情况下,请求节点数量与服务完成时间的变化关系. 当请求节点数量很少时,描述文件的解析时间占据总时间的比例较大,因此 TIME(CGA)明显小于 TIME(CGA*);然而随着请求数量的增长,GADL 通用解析算法的优势则得到了体现,在节点规模达到 10^2 数量级的时候,TIME(CGA)已经超越 TIME(CGA*),并以更高的变化率持续增长. 图(*b*)模拟了上述三种地址类型在 GADL 描述规范下的时间性能. 由于 HIP、CGA 基于密钥机制构造,存在大量的密钥交换、分配时间,因此相对与 IP 来说,*Trpc*、*Tcons* 有着更大的时间开销. 同理,CGA 地址的构造机制相对 HIP 更为复杂,导致 *Tcons*(*CGA* *) > *Tcons*(*HIP* *). 从总体上看,GADL 命名服务的完成时间均在合理的阈值范围内. 图(*c*)展示了更为一般的地址分配场景,IP、HIP、CGA 节点以 10、10^2、10^3 的数量规模同时请求命名服务. 在该情况下,传统命名机制需要对这三种地址类型分别设计构造和分配算法,从图中可以看出,命名完成时间基本处于线性增长状态;而对于本文提出的通用描述和解析算法而言,则能够同时处理上述请求,*Treq*、*Tpar*、*Trpc* 随着节点数量增长而趋近于 0. 因此相对传统地址分配机制来说,GADL 在多类型、大规模节点规模下,具备更加优秀的时间性能.

5　结束语

本文聚焦于命名解析这一互联网核心组件,提出了一种动态的地址命名与解析服务,支持在体系结构中动态引入命名空间、协议实体以及对应解析机制,并在对应用透明的同时保证原有体系结构的兼容性以及可理解性,有助于地址系统的演进发展. 应用实验表明,命名与解析服务具备良好的可扩展性和时间性能,能够为体系结构维度的演化创新奠定良好基础.

参考文献

[1] 徐恪,朱亮,朱敏. 互联网地址安全体系与关键技术[J]. 软件学报,2014,25(1):78 – 97.
Xu Ke, Zhu Liang, Zhu Min. Architecture and key technologies of internet address security [J]. Journal of Software, 2014, 25(1): 78 – 97. (in Chinese)

[2] 朱敏,徐恪,林嵩. 面向应用适应能力的互联网体系结构评估方法[J]. 计算机学报,2013,36(9):1785 – 1798.
ZHU Min, XU Ke, LIN Song. The evaluation method towards the application adaptability of internet architecture [J]. Chinese Journal of Computers, 2013, 36(9): 1785 – 1798. (in Chinese)

[3] Rexford J, Dovrolis C. Future Internet architecture: clean-slate versus evolutionary research. [J]. Communications of the Acm, 2010, 53(9): 36 – 40.

[4] Xu K, Zhu M, Hu G W, et al. Towards evolvable Internet architecture-design constraints and models analysis[J]. Science China Information Sciences, 2014, 57(11): 1 – 24. .

[5] 刘强,汪斌强,徐恪. 基于构件的层次化可重构网络构建及重构方法[J]. 计算机学报,2010,33(9):1557 – 1568.
Liu Qiang, Wang Binqiang, Xu Ke. Construction and reconfiguration scheme of the hierarchical reconfiguration network based on the components [J]. Chinese Journal of Computers, 2010, 33(9): 1557 – 1568. (in Chinese)

[6] Niebert N, Baucke S, EL-Khayat I, et al. The way 4WARD to the creation of a future internet[A]. Proceedings of the International Symposium on Personal, Indoor and Mobile Radio Communications [C]. Cannes, French Riviera, France, 2008. 1 – 5.

[7] Han D, Anand A, Dogar F, et al. XIA: efficient support for evolvable internetworking[A], Proceedings of the Usenix Conference on Networked Systems Design and Implementation[C]. SAN JOSE, CA, USA, 2012. 817 – 826.

[8] Choi J, Park C, Jung H, et al. Addressing in future Internet: problems, Issues, and Approaches[A]. Proceedings of the 3rd International Conference on Future Internet Technologies[C]. 2008.

[9] 朱亮,徐恪. 互联网通用地址体系框架[J]. 西安交通大学学报,2017,51(2):8 – 16.
Zhu Liang, Xu Ke. A general framework of internet address mechanisms [J]. Journal of Xi'an Jiaotong University, 2017, 51(2): 8 – 16. (in Chinese)

[10] T Aura. Cryptographically generated addresses (CGA).

RFC3972[EB/OL](2005-03-12).[2016-07-22].https://datatracker.ietf.org/doc/rfc3972.
[11] Moskowitz R,Hirschmann V,Jokela P,et al. Host identity protocol version 2(HIPv2). RFC 7401[EB/OL](2013-12-14).[2016-07-26].https://datatracker.ietf.org/doc/rfc7401/.

作者简介

朱 亮 男,1982 年出生,清华大学计算机系博士研究生,感兴趣的领域有互联网体系结构及其性能评价.

徐 恪 男,1974 年出生于江苏,2001 年获得清华大学工学博士学位,同年留校任教,现任清华大学计算机系教授、博士生导师. 研究领域包括新一代互联网体系结构、高性能路由器体系结构与路由协议、P2P 与 Overlay 网络、物联网等.
E-mail:xuke@csnet1.cs.tsinghua.edu.cn