ARPIM: IP Address Resource Pooling and Intelligent Management System for Broadband IP Networks

Chongfeng Xie, Jun Bi, Heng Yu, Chen Li, Chen Sun, Qing Liu, Zhilong Zheng, and Shucheng Liu

Abstract

IP address resources work as basic elements for providing broadband network services. However, the increase in number, diversity and complexity of modern network devices and services creates unprecedented challenges for current manual IP address management. Manually maintaining IP address resources could always be sub-optimal for IP resource utilization. Besides, it requires heavy human efforts from network operators. To achieve high utilization and flexible scheduling of IP network address resources, this article introduces APRIM, an innovative SDN-based IP address pooling and intelligent management system, in which we design a centralized address management system to realize dynamic allocation, reclaim, and reallocation of address blocks for the current BRAS/vBRAS deployment. We developed a prototype system and evaluated the system based on real-world networks and users in two provinces from China Telecom. Experimental results demonstrate that our system can largely improve the address utilization efficiently and reduce the network resource maintenance workload.

Introduction

As a typical large-scale information network system, broadband IP networks contain and maintain various resources, including IP address, link bandwidth, forwarding capacity, cache, session resources, and so on, which work as the basic elements to provide broadband services. Resource management is one of the key processes of network operation. Allocating resources in a timely manner to meet the needs of different categories of customers and achieving global optimal resource allocation efficiency at the same time within limited resources have always been the goals of network resource management.

Meanwhile, network operators are devoting increasing attention to IP address management, since IP addresses are the primary resources to provide connection and services on the broadband Internet. In most current cases, the IP address management system lacks an automated control mechanism. For instance, the address system integrated in broadband remote access servers (BRASs) is configured statically via command line interface (CLI), and the management of IP addresses is purely artificial. Network operators manually allocate IP addresses when they are exhausted in a BRAS. Some users might have to wait until new IP blocks are assigned to the BRAS. Therefore, the timeliness of address allocation cannot be guaranteed.

Moreover, the increase in number, diversity and complexity of modern network devices and services bring new challenges for the management of IP addresses in new IP networks.

1. The efficiency of manual assignment is often sub-optimal. Real-world address resources are often managed across multiple, partly disconnected systems. Different systems lack timely interaction about the usage of the addresses, leading to the situation where one network element falls short of IP addresses while another one possesses redundant addresses. Manual resource management could cause slow scheduling and reduce the efficiency of resource utilization.

2. The address configuration burden of network operators based on network elements could be non-trivial and heavy. IP address resources for various network systems need to be adjusted quickly due to frequent user and traffic dynamics. Besides, IPv6 transition technologies create the need to control and share addresses among entities. Addresses of different network slices should be configured on each transition instance for high availability (HA) support. Therefore, resource utilization of network systems could change very quickly. However, the current IP address management system depends on manual management and configuration, and lacks an open programmable interface for automatic IP resource management, which leads to a heavy maintenance burden and slow response to dynamics.

3. Inefficient and trivial manual management leads to serious fragmentation of IP addresses. IP address resources no longer consist of large blocks of consecutive addresses, but a randomly scattered set of many small blocks or even independent individual addresses. The granularity of the address distribution is often as trivial as /23, /24. Such fragmentation further decreases

To achieve high utilization and flexible scheduling of IP network address resources, the authors introduce APRIM, an innovative SDN-based IP address pooling and intelligent management system, in which they design a centralized address management system to realize dynamic allocation, reclaim, and reallocation of address blocks for the current BRAS/vBRAS deployment.

Chongfeng Xie, Chen Li, and Qing Liu are with China Telecom Co. Ltd.; Jun Bi, Heng Yu, Chen Sun, and Zhilong Zheng are with Tsinghua University and Tsinghua National Laboratory for Information Science and Technology.; Shucheng Liu is with Huawei Technologies Co. Ltd.

Digital Object Identifier: 10.1109/MCOM.2017.1601001

 0163-6804/17/$25.00 © 2017 IEEE

To improve the utilization efficiency of IP resources and reduce overall operating expense (OPEX)/capital expenditure (CAPEX) at the same time, operators have been looking for a more intelligent, agile, and flexible approach to control and manage IP address resources.

resource utilization efficiency and complicates manual management. Without open programmable interfaces and automated control, the clustering of small blocks or single IP addresses is difficult to realize. We have summarized the detailed problem statement in [1].

The problems of manual monitoring and management of networks have been recognized by the industry community. Huawei Technologies Co. Ltd. proposed the SDN-Based Refined O&M to achieve nanosecond-level service quality detection in data centers. Reference [2] proposed a communication method, communication system, resource pool management system, switch device, and control device to better utilize SDN flow tables and controller resources. However, no previous efforts addressed the challenges of the IP address resource monitoring and management.

To improve the utilization efficiency of IP resources and reduce overall operating expense (OPEX)/capital expenditure (CAPEX) at the same time, operators have been looking for a more intelligent, agile, and flexible approach for controlling and managing IP address resources. Assignment of such resources should work across multiple services, support flexible allocation, reclaiming, and reallocation capabilities, support various network elements such as BRAS, virtual BRAS (vBRAS), carrier-grade network address translation (NAT) (CGN), and firewalls, and support different types of addresses including IPv4 public/private network and IPv4/IPv6 addresses.

In this article, based on real-world Internet service provider (ISP) requirements from IP networks, we abandon the traditional manual and distributed configuration of IP address resources, exploit the benefit of software defined networking (SDN) [3] central control, and propose the Address Resource Pooling and Intelligent Management (ARPIM), a centralized IP address resources pooling and intelligent management system, to automatically allocate and revoke address resources. ARPIM migrates traditional manual IP resource management to centralized, programmable, and automated scheduling to increase the flexibility of address resource allocation. It maintains a centralized address resource pool and monitors the IP resource utilization of each network element, based on which it dynamically allocates or revokes addresses to achieve optimal resource scheduling.

This article makes the following contributions:

- We conclude the scenarios for address resource management in ISP networks and identify design requirements for the IP address resource pooling and management system to guide the system design.
- We propose ARPIM, an SDN-based centralized system, to address the design requirements by providing flexible, automated, and optimal management of IP address resources.
- We evaluate ARPIM with extensive experiments based on real ISP networks and users from China Telecom. Our experimental results show that ARPIM can improve IP resource utilization efficiency to a large extent while automating IP address resource management and reducing manual work.

Scenarios and Design Requirements of Address Management

We list some commonly seen scenarios for IP address resource management in ISP networks.

IP Allocation for BRASs/vBRASs: BRASs/vBRASs require pre-configured IPv4 and IPv6 address resources to allocate IP addresses to users through Dynamic Host Configuration Protocol (DHCP)/Point-to-Point Protocol over Ethernet (PPPoE) at the edge of IP networks. Generally, to meet the needs of different types of services and customers, each BRAS/vBRAS would have multiple local address pools and require timely allocation of IP addresses in order to address user dynamics.

IPv6 Transition: In order to meet the needs of different transition scenarios, networks often deploy more than one transitional technology as well as remaining redundant backups. IPv6 transition mechanisms (e.g., DS-Lite [4], Lw4over6 [5]) need to configure address pools, which are used as translated routable addresses. A centralized address management entity should be provided among different transition instances. In the early IPv6 transition stage, technologies such as NAT444 could occupy a larger proportion of addresses, while in the latter stage DS-Lite and Lw4over6 will require a larger proportion of addresses.

IP Allocation for Third Party Systems: Systems such as OSS and OpenStack [6] should be able to acquire IP addresses from the IP address allocation system through RESTful application programming interfaces (APIs) to allocate IP addresses to hosts under their management. Besides, in network functions virtualization (NFV), management systems (e.g., OpenStack) will also require addresses from the IP address allocation system in a RESTful manner. Therefore, the system should be able to provision such services in cooperation with Domain Name Service (DNS) or DHCP servers.

Based on the above cases, we illustrate our insights on the design requirements of the IP address pooling and management system.

•In order to obtain optimal efficiency of address resource allocation, we need integrated and centralized IP address management that offers an aggregated view on all stages of the life cycle of IP address resources, from selection to allocation to reclaiming.

•As address consumption in each device changes quickly over time due to changes of users, services, traffic, or session volumes, the management system should automatically gather resource utilization from devices and react dynamically.

•IP address resource management policies should adapt to a broad variety of usage scenarios and multiple types of network entities, both physical and virtual, including BRAS, vBRAS, broadband network gateway (BNG), virtual BNG (vBNG) [7], CGN, firewall, residential access network (RAN) [8], and so on.

•The IP address management system needs to handle IPv4 and IPv6 resources, networks including sub-netting, and prefixes with any valid configurable prefix lengths. All well-defined and Internet Engineering Task Force (IETF) Request for Comments (RFC) covered address types should be administrable.

•IP address management shall meet additional requirements including high reliability, availability, security, and performance, according to best practices for mission-critical infrastructure.

ARPIM Architecture

To meet all requirements, based on SDN, we design ARPIM, which primarily serves the devices on network edges. The system includes three major modules.:

- A centralized network resource pool manager that automatically performs IP address allocation and reclaiming according to network dynamics
- A centralized controller that communicates with underlying network devices through southbound interfaces such as NETCONF [9]
- Network elements enhanced with an address management agent (AMA) that monitors local resource usage and communicates with the centralized controller.

In addition, we also designed southbound interfaces based on the NETCONF/YANG [10] model for the controller to issue address policies and gather resource utilization status. The architecture of the system is shown in Fig. 1. The function of each module is elaborated in the following sections.

Network Resource Pool Manager

The network resource pool manager (later referred as "Resource Manager") maintains a global IP address pool in a database, and the online information of all network elements in the vBRAS management module. As shown in Fig. 2, to achieve intelligent scheduling of resources, it gathers the information of device deployment and IP address consumption of network elements in a centralized manner in the address inquiry module, dynamically decides address allocation or reclaiming strategies in the address allocation module according to device address usage ratio, and issues the policies to the controller through the controller management module. We design user interfaces (UIs) for the Resource Manager to display address utilization status and statistics information to the administrator. Besides, in order to expose the capability of address allocation for third party systems including OSS and OpenStack, the Resource Manager expose the address management capabilities through RESTful interfaces. Finally, we design southbound interfaces through which the Resource Manager communicates with the controller to issue policies, acquire address usage, and allocate or reclaim address blocks in a timely manner.

Centralized Controller

The ARPIM controller collects address utilization status of network elements through southbound interfaces, after which it regulates different interface protocols into a standard format and reports the information to the Resource Manager through northbound interfaces. Also, the controller is responsible for distributing address related policies to each network element and converging reports from each device in order to reduce the volume of information processed by the Resource Manager. The controller is implemented based on ONOS [11], a widely used open source controller with some functional extensions including

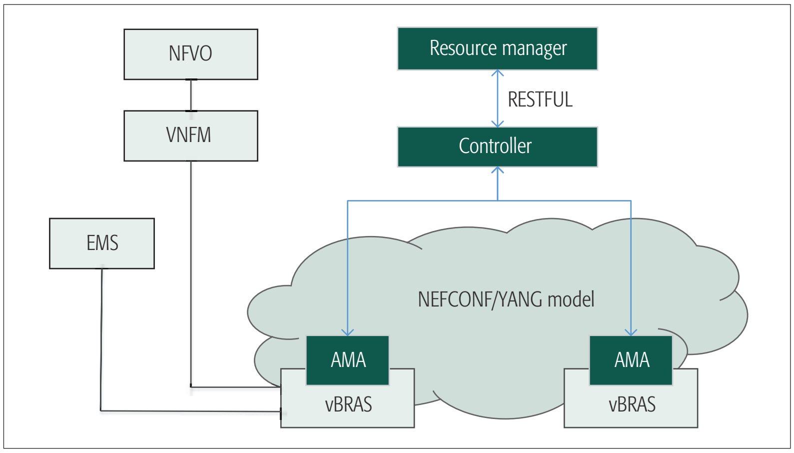

Figure 1. The ARPIM architecture.

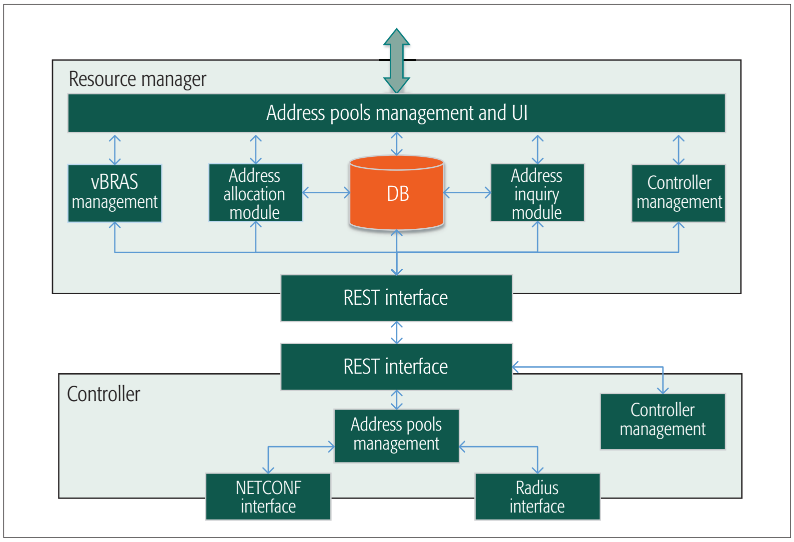

Figure 2. Architecture of the resource manager and controller.

network element address utilization status gathering and regulating, address policies distributing, and its support for various southbound interfaces. As shown in Fig. 2, we mainly implemented two types of southbound interfaces: NETCONF for virtualized network elements and Radius for hardware network elements.

Enhanced Network Elements

Enhanced network elements refer to all the equipment at the edge of the network, such as vBRASs, which are under the management of ARPIM. Each network element is extended with an address management agent (AMA) module that corresponds to the address pooling resources management functions. The AMA module receives the address allocation or reclaiming policies, collects local address utilization status, and reports the statistics to the controller regularly according to prior configurations through southbound interfaces.

ARPIM-Enhanced Network Architecture

We also consider implementing ARPIM in an NFV environment. The NFV orchestrator (NFVO) is responsible for the overall management of virtual network elements such as vBRASs in the underlying infrastructure, which is a pool of CPU, memory, storage, and other resources. The virtualized net-

Network elements calculate the utilization ratio of their own address pools, based on the proportion of the number of allocated address to the total number of address resources, and report to the controller regularly. If there is a rapid increase in the number of online users, the utilization ratio of the address pool resources will reach the pre-configured alarm threshold.

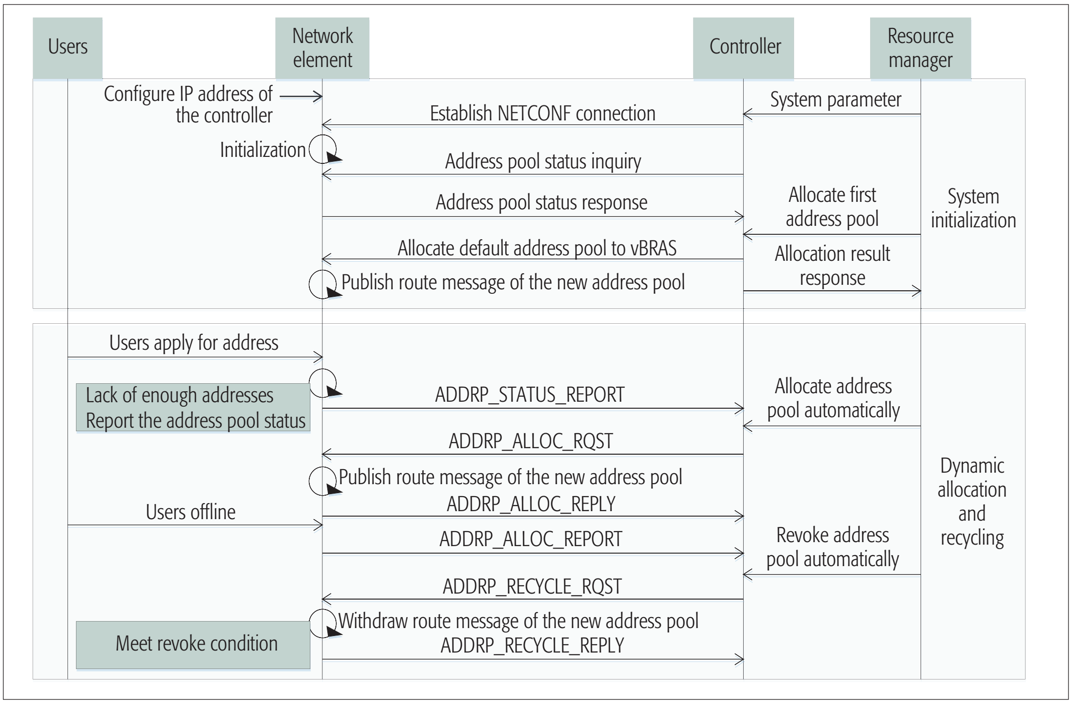

Figure 3. The workflow of centralized address resource pooling and scheduling system.

work function manager (VNFM) is responsible for life cycle and performance management of vBRAS devices, such as element instantiation, expansion, shrinking, and other functions. The element management system (EMS) is responsible for completing the main traditional new functions of the network element management tasks in the virtual environment.

Therefore, we implement the Resource Manager and the controller as functional modules in the VNFM, and implement the AMA as an extension in the EMS. In this way, we can support centralized intelligent address management in NFV networks.

Resource Manager Workflow

ARPIM migrates the address configuration from the traditional manual manner to centralized and automated scheduling to enhance the flexibility of address resource allocation. This section illustrates the workflow of the address pool management mechanism. We depict the workflow in Fig. 3. There are three major stages in the system, which are elaborated below.

Network Element Initialization

The IP address of the controller is pre-configured to network elements. When a device gets online, it establishes a NETCONF connection with the controller and applies for initial address pool resources, in the form of IP address blocks, from the Resource Manager. Then the device can allocate IP addresses to end users and broadcast the appropriate routing information. Meanwhile, the Resource Manager marks the allocated address block as occupied to avoid reallocation of the same resources.

Address Resource Allocation

Network elements calculate the utilization ratio of their own address pools, based on the proportion of the number of allocated addresses to the total number of address resources, and report to the controller regularly. If there is a rapid increase in the number of online users, the utilization ratio of the address pool resources will reach the pre-configured alarm threshold. Under detection of such a report, the Resource Manager starts an address allocation process.

First, the Resource Manager selects appropriate IP blocks from the central resource pool that can satisfy the device's requirements and allocates them to the controller with the symbol of identification and domain name of the network element. Second, the controller sends the new address resources to the specified device according to the Resource Manager's instructions. Finally, the network element that fell short of IP addresses will be able to quickly obtain new IP addresses and allocate them to the increasing number of users.

Free Address Reclaiming

In situations such as a rapid decrease in the number of online users, the utilization ratio of address pool resources will reach the reclaiming bottom threshold. The Resource Manager will start its resource reclaiming process of free address blocks. First, the Resource Manager sends the identification of the reclaimed address resources and identification of the relevant network element to the controller. Second, the controller notifies the particular device to reclaim the address resources and cancel the related routing information. Finally, a successful reclaiming is reported to the Resource Manager by the controller to change the status of the reclaimed address to "idle." In this way, the address resources can be recycled, which improves overall resource utilization efficiency across network elements even in different areas.

Decision Tree of the Resource Manager

In order to automatically manage IP address resources and obtain optimal utilization of address resource allocation, the decision tree of the Resource Manager plays a core role in the whole system. We summarize the policy decision tree in consideration of all kinds of situations in Fig. 4.

Southbound Interface Models

The design of address resource pooling and an intelligent scheduling system can accommodate both NFV virtual network elements and existing hardware network elements by developing a unified southbound interface set. Major message types between the controller and underlying network elements include regular report of IP resource utilization, IPv4/IPv6 address allocation, and free address reclaiming. We mainly introduce NETCONF YANG model-based information structure as follows.

IP Resource Utilization Report Model

```
module: ietf-address-pool-status
+--rw address-pool-status
|  +--rw address-pool* [address-pool-name]
|     +--rw address-pool-name        string
|     +--rw address-pool-id          string
|     +--rw domain-name              string
|     +--rw status              enumeration
|     +--rw address-pool-entries
|        +--rw ipv4-addresss-block* [ipv4-adress-block-name]
|        | +--rw ipv4-adress-block-namestring
|        | +--rw address-pool-id    string
|        | +--rw peak-address-usage-ratio    uint32
|        | +--rw average-address-usage-ratio uint32
|        +--rw ipv6-addresss-block* [ipv6-adress-block-name]
|        | +--rw ipv6-adress-block-name      string
|        | +--rw address-pool-id             string
|        | +--rw peak-address-usage-ratio    uint32
|  +--rw average-address-usage-ratio         uint32
```

This model describes the utilization information about IP address resources in network elements. The "address-pool-name" field describes the name of the address pool. The "status" field describes the status of the address pool as active or idle; the "peak-address-usage-ratio" describes the peak usage rate of the address block. The "average-address-usage-ratio" field indicates the average usage rate of the address block.

IPV4 Address Allocation Model

```
module: ietf-address-pools
  +--rw address-pools
   |     +--rw device-id                         int
   |     +--rw time                           double
   |  +--rw address-pool* [address-pool-name]
   |     +--rw address-pool-name              string
   |     +--rw address-pool-id                string
   |     +--rw domain-name                    string
   |     +--rw address-pool-entries
   |      | +--rw ipv4-adress-block* [ipv4-address-block-name]
   |      | | +--rw ipv4-address-block-name   string
   |      | | +--rw ipv4-address-block-id        int
   |      | | +--rw ipv4-prefix               string
   |      | | +--rw ipv4-prefix-length?          int
   |      | | +--rw user-gatewayinet:ipv4-address-no-zone
   |      | | +--rw gw-netmask       yang:dotted-quad
   |      | | +--rw type            address-pool-type
|  |      | +--rw lifetime         yang:date-and-time
   |      | | +--rw primary-dns          dns-primary
   |      | | +--rw secondary-dns      dns-secondary
```

This model is used for IP address allocation to the network elements. The "device-id" field describes the ID of the device that applied for address resource. The "address-pool-name" field describes the name of the address pool. The "ipv4-address-block-name" field describes the name of the ipv4 address block that is allocated to the device. The "lifetime" field describes the lifetime for the allocated address block, over which the device should renew its application for this address block.

Free IPV4 Address Reclaiming Model

```
module: ietf-address-pools
| +--rw address-pools
|     +--rw device-id                              int
|     +--rw address-pool [address-pool-name]
|     +--rw address-pool-name                   string
|     +--rw address-pool-id                        int
|     +--rw address-pool-entries
|     | +--rw ipv4-address-block* [ipv4-address-block-name]
|     |  | +--rw ipv4-address-block-name        string
|     |  | +--rw ipv4-address-block-id          string
|     +--rw leasing-time                           int
```

This model describes the interface information of the reclaiming process. The "device-id" field describes the ID of the device that releases address resources. The "address-pool-name" field describes the name of the address pool. The "ipv4-address-block-name" field describes the name of the IPv4 address block to be reclaimed. The "leasing-time" field describes the leasing time of the reclaimed address block.

Implementation and Evaluation

We have implemented the Resource Manager and the controller of ARPIM based on the ONOS controller, and extended the AMA module of a private software implementation of a vBRAS. Based on the Apache Karaf [12] Open Service Gateway Initiative (OSGI) framework adopted by ONOS, we implemented the Resource Manager and the controller of ARPIM as subsystem bundles and loaded them into ONOS as dynamic modules (9.2K LoC). The extended controller bundle can be further divided into three modules, including northbound interface module, core control module, and southbound interface module. The northbound interface implementation separates underlying device information from the Resource Manager. Therefore, in a situation where device failure

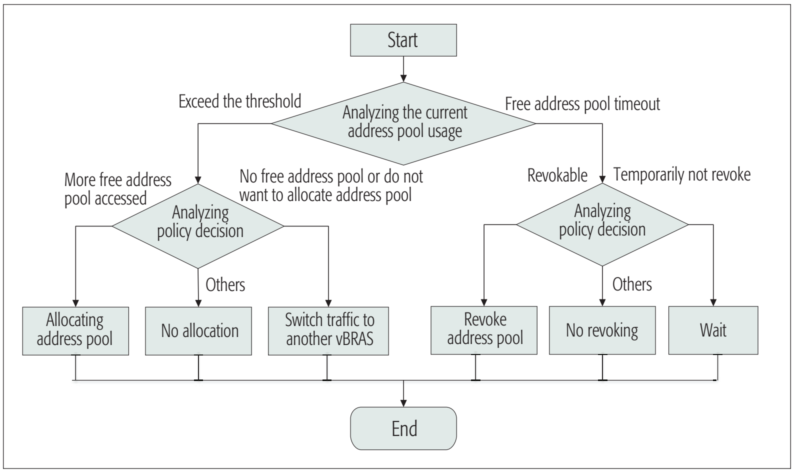

Figure 4. The decision tree of the resource manager.

(a) For traditional broadband access scenarios					
Device type	Hardware	BW (Gb/s)	Support user	BW/user (Mb/s)	CAPEX ($)
BRAS	1 box	20	16,000	1.25	15,000
vBRAS	16 CPU cores	20	16,000	1.25	27,000
(b) For enhanced networking scenarios such as NAT					
Device type	Hardware	BW (Gb/s)	Support user	Time to market	CAPEX ($)
BRAS +NAT	1 box+ 2 NAT cards	20	32,000	~Month	63,000
vBRAS +NAT	16 CPU cores	20	32,000	~Day	27,000

Table 1. Performance and CAPEX of vBRAS vs. BRAS.

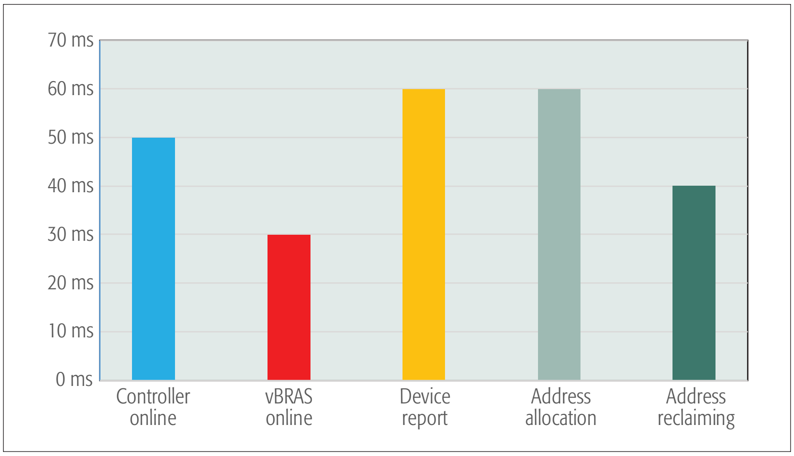

Figure 5. Time consumption of each stage in ARPIM.

occurs, the Resource Manager will not be affected. The core control module transforms message types between southbound and northbound formats. The southbound interface implementation communicates with network elements including vBRASs or BRASs through the NETCONF or Radius protocol.

For the vBRAS extension (2K LoC), based on the IETF YANG model, we designed and implemented the southbound interface extension to our original developed vBRAS software. All southbound interfaces follow RFC 6241 [9].

We deployed the system in two provinces from China Telecom, based on which we evaluated system performance with real network users.

Performance and CAPEX of vBRAS vs. BRAS

ARPIM considers NFV as an important using scenario for IP address management. Therefore, we implement ARPIM in an NFV environment in a metro-area network with vBRAS deployments. Table 1a shows the real-world performance and CAPEX of vBRAS and BRAS. To provide equally high bandwidth (20 Gb/s) and support as many users (16,000) as a BRAS, a vBRAS needs an X86-based hardware server with 16 CPU cores. The estimated total CAPEX of vBRAS is $27,000, which is 80 percent higher than hardware BRAS. Therefore, currently a vBRAS has no advantage in CAPEX compared to a BRAS with equal performance.

However, a vBRAS provides much higher flexibility and lower CAPEX for upgrading network functions and supporting new network functions. As shown in Table 1b, for some scenarios where ISPs implement a dedicated BRAS for sparse value-added services in local networks, the BRAS should be enhanced with NAT functions. In this situation, the CAPEX of vBRAS remains $18,000 with a simple software upgrade, while hardware BRAS suffers an increase of $48,000 in CAPEX due to the additional NAT cards. Therefore, the CAPEX of a vBRAS is 57 percent lower than a traditional hardware BRAS. Moreover, virtualized implementation could, to a large extent, decrease the time for new network functions to come to the market. To enhance a hardware BRAS with NAT, network operators should deploy additional hardware devices, adjust link resources, load bandwidth, and so on while virtualized implementation could be deployed with simple software upgrades.

Therefore, from the ISP's point of view, NFV could still reduce the overall CAPEX and the time for new network functions to come to market.

OPEX of ARPIM vs. Traditional Manual Manner

It is common for there to be hundreds of BRAS devices in a metro area network. Traditional manual IP address configuration of such a large number of devices will cost lots of human work and increase OPEX. During each address allocation process, the network operator has to configure many fields into the BRAS device including the IP address block name, the gateway, the address range, the DNS server for this address block, and the domain name. Besides, the operator has to manually configure related routing information about this address block. All the above human configurations of ONE address block for ONE device could cost minutes, and configuration of the entire metro area network would incur a heavy burden on OPEX. However, in ARPIM, through central IP address pooling and automated configuration, the controller online registration process, vBRAS device online registration process, device status report process, and address allocation process could each be implemented in 60 ms (Fig. 5), which could save lots of human effort and thus reduce OPEX.

In the situation where the IP resources for one device are exhausted, as all IP addresses have been allocated to other devices, the operator has to manually log in each device to check if there is a free address block, delete related address information and routing information, and then re-allocate the reclaimed address block to the exhausted device. However, in ARPIM, a BRAS device can report its IP utilization status in as little as 60 ms. The centralized Resource Manager could analyze the resource utilization of each device according to its regular report and reclaim free addresses to the resource pool in 40 ms. In this way, ARPIM could offer timely reaction for IP address exhaustion situations in seconds. This could not only reduce OPEX, but provide better service experience for network tenants.

Address Utilization Efficiency of ARPIM vs. the Traditional Manner

In traditional configuration, IP addresses are manually pre-configured to network elements. IP addresses are dedicated to related devices. The device will not release the IP address blocks even if they are entirely free. However, when the IP address resources in one device are exhausted, the operator of a traditional BRAS has two reactive approaches.

First, the operator could allocate a new address block to the device. However, if the address block in another device is entirely free, the overall resource utilization efficiency is decreased.

Second, the operator could "randomly borrow" IP addresses from a second device by migrating the entire or part of an address block from one device to the other. The size of the borrowed address block depends on how many addresses are free in the selected device. In some extreme cases, the operator has to fetch one IP address to fuel the exhausted device, which could cause serious fragmentation of IP addresses and decrease address utilization efficiency.

On the other hand, ARPIM forms a centralized shared IP address pool for all devices. The free address blocks reported by each device could be reclaimed in a timely manner, assuring quick re-allocation for devices with few IP address resources. In this way, IP address blocks are less likely to be partitioned, and the resource fragmentation situation could be alleviated.

ARPIM Scalability

We built an ONOS cluster comprising three controller instances, and measured the maximum network elements that can be supported by the cluster. Evaluation results demonstrate that the cluster could maintain 3000 sessions (i.e., network elements), which could cover a metro area network. ARPIM could support more network elements through extending the cluster size.

Conclusion and Future Work

In this article, we propose ARPIM, a centralized IP address resource pooling and intelligent management system based on SDN to satisfy real demands from ISPs. We introduce the system architecture, workflow, address resource scheduling algorithm, and interface design of this technology. Experimental results demonstrate that this technology can not only improve the utilization of IP network address resources immensely, but also reduce the address resource configuration burden of network managers. To the best of our knowledge, this is the first article by an Internet service provider about centralized IP resource pooling and intelligent management for network elements.

For future work, we will deploy this new technology on a large-scale network system to validate the feasibility and performance. We will also try to apply the centralized management approach to other network resources, including forwarding capacity, cache, and so on, to achieve higher utilization.

For the future work, we will deploy this new technology on large-scale network system to validate the feasibility and performance. We will also try to apply the centralized management approach to other network resources, such as forwarding capacity, cache and so on, to achieve higher utilization.

Acknowledgment

This research is supported by the National Natural Science Foundation of China (No.61472213 and No.61502267). Jun Bi is the corresponding author.

References

[1] X. Chongfeng *et al.*, "Problem Statement for Centralized Address Management," IETF Internet Draft, July 8, 2016, expired Jan. 8, 2017.

[2] C. Li *et al.*, "Communication Method, Communication System, Resource Pool Management System, Switch Device and Control Device," U.S. Patent App. 15/122,323, Dec. 2014.

[3] N. McKeown *et al.*, "OpenFlow: Enabling Innovation in Campus Networks," *ACM SIGCOMM Comp. Commun. Rev.*, vol. 38, no. 2, 2008, pp. 69–74.

[4] D. W. Hankins and T. Mrugalski, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Options for Dual-Stack Lite," IETF RFC 6334, Aug. 2011.

[5] I. Farrer and D. Alain, "lw4over6 Deterministic Architecture," IETF Internet Draft, Oct. 25, 2012, expired Apr. 25, 2013.

[6] O. Sefraoui *et al.*, "OpenStack: Toward an Open-Source Solution for Cloud Computing," *Int'l. J. Computer Applications*, vol. 55, no. 3, 2012.

[7] Network Function Virtualisation (NFV): Use Cases: http://www.etsi.org/deliver/etsi_gs/nfv/001_099/001/01.01.01_60/gs_nfv001v010101p.pdf, accessed Feb. 12, 2017.

[8] Cloud RAN: http://www.ericsson.com/res/docs/whitepapers/wp-cloud-ran.pdf, accessed Feb. 12 2017.

[9] R. Enns, *et al.*, "NETCONF Configuration Protocol," IETF RFC 6241, June 2011.

[10] M. Bjorklund, "YANG-A Data Modeling Language for the Network Configuration Protocol (NETCONF)," IETF RFC 7950, Aug. 2016.

[11] ONOS, http://onosproject.org/, accessed Feb. 12 2017.

[12] J. Edstrom *et al.*, *Learning Apache Karaf*, Packt Publishing Ltd, 2013.

Biographies

Chongfeng Xie (xiechf.bri@chinatelecom.cn) received his Ph.D. degree in electronic engineering from Tsinghua University, China. He was a visiting scholar at the University of California, Los Angeles (UCLA) in 2009. Now he works in the IP and Future Network Research Center at China Telecom Beijing Research Institute (CTBRI). His research interests include network architecture and protocols, IPv6, SDN, NFV, and so on. He has published more than 20 papers and 10 Internet RFCs or drafts.

Jun Bi [SM] (junbi@tsinghua.edu.cn) received his B.S., M.S., and Ph.D. degrees in computer science from Tsinghua University, China. He was a postdoctoral scholar and a research scientist at Bell Labs. Currently he is Changjiang Scholar Distinguished Professor and Director of the Network Architecture Research Division, Institute for Network Sciences and Cyberspace, Tsinghua University. His research interests include Internet architecture and protocols. He has published over 200 papers and 4 Internet RFCs. He is a Senior Member of ACM.

Heng Yu (hengyu1213@163.com) is an undergraduate student at the Institute for Network Sciences and Cyberspace, Tsinghua University. His research fields include software-defined networking and network function virtualization.

Chen Li (lichen.bri@chinatelecom.cn) is a senior engineer of the IP and Future Network Research Center, China Telecom Corporation Limited Beijing Research Institute. His research fields include next generation networks, software-defined networking, and network functions virtualization. He has led or participated in several projects supported by the government.

Chen Sun (c-sun14@mails.tsinghua.edu.cn) is a Ph.D. student at the Institute for Network Sciences and Cyberspace, Tsinghua University. His research fields include software-defined networking and network function virtualization.

Qing Liu (liuqing.lq1988@gmail.com) received her M.S. degree from the School of Information Science and Engineering, Southeast University, China. She studied for one year as an exchange student at Aachen University, Germany. She is now working in the Network Operation Center, China Telecom Jiangsu branch. Her research fields include network architecture, network management and operation, and network functions virtualization, among others.

Zhilong Zheng (zhengzl15@mails.tsinghua.edu.cn) is a Ph.D. student at the Institute for Network Sciences and Cyberspace, Tsinghua University. His research fields include software-defined networking and network functions virtualization.

Shucheng Liu (liushucheng@huawei.com) is currently the research project manager for SDN NBI with Huawei Technologies Co. Ltd. His research interests include NFV/SDN, IPv6, ICN, and IoT. He has actively contributed in IETF, IRTF, ETSI, and served as a referee/TPC member for several conferences and journals in the computer networking area.